

COMMUNICATIONS

CACM.ACM.ORG

OF THE

ACM

05/2011 VOL.54 NO.5

Brain-Computer Interfaces

Weapons of
Mass Assignment

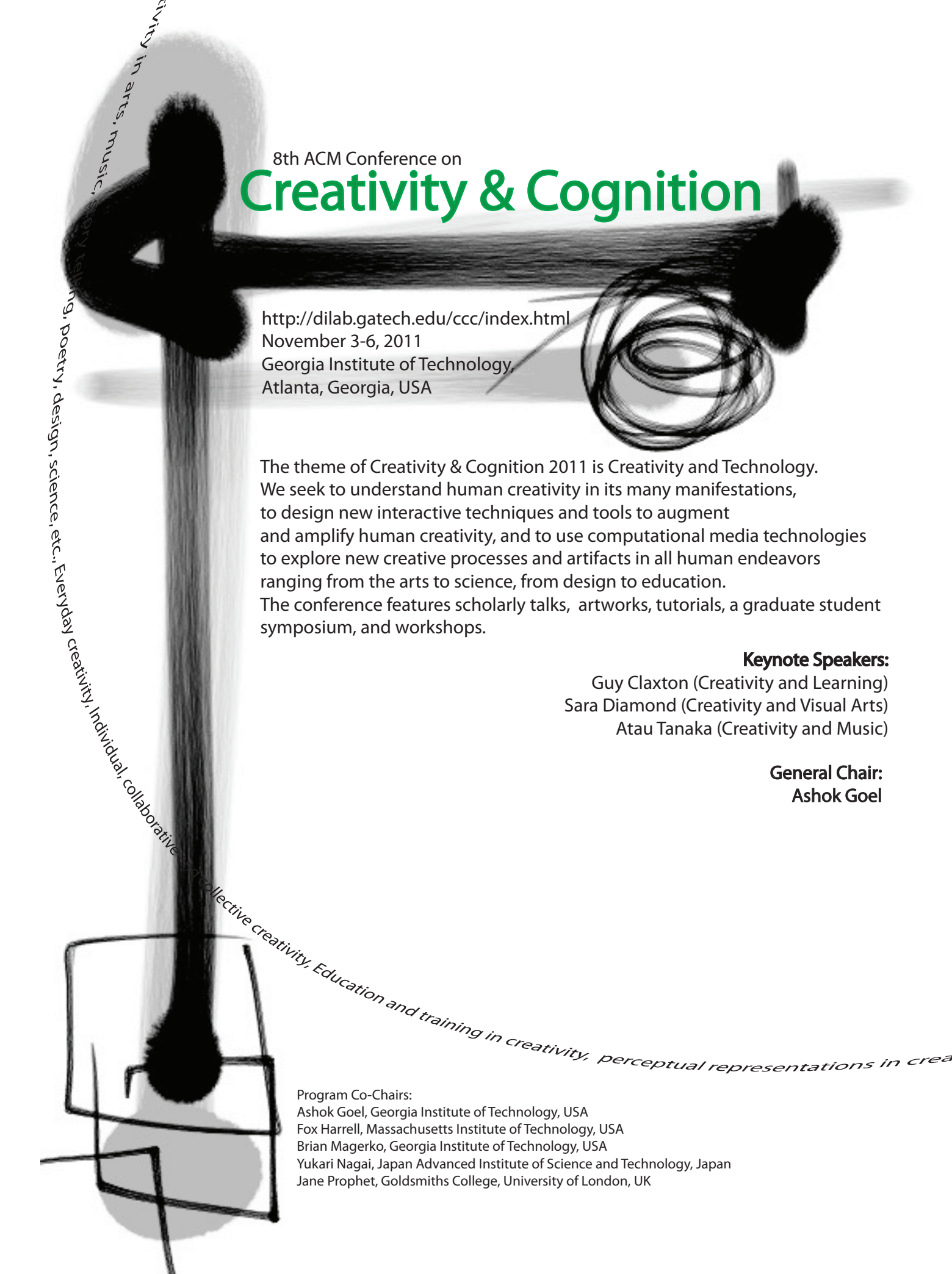
Online Advertising,
Behavioral Targeting,
and Privacy

The Future of
Microprocessors

I, Domestic Robot

Proving Program
Termination





8th ACM Conference on Creativity & Cognition

<http://dilab.gatech.edu/ccc/index.html>

November 3-6, 2011

Georgia Institute of Technology,
Atlanta, Georgia, USA

The theme of Creativity & Cognition 2011 is Creativity and Technology. We seek to understand human creativity in its many manifestations, to design new interactive techniques and tools to augment and amplify human creativity, and to use computational media technologies to explore new creative processes and artifacts in all human endeavors ranging from the arts to science, from design to education. The conference features scholarly talks, artworks, tutorials, a graduate student symposium, and workshops.

Keynote Speakers:

Guy Claxton (Creativity and Learning)
Sara Diamond (Creativity and Visual Arts)
Atau Tanaka (Creativity and Music)

General Chair:

Ashok Goel

Program Co-Chairs:

Ashok Goel, Georgia Institute of Technology, USA
Fox Harrell, Massachusetts Institute of Technology, USA
Brian Magerko, Georgia Institute of Technology, USA
Yukari Nagai, Japan Advanced Institute of Science and Technology, Japan
Jane Prophet, Goldsmiths College, University of London, UK



THE ACM A. M. TURING AWARD

by the community ♦ from the community ♦ for the community



ACM, Intel and Google congratulate

LESLIE G. VALIANT

for transformative contributions to the theory of computation, including the theory of probably approximately correct (PAC) learning, the complexity of enumeration and of algebraic computation, and the theory of parallel and distributed computing.



“Leslie Valiant’s research in the theory of computation has revolutionized both machine learning and artificial intelligence, making machines almost think. His approach invites comparison with Alan Turing himself—a novel formulation starting from a deep fundamental insight. Intel is pleased to support this year’s ACM Turing Award.”

Shekhar Borkar
Intel Fellow, Microprocessor Technology Lab
Intel Corporation

For more information see www.intel.com/research.



“Google joins in honoring Leslie Valiant for his profound impact on computer science research and his inspired innovations in machine learning, an area of growing importance in computing. We are pleased to sponsor this award, which motivates and recognizes the great advances in computing that together have had such a beneficial impact on the world.”

Alfred Spector
Vice President, Research and Special Initiatives
Google Inc.

For more information, see <http://www.google.com/corporate/index.html> and <http://research.google.com/>.



Financial support for the ACM A. M. Turing Award is provided by Intel Corporation and Google Inc.

Departments

- 5 **Editor's Letter**
Technology Has Social Consequences
By Moshe Y. Vardi
-
- 6 **Letters To The Editor**
Preserve Privacy in Statistical Correlations
-
- 9 **In the Virtual Extension**
-
- 10 **BLOG@CACM**
Stonebraker on Data Warehouses
Data warehouses are not only increasing in size and complexity, but also in their importance to business. Michael Stonebraker shares 10 key ideas on the topic.
-
- 12 **CACM Online**
Let ACM Help You Find Your Next Job 'Online'
By Scott E. Delman
-
- 41 **Calendar**
-
- 118 **Careers**

Last Byte

- 120 **Puzzled**
Games, Roles, Turns
By Peter Winkler



About the Cover: Technologies that allow users to communicate or control devices by 'thought' were once the stuff of science fiction. Today, however, extraordinary strides in the field of brain-computer interfaces (BCI) are very real and having a huge impact on people with neuromuscular disorders and injuries.

This month's cover story by Dennis McFarland and Jonathan Wolpaw (p. 60) explores the science that translates electric signals produced by the brain into action or words. Photograph by Justin Stephens.

News

- 13 **Sorting Through Photos**
Teaching computers to understand pictures could lead to search engines capable of identifying and organizing large datasets of visual information.
By Neil Savage
-
- 16 **I, Domestic Robot**
With recent advances in laser rangefinders, faster algorithms, and open source robotic operating systems, researchers are increasing domestic robots' semantic and situational awareness.
By Gregory Goth
-
- 18 **Data Optimization in Developing Nations**
Artificial intelligence and machine learning could expand access to health care, improve the quality of education, and respond effectively to natural disasters in the developing world.
By Leah Hoffmann
-
- 22 **Deus Ex Machina**
Computational metaphysics is helping philosophers answer age-old questions, such as whether God exists.
By Marina Krakovsky
-
- 23 **Web Science Meets Network Science**
A pair of divergent scientific communities discusses their similarities and differences, and search for common ground.
By Alex Wright

Viewpoints

- 25 **Economic and Business Dimensions**
Online Advertising, Behavioral Targeting, and Privacy
Studying how privacy regulation might impact economic activity on the advertising-supported Internet.
By Avi Goldfarb and Catherine E. Tucker
-
- 28 **Education**
Reaching Learners Beyond our Hallowed Halls
Rethinking the design of computer science courses and broadening the definition of computing education both on and off campus.
By Brian Dorn
-
- 31 **Law and Technology**
Bell Labs and Centralized Innovation
Replaying the long-term costs of monopolized innovation.
By Tim Wu
-
- 34 **Interview**
An Interview with Steve Furber
Steve Furber, designer of the seminal BBC Microcomputer System and the widely used ARM microprocessor, reflects on his career.
By Jason Fitzpatrick
-
- 40 **Viewpoint**
The Importance of Reviewing the Code
Highlighting the significance of the often overlooked underlying software used to produce research results.
By Juan A. Añel

Practice



- 44 **The One-Second War**
Finding a lasting solution to the leap seconds problem has become increasingly urgent.
By Poul-Henning Kamp
- 49 **Mobile Application Development: Web vs. Native**
Web apps are cheaper to develop and deploy than native apps, but can they match the native user experience?
By Andre Charland and Brian LeRoux
- 54 **Weapons of Mass Assignment**
A Ruby on Rails app highlights some serious, yet easily avoided, security vulnerabilities.
By Patrick McKenzie

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

Contributed Articles

- 60 **Brain-Computer Interfaces for Communication and Control**
The brain's electrical signals enable people without muscle control to physically interact with the world.
By Dennis J. McFarland and Jonathan R. Wolpaw
- 67 **The Future of Microprocessors**
Energy efficiency is the new fundamental limiter of processor performance, way beyond numbers of processors.
By Shekhar Borkar and Andrew A. Chien
- 78 **Privacy-Preserving Network Forensics**
Privacy-preserving attribution of IP packets can help balance forensics with an individual's right to privacy.
By Mikhail Afanasyev, Tadayoshi Kohno, Justin Ma, Nick Murphy, Stefan Savage, Alex C. Snoeren, and Geoffrey M. Voelker
- VE** **Challenges and Business Models for Mobile Location-based Services and Advertising**
Mobile advertising will become more pervasive and profitable, but not before addressing key technical and business challenges.
By Subhankar Dhar and Upkar Varshney
- VE** **Is Open Source Security a Myth?**
What does vulnerability and patch data say?
By Guido Schryen
- VE** **Invisible Work in Standard Bibliometric Evaluation of Computer Science**
Most of a computer scientist's production can go uncounted if a standard bibliographic service is used.
By Jacques Wainer, Siome Goldenstein, and Cleo Billa

Review Articles

- 88 **Proving Program Termination**
In contrast to popular belief, proving termination is not always impossible.
By Byron Cook, Andreas Podelski, and Andrey Rybalchenko

Research Highlights

- 100 **Technical Perspective**
Complex Financial Products: Caveat Emptor
By David C. Parkes
- 101 **Computational Complexity and Information Asymmetry in Financial Products**
By Sanjeev Arora, Boaz Barak, Markus Brunnermeier, and Rong Ge
- 108 **Technical Perspective**
Images Everywhere Looking for Models
By Guillermo Sapiro
- 109 **Self-Similarity-based Image Denoising**
By Antoni Buades, Bartomeu Coll, and Jean-Michel Morel



Association for Computing Machinery
Advancing Computing as a Science & Profession



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

Executive Director and CEO

John White

Deputy Executive Director and COO

Patricia Ryan

Director, Office of Information Systems

Wayne Graves

Director, Office of Financial Services

Russell Harris

Director, Office of Marketing and Membership

David M. Smith

Director, Office of SIG Services

Donna Cappel

Director, Office of Publications

Bernard Rous

Director, Office of Group Publishing

Scott E. Delman

ACM COUNCIL

President

Alain Chesnais

Vice-President

Barbara G. Ryder

Secretary/Treasurer

Alexander L. Wolf

Past President

Wendy Hall

Chair, SGB Board

Vicki Hanson

Co-Chairs, Publications Board

Ronald Boisvert and Jack Davidson

Members-at-Large

Vinton G. Cerf;

Carlo Ghezzi;

Anthony Joseph;

Mathai Joseph;

Kelly Lyons;

Mary Lou Soffa;

Salil Vadhan

SGB Council Representatives

Joseph A. Konstan;

G. Scott Owens;

Douglas Terry

PUBLICATIONS BOARD

Co-Chairs

Ronald F. Boisvert; Jack Davidson

Board Members

Nikil Dutt; Carol Hutchins; Joseph A. Konstan;

Ee-Peng Lim; Catherine McGeoch;

M. Tamer Ozsu; Holly Rushmeier;

Vincent Shen; Mary Lou Soffa

ACM U.S. Public Policy Office

Cameron Wilson, Director

1828 L Street, N.W., Suite 800

Washington, DC 20036 USA

T (202) 659-9711; F (202) 667-1066

Computer Science Teachers Association

Chris Stephenson

Executive Director

2 Penn Plaza, Suite 701

New York, NY 10121-0701 USA

T (800) 401-1799; F (541) 687-1840

Association for Computing Machinery (ACM)

2 Penn Plaza, Suite 701

New York, NY 10121-0701 USA

T (212) 869-7440; F (212) 869-0481

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

publisher@cacm.acm.org

Executive Editor

Diane Crawford

Managing Editor

Thomas E. Lambert

Senior Editor

Andrew Rosenbloom

Senior Editor/News

Jack Rosenberger

Web Editor

David Roman

Editorial Assistant

Zarina Strakhan

Rights and Permissions

Deborah Cotton

Art Director

Andrij Borys

Associate Art Director

Alicia Kubista

Assistant Art Directors

Mia Angelica Balaquiot

Brian Greenberg

Production Manager

Lynn D'Addesio

Director of Media Sales

Jennifer Ruzicka

Public Relations Coordinator

Virginia Gold

Publications Assistant

Emily Williams

Columnists

Alok Aggarwal; Phillip G. Armour;

Martin Campbell-Kelly;

Michael Cusumano; Peter J. Denning;

Shane Greenstein; Mark Guzdial;

Peter Harsha; Leah Hoffmann;

Mari Sako; Pamela Samuelson;

Gene Spafford; Cameron Wilson

CONTACT POINTS

Copyright permission

permissions@cacm.acm.org

Calendar items

calendar@cacm.acm.org

Change of address

acmhlp@cacm.org

Letters to the Editor

letters@cacm.acm.org

WEB SITE

<http://cacm.acm.org>

AUTHOR GUIDELINES

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

ADVERTISING

ACM ADVERTISING DEPARTMENT

2 Penn Plaza, Suite 701, New York, NY

10121-0701

T (212) 869-7440

F (212) 869-0481

Director of Media Sales

Jennifer Ruzicka

jen.ruzicka@hq.acm.org

Media Kit acmm mediasales@acm.org

EDITORIAL BOARD

EDITOR-IN-CHIEF

Moshe Y. Vardi

eic@cacm.acm.org

NEWS

Co-chairs

Marc Najork and Prabhakar Raghavan

Board Members

Hsiao-Wuen Hon; Mei Kobayashi;

William Pulleyblank; Rajeev Rastogi;

Jeannette Wing

VIEWPOINTS

Co-chairs

Susanne E. Hambrusch; John Leslie King;

J Strother Moore

Board Members

P. Anandan; William Aspray;

Stefan Bechtold; Judith Bishop;

Stuart I. Feldman; Peter Freeman;

Seymour Goodman; Shane Greenstein;

Mark Guzdial; Richard Heeks;

Rachelle Hollander; Richard Ladner;

Susan Landau; Carlos Jose Pereira de Lucena;

Beng Chin Ooi; Loren Terveen

PRACTICE

Chair

Stephen Bourne

Board Members

Eric Allman; Charles Beeler; David J. Brown;

Bryan Cantrill; Terry Coatta;

Stuart Feldman; Benjamin Fried;

Pat Hanrahan; Marshall Kirk McKusick;

Erik Meijer; George Neville-Nel;

Theo Schlossnagle; Jim Waldo

The Practice section of the CACM

Editorial Board also serves as

the Editorial Board of *Queue*.

CONTRIBUTED ARTICLES

Co-chairs

Al Aho and Georg Gottlob

Board Members

Yannis Bakos; Elisa Bertino; Gilles Brassard;

Alan Bundy; Peter Buneman; Andrew Chien;

Peter Druschel; Blake Ives; James Larus;

Igor Markov; Gail C. Murphy; Shree Nayar;

Lionel M. Ni; Sriram Rajamani;

Marie-Christine Rousset; Avi Rubin;

Krishnan Sabnani; Fred B. Schneider;

Abigail Sellen; Ron Shamir; Marc Snir;

Larry Snyder; Veda Storey; Manuela Veloso;

Michael Vitale; Wolfgang Wahlster;

Andy Chi-Chih Yao

RESEARCH HIGHLIGHTS

Co-chairs

David A. Patterson and Stuart J. Russell

Board Members

Martin Abadi; Stuart K. Card; Jon Crowcroft;

Shafi Goldwasser; Monika Henzinger;

Maurice Herlihy; Dan Huttenlocher;

Norm Jouppi; Andrew B. Kahng;

Gregory Morrisett; Michael Reiter;

Mendel Rosenblum; Ronitt Rubinfeld;

David Salesin; Lawrence C. Saul;

Guy Steele, Jr.; Madhu Sudan;

Gerhard Weikum; Alexander L. Wolf;

Margaret H. Wright

WEB

Co-chairs

James Landay and Greg Linden

Board Members

Gene Golovchinsky; Marti Hearst;

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



ACM Copyright Notice

Copyright © 2011 by Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to publish from permissions@acm.org or fax (212) 869-0481.

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

Subscriptions

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

ACM Media Advertising Policy

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

Single Copies

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

COMMUNICATIONS OF THE ACM

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

POSTMASTER

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



Association for
Computing Machinery



Printed in the U.S.A.



DOI:10.1145/1941487.1941488

Moshe Y. Vardi

Technology Has Social Consequences

A conference program committee (PC) member received a paper for review. He distributed the manuscript to his research group to “solicit their opinions

of the paper” and the group embarked on improving the results of the paper under review. The research group then submitted their own paper to another conference, their submission occurring three months before the first paper was to be presented at a conference. When eventually confronted (the short gap between the appearance of the two papers triggered questions), the PC member responded with “Was that wrong? Should I have not done that?” (The reader may want to search for this phrase on YouTube.)

Amazingly, this PC member was not aware that a conference paper submission constitutes privileged communication. In theory, reviewers should immediately “forget” what they have read. For reviewers to use such privileged material for their own work immediately creates a blatant conflict of interest. How did this PC member, a full professor in a respected university, not know such a fundamental rule of scholarly reviewing?

To understand how the ethics of program committees has declined, one must review the history of computer science program committees over the last 50 years. Until the mid-1990s, program committees met in face-to-face meetings. This had two significant consequences. First, PC members bore the cost of attending PC meetings, leading them to be careful with accepting PC service commitments. It was rare then for one to serve on more than one PC per year. Sec-

ond, junior PC members had a chance to interact intensively with senior PC members. There was nontrivial social pressure on junior PC members to demonstrate their competence in PC meetings. Thus, PC service provided important socializing experience, where unwritten norms and customs were learned by observation.

With the emergence of the World-Wide Web in the mid-1990s, physical PC meetings suddenly seemed wasteful, as it became possible to conduct virtual meetings without incurring travel expenses and headaches. Conference-management software tools emerged and many communities abandoned physical meetings in favor of virtual ones. I was very much in favor of this change back then! It took, however, a few years for the adverse consequences of this change to become visible.

Economists would tell you that a commodity priced too low would end up being overconsumed. PC service is a commodity with positive utility. Our community views PC service as a form of professional recognition; in fact, it is one of the few markers of professional recognition available to junior researchers. Since the “cost” of PC service has dropped with the switch to virtual meetings, “consumption” has gone up. Indeed, it is quite common today to see researchers serving on several PCs per year.

Of course, one cannot expect the same level of effort from someone

who serves on one PC per year as compared to someone who serves on multiple PCs per year. Indeed, in the 1980s it was typical to see every submission read by five to six PC members, today the norm is often three to four reviewers for submission. Furthermore, these reviewers are often not PC members but “subreviewers.” In fact, while the concept of subreviewer was originally developed with the purpose of bringing additional expertise to PCs, today it is viewed as an opportunity to train junior researchers in the art of paper reviewing. Thus, the role of a PC member seems to have evolved from that of a reviewer to that of “review orchestrator.”

What has been the outcome of this development? Many of us are quite familiar with this outcome. The quality of conference reviewing has declined and the selection process has become far more random. Two years ago, I wrote in this space about “Conferences vs. Journals in Computing Research.” The declining quality of conference reviewing was one of triggers that spurred me to write that editorial.

The loss in quality of conference reviewing is just one result of the move to virtual PC meetings. Another outcome is the loss of socialization that took place in PC meetings. It is this lost socialization that contributed to a senior researcher being ignorant of one of the most basic rules of scholarly reviewing.

We all know that technology has social consequences. This applies to us as well. The switch to purely virtual meetings did not serve us well. Many communities have already realized that and are combining virtual and physical meetings to merge the strengths of both formats. Such practices, I believe, should be widely adopted. Technology can be managed!

Moshe Y. Vardi, EDITOR-IN-CHIEF

Preserve Privacy in Statistical Correlations

MANY THANKS FOR Cynthia Dwork's article "A Firm Foundation for Private Data Analysis" (Jan. 2011), explaining why, in trying to formalize what is perfect privacy, we cannot use the late University of Stockholm economist Tore E. Dalenius's criterion that asking allowed queries of a statistical database, we should not be able to learn new (private) information about a particular individual. When preparing to discuss Dwork's article at a recent colloquium in our computer science department, we came up with an even simpler explanation of such an impossibility:

One important purpose of collecting statistical data is to help identify correlations between, say, weight and blood pressure. Suppose, for example, it turns out that blood pressure is equal to weight, and we know that person A (not in this database) weighs 180 pounds. Without the database, A's blood pressure might be private, but once we learn the perfect correlation from it, we can conclude that A's blood pressure is 180.

In real life, we never see such perfect correlation, but, by analyzing the database and discovering some correlation, we know more about the probability of different values of blood pressure than we would otherwise know.

Vladik Kreinovich and Luc Longpre,
El Paso, TX

Recall the Lost Frontiers of Virtual Worlds

The Future Tense essay "Rebirth of Worlds" (Dec. 2010) lamented the demise of historic, online interactive 3D destinations. Since 1997 when they first appeared on the Web, virtual worlds have inspired artists, engineers, and scientists alike to explore and build the emerging frontiers of cyberspace. As Rumilisoun (a.k.a. William Sims Bainbridge) wrote, despite the wonderful destinations across entertainment, education, and community, we are left to ask, "How can I still get there?"

What came through clearly in "Rebirth of Worlds" is the author's nostalgia for the experience of those worlds—their realities and possibilities. Such compelling emotional, perceptual, existential content may indeed be gone for good. Loss of an appealing game world is lamentable, but it is even more disheartening with engineering and scientific content, where we require the durability and reproducibility of our interactive 3D digital content—models, behaviors, worlds, and scenarios—for decades to come.

Enterprise-scale adopters, along with many others, also feel the pain of virtual-world babelization, as developing and maintaining innovative assets like worlds, avatars, and business logic across platforms become increasingly complex. Content models and network protocols are fragmented, making it difficult to create integrated information spaces and a compelling user experience. In the tumult of proprietary virtual-world technology, lack of reuse is a major obstacle to achieving improved efficiencies and economies of scale.

In the face of this market churn is a proven path for interactive 3D environments that includes royalty-free, extensible content models designed for the Web and semantic integration. Consumers and computer professionals alike should therefore demand and participate in the development of international standards needed to raise the greatest common denominator of future-proof 3D content.

Nicholas F. Polys (president of
Web3D Consortium), Blacksburg, VA

Let Implementation Semantics Unlock the Pop

The lock-free pop operation Nir Shavit described in his article "Data Structures in the Multicore Age" (Mar. 2011) depends on the semantics of the Java implementation in an important way. The push operation allocates a new node object during the call, and it is this object that is placed on the stack.

In addition, the assignment of `oldTop` at line 13 creates a reference to the top node, keeping it alive until the return from the function.

This is of interest because if any of these constraints is not true, the pop operation would not work. In particular, if one would naively implement a push-and-pop mechanism along these lines in a language like C++, and let the clients provide the object to be pushed, and returned that object to the clients when the pop occurred, the program would be wrong. This is because after fetching `oldTop` (line 13) and `newTop` (line 17) other threads could remove the top node, remove or push other nodes, then push the top node again. The `compareAndSet` would then succeed, even though `newTop` was no longer the correct new value. Similarly, if the implementation allocated a node in push, and freed it in pop, the program would be wrong because the freed-node storage might be reused in a subsequent push, leading to the same error.

The Java implementation also involves hidden costs, including allocation and garbage collection of the node objects and concurrency control required in the memory-allocation system to make it work. These costs must be considered, as they are essential to the correctness of the program. Be warned about not using apparently identical algorithms that do not satisfy the hidden constraints.

Marc Auslander,
Yorktown Heights, NY

Protect Software Consumers Like Everyone Else

I regret that Joel F. Brenner responded to my letter to the editor "Hold Manufacturers Liable" (Feb. 2011) concerning his Viewpoint "Why Isn't Cyberspace More Secure?" (Nov. 2010) with two strawman arguments and one outright misstatement.

Brenner said software "is sold pursuant to enforceable contracts." As the Viewpoint "Do You Own the Soft-

ware You Buy?” by Pamela Samuelson (Mar. 2011) made clear, software is not “sold.” Every EULA insists software is licensed and only the media on which it is recorded are sold; a series of court decisions, of which the *Vernor v. Autodesk* decision Samuelson cited is the most recent and one of the most conclusive, have upheld this stance.

This mischaracterization by Brenner is one of the keys to understanding how manufacturers of such shoddy goods get off essentially scot-free. If software were actually sold, the argument that it should be exempt from the protections of the Uniform Commercial Code would be much more difficult to maintain, in addition to other benefits thoroughly discussed elsewhere (including by Samuelson in her column).

Even though EULAs have been held enforceable, such a determination comes at the expense of the consumer. Almost without exception, EULAs have the effect of stripping the consumer of essentially all reasonable rights and expectations, compared with other goods and services. And while click-through and shrink-wrap EULAs have indeed been found to be enforceable, many reasonable people (including me) believe it should not be the case, since the vast majority of consumers do not read these “contracts” and do not understand their consequences. Brenner apparently does not consider them a significant problem.

Finally, Brenner simply reiterated his assertion that “Congress shouldn’t decide what level of imperfection is acceptable.” I agree. There are basic consumer protections that apply to all other goods, as embodied in the UCC. Neither a further act of Congress nor detailed specifications of product construction are required to give consumers the right to expect, say, a stove, properly used and maintained, will not burn down their house. The corresponding right of freedom from gross harm, like the other protections of the UCC, is not available for software, though it and they should be; Brenner apparently disagrees.

I emphasized good engineering practices in my February letter not because (as Brenner seems to believe) I thought they were sufficient to guarantee a reasonable level of product quality, but because they are well-established

means toward the end of meeting the basic standards of non-harm and reliability taken as a given for all other products. In any case, Brenner did not say why he thinks a different process should be used for setting functional safety and reliability standards for software than for other consumer goods. Simply asserting “software is different” is not a reasoned argument.

L Peter Deutsch, Palo Alto, CA

Author’s Response:

Thanks to Deutsch for correcting my error. Software is of course licensed rather than sold. As Deutsch says, this is why UCC product-liability standards for purchased goods haven’t improved software quality. But his point strengthens my argument. I was explaining, not defending, the status quo, which is lamentable precisely because liability is weak. I cannot fathom why Deutsch thinks I’m indifferent to higher engineering standards for software. They represent the only basis on which a liability regime can be founded, even for licensed products.

Joel F. Brenner, Washington, D.C.

Correction

Sarah Underwood’s news story “British Computer Scientists Reboot” (Apr. 2011) incorrectly attributed statements by King’s College London professor Michael Luck to King’s College London professor Andrew Jones. This has been corrected in the online article. We apologize for the error. 

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

© 2011 ACM 0001-0782/11/05 \$10.00

Coming Next Month in COMMUNICATIONS

*A Profile of ACM’s 2010
A.M. Turing Award Recipient,
Les Valiant.*

*Advancing the State
of Home Networking*

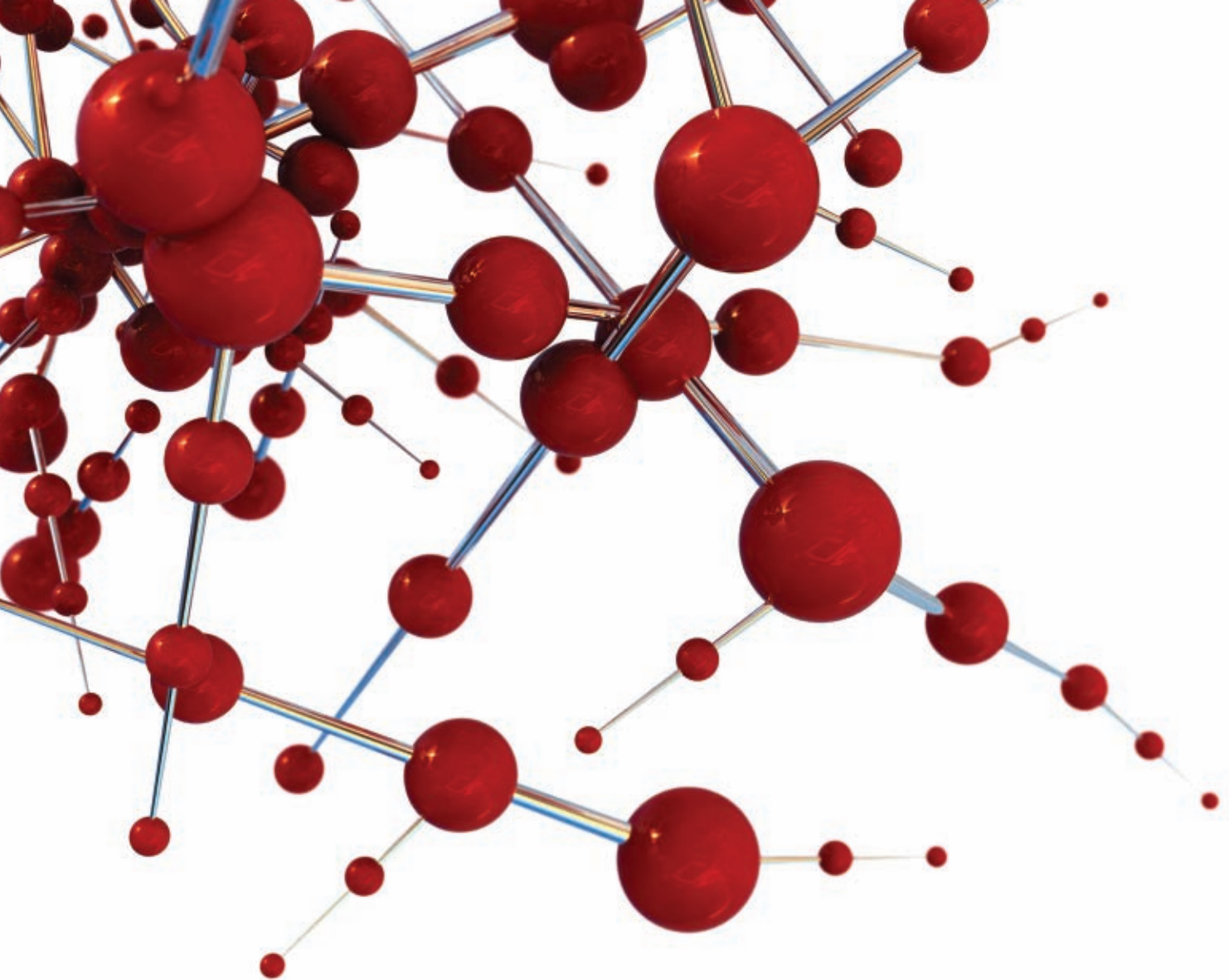
**Also, the latest news on digital
journalism, flexible plastic displays,
and fruit fly-inspired network
breakthroughs.**



ACM’s
interactions
magazine explores
critical relationships
between experiences, people,
and technology, showcasing
emerging innovations and industry
leaders from around the world
across important applications of
design thinking and the broadening
field of the interaction design.
Our readers represent a growing
community of practice that
is of increasing and vital
global importance.

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





**CONNECT WITH OUR
COMMUNITY OF EXPERTS.**

www.reviews.com



Association for
Computing Machinery

Reviews.com

They'll help you find the best new books
and articles in computing.

Computing Reviews is a collaboration between the ACM and Reviews.com.

DOI:10.1145/1941487.1941490

In the Virtual Extension

To ensure the timely publication of articles, Communications created the Virtual Extension (VE) to expand the page limitations of the print edition by bringing readers the same high-quality articles in an online-only format. VE articles undergo the same rigorous review process as those in the print edition and are accepted for publication on merit. The following synopses are from articles now available in their entirety to ACM members via the Digital Library.

contributed article

DOI: 10.1145/1941487.1941515

Challenges and Business Models for Mobile Location-based Services and Advertising

Subhankar Dhar and Upkar Varshney

Location-based services have attracted considerable attention due to their potential to transform mobile communications and the potential for a range of highly personalized and context-aware services. Since the days of early location-tracking functionalities introduced in Japan in 2001 and in some U.S. networks, location-based services have made considerable progress.

The potential for location-based services is evident from powerful and ubiquitous wireless devices that are growing in popularity. Many surveys predict billions of dollars in revenues for mobile advertising. Mobile network operators are well positioned to take up a significant percentage of this advertising revenue as they negotiate deals with content providers. Recent deals between media companies, advertising agencies, and the Internet/software industry also demonstrate significant optimism for future growth.

However, there are many challenges that have slowed down the deployment, offering, and wide-scale adoption of location-based services. The challenges include emerging technologies, suitable applications, and business models. This article addresses both technical- and business-related challenges in location-based services, specifically in mobile advertising. The authors explore how location-based mobile advertising can generate revenues and sustain successful business models. However, they are quick to note that while mobile advertising will become more pervasive and profitable, it will not happen before key technical and business challenges are addressed.

contributed article

DOI: 10.1145/1941487.1941516

Is Open Source Security a Myth?

Guido Schryner

During the past few decades we became accustomed to acquiring software by procuring licenses for a proprietary or binary-only immaterial object. We regard software as a product we have to pay for, just as we would pay for material objects. However, in more recent years, this widely cultivated habit has begun to be accompanied by a software model characterized by software that comes with a compilable source code. This type of software is referred to as open source software (OSS).

While there is consensus that opening up source code to the public increases the number of reviewers, the impact of open source on software security remains controversial. While the security discussion is rife with beliefs and guesses, only a few quantitative models and some empirical studies appear in the literature; and most of those studies examine only one or a few packages.

This article presents a comprehensive empirical investigation of published vulnerabilities and patches of 17 widely deployed open source and closed source software packages. The empirical analysis uses comprehensive vulnerability data contained in the NIST National Vulnerability Database and a newly compiled data set of vulnerability patches. Based on these comprehensive data sets, this study is capable of providing empirical evidence that open source and closed source software development do not significantly differ in terms of vulnerability disclosure and vendors' patching behavior—a phenomenon that has been widely assumed, but hardly investigated.

contributed article

DOI: 10.1145/1941487.1941517

Invisible Work in Standard Bibliometric Evaluation of Computer Science

Jacques Wainer, Siome Goldenstein, and Cleo Billa

Multidisciplinary committees routinely make strategic decisions, rule on subjects ranging from faculty promotion to grant awards, and rank and compare scientists. Though they may use different criteria for evaluations in subjects as disparate as history and medicine, it seems logical for academic institutions to group together mathematics, computer science, and electrical engineering for comparative evaluation by these committees.

These evaluations will be more frequent as the number of scientists increases. Since the number of funding sources grows more slowly, and research practices vary among different subjects, using the same criteria in different areas may produce notable injustices. The ongoing discussion on CS research evaluation helps build the case for the CS community defending itself from expected injustices in future comparative evaluations.

Traditional assessment criteria are based on Thomson Reuters' Web of Science (WoS) indexing service, quantifying the production and number of citations of individual scientists, university departments, whole universities, countries, and scientific areas.

Here, the authors provide some quantitative evidence of unfairness, defining researchers' invisible work as an estimation of all their scientific publications not indexed by WoS or Scopus. Thus, the work is not counted as part of scientists' standard bibliometric evaluations. To compare CS invisible work to that of physics, mathematics, and electrical engineering, they generated a controlled sample of 50 scientists from each of these fields and focused on the distribution of invisible work rate for each of them using statistical tests.

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



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

DOI:10.1145/1941487.1941491

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

Stonebraker on Data Warehouses

Data warehouses are not only increasing in size and complexity, but also in their importance to business. Michael Stonebraker shares 10 key ideas on the topic.



Michael Stonebraker
From "My Top 10
Assertions About
Data Warehouses"

<http://cacm.acm.org/blogs/blog-cacm/98136>

Data warehouses, business intelligence, business analytics, and complex analytics are the subject of increasingly intense marketing wars, seemingly accelerated by Oracle's introduction of the Exadata appliance. Here is my spin on the situation. Please note that I have a financial interest in several database companies, and may be biased in a number of ways. The reader should always keep this in mind.

1. Star and snowflake schemas are a good idea in the data warehouse world.

In short, data warehouses store a large collection of facts. The overwhelming majority of these facts are the "five Ws" (who, what, where, when, and why) along with a collection of attributes about the fact. For example, a typical retail organization stores facts about historical transactions. These facts include the customer (who), the retail store (where), the time of the sale (when), and the purchased product

(what), along with attributes of the sale (e.g., price, sales tax, credit card, etc.).

One should organize such data as shown in the figure here. Such a schema is called a star schema, with a central fact table and surrounding dimension tables. If stores are organized into divisions, then the star schema has another table between store and fact, and becomes a snowflake schema. Star and snowflake schemas are clean, simple, easy to parallelize, and usually result in very high-performance database management system (DBMS) applications.

If you are a data warehouse designer and come up with something other than a snowflake schema, you should probably rethink your design.

However, you will often come up with a design having a large number of attributes in the fact table; 40 attributes are routine and 200 are not uncommon. Current data warehouse administrators usually stand on their heads to make "fat" fact tables perform on current relational database management systems (RDBMSs).

2. Column stores will dominate the data warehouse market over time, replacing row stores.

The problem with row stores is they store data in the fact table, row by row, on disk. A typical business intelligence query requires half-a-dozen attributes or less (e.g., find me the average price of widgets by store by month for the past two years). A row store will read all 200 attributes, even though only six are required. In contrast, a DBMS that organizes data by column will read only the six required, a savings of a factor of 33.

Since fact tables are getting fatter over time as business analysts want access to more and more information, this architectural difference will become increasingly significant. Even when "skinny" fact tables occur or where many attributes are read, a column store is still likely to be advantageous because of its superior compression ability.

For these reasons, over time, column stores will clearly win.

3. The vast majority of data warehouses are not candidates for main memory or flash memory.

Data warehouses are increasing in size faster than storage is getting cheaper. Business analysts have an appetite for as much attribute data as they can get their hands on, and want to keep increasingly long periods of history. Hence, data warehouse problems are getting "net harder," not "net easier." Put differently, most data warehouses are measured in Gbytes today, Tbytes tomorrow, and Pbytes the next day.

4. Massively parallel processor (MPP) systems will be omnipresent in this market.

Massively parallel processor systems are the only kind of computer ar-

chitecture that will scale to Pbytes. All vendors, with a very few exceptions, are or will soon support MPP. Don't bet on anything that is not in the MPP camp.

5. “No knobs” is the only thing that makes any sense.

It is pretty clear that human operational costs dominate the cost of running a data warehouse. This is mainly the system administration and database administration that is involved in keeping a MPP system up and in managing a Pbyte-sized warehouse. Database administrator (DBA) costs include designing schemas, reprovisioning databases to add or drop resources, adding and dropping users, etc.

Almost all DBMSs have 100 or more complicated tuning “knobs.” This requires DBAs to be “4-star wizards” and drives up operating costs. The only thing that makes sense is to have a program that adjusts these knobs automatically. In other words, look for “no knobs” as the only way to cut down DBA costs.

6. Appliances should be “software only.”

In my 40 years of experience as a computer science professional in the DBMS field, I have yet to see a specialized hardware architecture—a so-called database machine—that wins.

In other words, one can buy general-purpose CPU cycles from the major chip vendors or specialized CPU cycles from a database machine vendor. Since the volume of the general-purpose vendors are 10,000 or 100,000 times the volume of the specialized vendors, their prices are an order of magnitude under those of the specialized vendor. To be a price-performance winner, the specialized vendor must be at least a factor of 20–30 faster.

I have never seen a specialized hardware architecture that is faster by this amount.

Put differently, I think database ap-

pliances are a packaging exercise—i.e., preconfigure general-purpose hardware and preload the DBMS on it. This results in a software-only appliance.

7. Hybrid workloads are not optimized by “one size fits all.”

If one has a workload that is part online transaction processing (OLTP) and part data warehouse, then he or she has two options: 1) Run a general-purpose RDBMS that stores both kinds of data; and 2) Run two systems, an OLTP engine and a data warehouse engine, coupled together with a high-speed interchange to move operational data into the data warehouse.

Row stores are not good at data warehouse applications (see #1). Column stores are optimized for data warehouses and are not good at OLTP. Hence, neither is a good candidate for a one-size-fits-all implementation. Instead, there are a number of interesting new ideas to accelerate OLTP, including main-memory SQL engines, main memory caches, and flash systems. When coupled with a column store in a two-system configuration, I assert the result will be a factor of 50 or so faster than solution 1.

8. Essentially all data warehouse installations want high availability (HA).

If there is data corruption in a 10 Tbyte warehouse, recovering the database from the database log will take a very long time. Of course, the exact amount of time depends on log formats, number of disks allocated to the log, and so on. However, restoring a 10 Tbyte dump from 10 disks is likely to take hours, not minutes, and then one has to perform a roll forward from the dump. Hardly anybody is interested in taking this kind of downtime.

Instead, most installations use replication and fail over to a replica if there is data corruption. Then, the corrupted copy can be rebuilt as a background

task. Hence, HA is used for recovery, not the DBMS log. Obviously, this requires the DBMS to support HA; otherwise, it is a manual DBA hassle to accomplish the same thing in user code.

9. DBMSs should support online re-provisioning.

Not always, but often, I hear a request for online reprovisioning. In other words, one initially allocates 10 nodes to accomplish warehouse processing. The load later rises, and the desire is to allocate 20 nodes to the task originally done by 10. This requires the database to be repartitioned over double the number of nodes.

Hardly anybody wants to take the required amount of downtime to dump and reload the DBMS. A much better solution is for the DBMS to support reprovisioning, without going offline.

10. Virtualization often has performance problems in a DBMS world.

I hear many users say their long-term goal is to move to the cloud, whether using the public cloud or inside the firewall on “an enterprise cloud.” Here, a collection of servers is allotted to several-to-many DBMS applications inside an enterprise firewall. Often, such systems are managed by using virtualization software to present virtual nodes to the DBMS and its applications.

My experience is that CPU resources can be virtualized with modest overhead (say, 10%). However, data warehouses entail disk-based data. In this world, all MPP DBMSs “move the query to the data.” Obviously, this requires knowing the physical data distribution. Virtualization will destroy this knowledge, and turn what were originally reads to a local disk into reads to non-local disks. In other words, local I/O gets replaced by remote I/O, with an obvious significant performance hit.

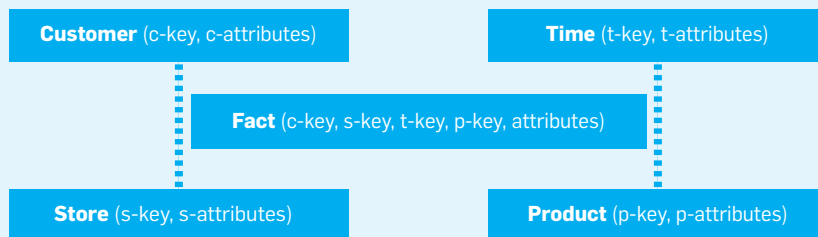
Until better and cheaper networking makes remote I/O as fast as local I/O at a reasonable cost, one should be very careful about virtualizing DBMS software.

Of course, the benefits of a virtualized environment are not insignificant, and they may outweigh the performance hit. My only point is to note that virtualizing I/O is not cheap. ■

Michael Stonebraker is an adjunct professor at the Massachusetts Institute of Technology.

© 2011 ACM 0001-0782/11/05 \$10.00

A Diagram of a Star Schema.





DOI:10.1145/1941487.1941492

Scott E. Delman

Let ACM Help You Find Your Next Job 'Online'

According to a Harris Interactive Poll of over 2,400 hiring managers and human resource professionals conducted in late 2010, recruiting in the information technology and technology sectors is expected to increase 26% and 19% over 2010 respectively. This is not only good news for the economy in general, but excellent news for our industry and ACM's membership specifically. But even though hiring and recruiting are on an upswing, finding the right position and the perfect match between candidates and employers is not an easy task.

There are literally hundreds of online resources posting jobs in the technology sector and academia with hundreds of thousands of candidates competing for those positions worldwide. However, for the most coveted jobs and for those organizations looking to hire the most experienced and talented employees, faculty, managers, and executives few online resources are more valuable than ACM's own Career and Job Center (<http://jobs.acm.org>), which is the engine driving the Careers section of *Communications* Web site.

This site provides basic, but well-organized career-oriented resources, such as the ability to post your résumé or vitae, the ability to search an active database of hundreds of open positions, and the ability to set up alerts and feeds of positions that fit your personalized criteria. Over the past three years, 1,500+ employers have

posted more than 1,782 jobs on ACM's Job Board, nearly 14,000 ACM members have signed up to use the free service, and over 4,800 ACM members have posted their résumés in the secure database made available to employers. The site is among the most heavily trafficked of all ACM Web sites with over 72,000 unique visitors from over 181 countries using the site over the past year generating nearly 150,000 visits



and 550,000 page views. Our users spend on average over three minutes on the site during each visit and the site attracts a significant amount of repeat business from many of the most respected employers from both academia and industry around the world. If you haven't already visited the site at <http://jobs.acm.org>, check it out! This is one of the most valuable benefits you have as an ACM member, and it just may prove a key resource as you take the next step in your career.

ACM Member News

JUDITH S. OLSON
NAMED 2011–2012
ATHENA LECTURER



In an age of telecommuting, distributed work teams, and social media, getting

coworkers who are geographically dispersed to work well together has never been more important. As a leader in computer science and psychology, Judith S. Olson, the Donald Bren Professor of Information and Computer Sciences at the University of California, Irvine, has greatly improved society's understanding of the factors that make some geographically dispersed teams succeed while others fail miserably. In recognition of her fundamental contributions to computer science, ACM's Council on Women in Computing has named Olson the 2011–2012 Athena Lecturer.

Trained as a cognitive psychologist, she entered the new field of informatics in 1983 to learn how people could better use emerging technologies like personal computers and videoconferencing. Through lab and field experiments, Olson found that, despite increasingly collaborative technology, it is still difficult for a team separated by distance to work as well as a team that shares a common space. Some of the biggest issues: building trust and understanding among teams.

In one instance, she found that instant messaging builds trust better than exchanging curriculum vitae or other work-related documents. "The level of trust was much higher with online chat and conversation than it was by just seeing someone's background," says Olson. "It's humanizing, and you react to each other and build on each other. The interaction turns out to be very important. It's like a digital water cooler."

Olson has created a checklist of 25 items, which will be part of a Web service to be launched later this year, to help telecommuting groups evaluate and improve their long-distance working relationships.

—Graeme Stemp-Morlock

Sorting Through Photos

Teaching computers to understand pictures could lead to search engines capable of identifying and organizing large datasets of visual information.

THE NUMBER OF photos on the Internet is large and rapidly getting larger. The photo-hosting Web site Flickr uploaded its five-billionth picture on September 18, 2010, and Pingdom, a Swedish company that monitors Internet performance, estimated last year that Facebook is adding photos at a rate of 30 billion a year.

Such numbers present both a challenge and an opportunity for scientists focused on computer vision. The challenge lies in figuring out how to design algorithms that can organize and retrieve these photos when, to a computer, a photograph is little more than a collection of pixels with different light and color values. The opportunity comes from the enormous wealth of data, both visual and other types, which researchers can draw on.

"If you think of organizing a photo collection purely by the image content, it's sort of a daunting task, because understanding the content of photos is a difficult task in computer science," says Jon Kleinberg, professor of computer science at Cornell University. People look at two-dimensional pictures and immediately conjure a mental three-dimensional image, easily identifying not only the objects in



MIT's Visual Dictionary project, which is creating a large dataset of labeled images, relies on humans' ability to recognize images even when they are just 32 x 32 pixels.

the picture but their relative sizes, any interactions between the objects, and even broad understandings of the season, time of day, or rough location of the scene. When computers look at a photo, "they're seeing it as a huge collection of points drawn on a plane," notes Kleinberg.

But scientists are finding ways to extract the hidden information, using machine learning algorithms to first identify objects and then to uncover relationships between them, and by relying on hints provided by users' photo tags, associated text, and other relationships between different pictures.

"In the old days we used to do image matching," says Nuno Vasconcelos, head of the Statistical Visual Computing Laboratory at the University of California, San Diego. Computers would derive some statistical model of an example image and then look for matches with other images. "It works to some extent," says Vasconcelos, "but it doesn't work very well." The programs would find low-level matches, based on factors such as color or texture. A beach scene, all sand and sky, might be matched with a picture of a train, with an equal amount of sky and a color similar to sand.

Nowadays, Vasconcelos says, the emphasis is on trying to understand what the image is about. Starting with a set of images labeled by humans, a machine learning algorithm develops a statistical model for an entire class of images. The computer calculates the probability that a picture is a beach scene—based on labels such as "beach," "sand," "ocean," and "vacation"—and then matches the picture with other images with the same probability.

To train such algorithms, scientists need large sets of labeled images. While datasets with a few thousand photos exist, the algorithms become more accurate with much larger sets.

Fei-Fei Li, an assistant professor at the Stanford Vision Lab, starting developing such a dataset in ImageNet, along with Kai Li, a computer scientist at Princeton University.

They started with WordNet, a hierarchical database of English words in which distinct concepts are grouped into sets of synonyms called synsets; there are 80,000 synsets just for nouns. The researchers entered each of the synonyms into Internet search engines to collect about 10,000 candidate images per synset. Then, using labor provided by Amazon Mechanical Turk, in which people earn small payments for tasks that require human input, they had people verify whether a candidate image contained the object listed in the synset. The goal is to have 500 to 1,000 images per synset. So far, they've amassed more than 11 million labeled images in about 15,500 categories, putting them between a third and halfway toward their goal.

About 100 people participated in the ImageNet Challenge last summer to see if they could use the dataset to train computers to recognize objects in 1,000 different categories, from "French fries" to "Japanese pagoda tree." Once the computers have shown they can identify objects, Fei-Fei says

the next objective will be to recognize associations between those objects. Noticing context can aid in object recognition, she explains. "If we see a car on the road, we don't keep thinking 'Is it a boat? Is it an airplane?'"

Using Human Recognition

The Visual Dictionary project at the Massachusetts Institute of Technology (MIT) also seeks to develop a large dataset of labeled images, but relies on the fact that humans can recognize images even when they're only 32×32 pixels. A Web page displays a mosaic representing 7.5 million images associated with 53,464 terms, with closely related words placed near each other on the mosaic. Each tile on the mosaic shows the average color of all the pictures found for that term, and clicking on it displays a box containing a definition and a dozen associated images. As people click on each tiny picture to verify that it matches the word, the computer records those labels. In another MIT project, LabelMe, the labeling gets even more specific, identifying not just a person, but heads, legs, and torsos, as well as roads, cars, doors, and so on.

The small size of these photos helps keep down the demand on computing capacity, but it also reveals something

Milestones

Ben Franklin Medal and Other CS Awards

The Franklin Institute, Anita Borg Institute for Women and Technology, and other organizations recently honored leading computer scientists for their research and leadership qualities.

BEN FRANKLIN MEDAL

The Franklin Institute presented the 2011 Benjamin Franklin Medal in Computer and Cognitive Science to John Anderson, R. K. Mellon University Professor of Psychology and Computer Science at Carnegie Mellon University, for the development of Adaptive Control of Thought. His work reflects the first large-scale computational theory of the process by which humans perceive, learn, and reason, and its application to computer tutoring systems.

WOMEN OF VISION AWARDS

The Anita Borg Institute presented its 2011 Women of Vision awards. Chieko Asakawa, an IBM Fellow at IBM Research-Tokyo, was honored with the Leadership Award; Mary Lou Jepsen, CEO of Pixel Qi, the Innovation Award; Karen Panetta, professor of electrical and computer engineering at Tufts University, the Social Impact Award; and IBM received the Anita Borg Top Company For Technical Women Award.

CRA AWARDS

Computing Research Association (CRA) board of directors selected Charles Lickel, retired executive vice president, IBM, to receive the 2011 A. Nico Habermann Award for his accomplishments at the national, local, and

individual levels for increasing underrepresented groups, and particularly for researchers in the gay, lesbian, bisexual, and transgendered computing community. The CRA Board of Directors also selected Jeannette M. Wing, President's Professor of Computer Science and Head, Computer Science Department, Carnegie Mellon University, to receive the 2011 Distinguished Service Award for her national and international thought leadership with respect to Computational Thinking, and for her extraordinary performance as National Science Foundation Assistant Director for Computer and Information Science and Engineering from 2007–2010.

ACM SIGCHI AWARDS

The ACM Special Interest Group

on Computer Human Interaction presented the Lifetime Practice Award, which recognizes the very best and most influential applications of human-computer interaction, to Larry Tesler, an independent consultant, for his "work at Xerox PARC and Apple [which] has impacted literally every computer user today." The Lifetime Research Award was presented to Terry Winograd, a computer science professor at Stanford University, for "fundamental contributions to the design of interactive computer systems by taking a broad view of HCI, considering it in the context of natural language processing, machine and human intelligence, cognitive science, human-machine communication, design, and software design."

—Jack Rosenberger

Scientists are extracting the hidden information in photos by using machine learning algorithms to identify objects and uncover relationships between them, and by relying on users' photo tags and associated text.

important about vision, says Antonio Torralba, an associate professor at MIT who heads the project. "If humans are able to do recognition at such low resolution, there are two possibilities. Either the human visual system is amazing or the visual world is not that complex," he says. In fact, he adds, the world is complex. "Most of the information is because you know a lot about the world," he says.

The fact that much of the semantic content of a photo is actually supplied by the human viewing it leads researchers to try to derive clues about the content from what humans do with the pictures. Kleinberg makes the analogy with Web search, which not only looks at the textual content of Web pages, but also their structure, such as how they are organized and what hyperlinks they contain. Kleinberg uses the geotagging of photos to learn what they're about, with the tags supplied either by Flickr users clicking on the Web site's map or by GPS-based tags automatically created by a user's camera. It turns out that sorting location tags on a 60-mile scale identifies population centers, and on a 100-meter scale identifies landmarks—the things people like to take pictures of.

For each of those scales, Kleinberg has the computer comb the textual description looking for the words whose use peaks most—not the most commonly used words but the words that are used more in one particular geographic area than any other. For in-

stance, in one area the most outstanding word is Boston. Focusing on that region at the 100-meter scale finds the peak term to be Fenway Park. Pictures so labeled might actually be pictures of a car outside Fenway Park, or your dad at the ball game, or a pigeon on the Green Monster (the left-field wall at Fenway), but when the computer compares all the labeled photos to find the biggest cluster that are mutually similar to each other, a photo of the baseball diamond emerges as the typical image of Fenway Park. "There was no a priori knowledge built into the algorithm by us," Kleinberg says.

Even with the growing datasets, more powerful processors, and constantly improving algorithms, researchers say the ability of computers to identify and organize photographs is still in its early stages. "There's so much stuff that we still need to do, but it's also true that we are doing a lot more than we were doing 10 years ago," Vasconcelos says. "It really looks like 10 years from now we will be able to do a lot better than we can do today." **C**

Further Reading

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L.

ImageNet: A large-scale hierarchical image database, *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, June 20–25, 2009.

Grandall, D., Backstrom, L., Huttenlocher, D., and Kleinberg, J.

Mapping the world's photos, *Proceedings of the 18th International World Wide Web Conference*, Madrid, Spain, April 20–24, 2009.

Hays, J. and Efros, A.A.

IM2GPS: estimating geographic information from a single image, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, AK, June 23–28, 2008.

Torralba, A., Fergus, R., and Freeman, W.T.

80 million tiny images: a large dataset for non-parametric object and scene recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 11, Nov. 2008.

Vasconcelos, N.

From pixels to semantic spaces: advances in content-based image retrieval, *IEEE Computer* 40, 7, July 2007.

Neil Savage is a science and technology writer based in Lowell, MA. David A. Patterson, University of California, Berkeley, contributed to the development of this article.

© 2011 ACM 0001-0782/11/05 \$10.00

In Memoriam

David E. Rumelhart 1942–2011

David E. Rumelhart, a pioneer in computer simulations of perception, died on March 13 in Chelsea, MI, at the age of 68 after suffering from a debilitating neurological condition.

A psychologist, Rumelhart made many contributions to the formal analysis of human cognition, working mainly within the frameworks of mathematical psychology, artificial intelligence, and parallel distributed processing.

While working at the University of California, San Diego, Rumelhart developed a computer simulation of how three or more layers of neurons could work together to process information, which is necessary for the brain to perform complex tasks. This system, which was more sophisticated than previous models, was described in a landmark paper he wrote in 1986 with Geoffrey Hinton and Ronald Williams for *Nature*, and led to new, more powerful systems for visual object recognition and handwritten character classification.

Rumelhart was also well known for his textbook, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, written with Jay McClelland, says Hinton, the Raymond Reiter Distinguished Professor of Artificial Intelligence in the computer science department at the University of Toronto. *Parallel Distributed Processing* described the authors' computer simulations of perception, and provided the first testable models of neural processing. It is regarded as a central text in the field of cognitive science.

The Robert J. Glushko and Pamela Samuelson Foundation honored Rumelhart in 2000 with the creation of the David E. Rumelhart Prize, an annual award given to an individual or team making a significant contemporary contribution to the theoretical foundations of human cognition.

—Bob Violino

I, Domestic Robot

With recent advances in laser rangefinders, faster algorithms, and open source robotic operating systems, researchers are increasing domestic robots' semantic and situational awareness.

INDUSTRIAL ROBOTS, FIXED-LOCATION and single-function machines, have long been staples of advanced manufacturing settings. Medical robots, which can help surgeons operate with smaller incisions and cause less blood loss than traditional surgical methods, are making fast inroads in metropolitan and suburban hospitals. Rescue robots, included wheeled and snake-like robots, are increasingly common, and were deployed in the search for survivors in the aftermath of the earthquake and tsunami that recently struck Japan. On the other hand, the promise of multipurpose domestic assistance robots, capable of a wide range of tasks, has been a distant goal.

However, recent advances in hardware such as laser rangefinders, open source robotic operating systems, and faster algorithms have emboldened researchers. Robots are now capable of folding laundry, discerning where to place an object on cluttered surfaces, and detecting the presence of people in a typical room setting.

"It's easy for me to be optimistic, but if robots aren't actually being useful and fairly widespread in 10 years, then I will be fairly disappointed," says Charles Kemp, assistant professor of biomedical engineering at Georgia Tech University.

Sensors Enable Awareness

In recent months, numerous research teams have published papers detailing advances in robots' perceptual capabilities. These perceptual advances enable the robots' mechanical components to complete domestic tasks hitherto impossible.

Kemp and his research team have pioneered semantic and situational awareness in robots through several methods, including the creation of radio frequency identification (RFID)



Willow Garage's PR2, an open source robotics research and development platform.

semantic tags on common objects such as light switches, and by combining sensor data taken from both two-dimensional camera data and three-dimensional point clouds gathered by laser rangefinders.

University of Bonn researchers Jörg Stückler and Sven Behnke also demonstrated success, using a combination of 2D laser and camera sensors. They programmed a mobile service robot to combine laser rangefinder data that hypothesizes the presence of a person's legs and torso with 2D frontal and profile images of the detected face.

Stückler and Behnke also modeled the semantic probability of detecting a person's presence in different locations of a room—high probability in a chair and low probability on a bookshelf, for instance—and supplied the robot with that knowledge. The prior knowledge of the room semantics and precalculat-

ed range of likely valid facial height helps the Bonn researchers discern false positive returns.

Steve Cousins, CEO of Willow Garage, which manufactures the open platform general-purpose PR2 robot, says further advances in perceptual capabilities may be even more likely with the recent debut of sensing technology that enables a computer to analyze an area in three dimensions and then to create what the technology's manufacturer, PrimeSense, calls a synchronized depth image. The technology sells for less than 1/20th of the de facto standard research rangefinder, which costs about \$5,000. Both Cousins and Kemp believe the low cost of the PrimeSense sensor (it is a key component of Microsoft's Kinect gaming system) may lead to a surge in situational and semantic robotic research. Kemp says his team recently installed one of the new sensors to its PR2.

In essence, Kemp says its real-time technology greatly simplifies a robot's data-gathering process.

Prior to installing the new sensor, on projects such as the work on making the robot discern clutter, he says "we had to tilt the laser rangefinder up and down, then snap a picture and relate those two things. That's a pretty slow process and really expensive."

A Semantic Database

Kemp says there are two distinct research areas for similar problem sets in domestic robotics: those related to perceptual problem sets, and those related to mechanical awareness. For example, a roving robot meant to help a person with basic housekeeping chores must not only know how to differentiate a refrigerator door handle from a light switch, but it must also be able to calculate which approach its arms must take, and how firmly it must grip the respective levers.

In the experiment using RFID tags, Kemp created a semantic database the robot could refer to after identifying an object. The database contains instructions on how the robot should act upon an object. For example, under “actions,” after a robot identifies and contacts a light switch, the commands are “off: push bottom” and “on: push top.” Each of these actions is further sub-programmed with a force threshold the robot should not exceed.

Kemp is also investigating another approach to providing robots with such situational awareness that entails equipping human subjects with touch sensors. The sensors are held during the completion of common tasks such as opening refrigerators and cabinet doors in multiple settings. The information on the kinematics and forces of such actions is then entered into a database a service robot can access when it approaches one of these objects en route to performing a task.

“If the robot knows it is a refrigerator, it doesn’t have to have worked with that specific refrigerator before,” he says. “If the semantic class is ‘refrigerator’ it can know what to expect and be more intelligent about its manipulation. This can make it more robust and introduces this notion of physically grounded common sense about things like how hard you should pull when opening a door.”

Offboard computation akin to the kinematic database is also being done to improve already successful robotic tasks. A team of researchers led by Pieter Abbeel, an assistant professor of computer science at the University of California, Berkeley, programmed a general-purpose Willow Garage PR2 robot to fold towels randomly laid down on a tabletop by using a dense optical flow algorithm and high-resolution stereo perception of the towels’ edges and likely corners. Abbeel’s experiment yielded a perfect 50-out-of-50-attempt success rate; the robot was able to recalculate failures in the 22 instances that were not initially successful by dropping the towel, regrasping a corner, and carrying on until the task was completed.

Abbeel says his team has been able to greatly reduce the amount of time necessary to fold each towel in subsequent experiments, from 25 minutes to approximately four minutes, by uti-

lizing a new approach: rather than rely heavily upon onboard perceptual data, Abbeel has performed parallel computations on the Amazon cloud on mesh models. Those models, he says, are “triangles essentially put together like people using computer graphics or physics-based simulations. Once you have that mesh model, you can do a simulation of how this article of clothing would behave depending on where you pick it up.”

The new approach, he says, relies on observations that the bottommost point of any hanging article is usually a corner. Two consecutive grasps of a towel, he says, will be highly likely to yield two diagonally opposed corners. For t-shirts, he says, likely consecutive grasps will be at the end of two sleeves for a long-sleeved shirt or the end of one sleeve and diagonally across at the hip for a short-sleeved shirt.

“There are a few of these configurations you are very likely to end up in, then all you need to do perception-wise is to differentiate between these very few possibilities,” Abbeel says.

ROS is Boss

Another hallmark advance of the domestic robot community is the growth of an open-source ecosystem, built around the BSD-licensed Robot Operating System (ROS), largely maintained by Willow Garage and Stanford University.

“Our goal has basically been to set the foundation for a new industry to start,” Cousins says. “We want two people to be able to get together in a garage and get a robotics business off the ground really quickly. If you have to build software as well as hardware from scratch, it’s nearly impossible to do that.”

Abbeel says the ROS ecosystem may go a long way to taking the robots out of the lab and into real-world locations.

“In order for these robots to make their way into houses and become commercially viable, there will need to be some sort of bootstrapping,” Abbeel says. “It will be very important for people to do some applications extremely well, and there has to be more than one. So I hope what may be happening, with robots in different places, is that different schools will develop a true sensibility for the robot, and these things could potentially bootstrap the

process and bring the price down. A single app won’t be enough.”

Cousins says the combination of falling hardware prices for devices such as the PrimeSense sensor, and the blooming ROS ecosystem might be analogous to the personal computer research of the early 1970s, specifically comparing the PR2 to the iconic Xerox Alto desktop computer. List price on the PR2 is \$400,000.

“Right now the PR2 is the platform to work on if you want to do mobile manipulation research,” Cousins says. “It’s a little expensive, but in today’s dollars it’s about the same as the Alto. It’s not going to be the robot you put into your grandmother’s home, but the software we develop on the PR2 will likely be a key component of the market. I think ROS is going to be driving those future personal robots.” 

Further Reading

Stückler, J. and Behnke, S. *Improving people awareness of service robots by semantic scene knowledge, Proceedings of RoboCup International Symposium, Singapore, June 25, 2010.*

Maitin-Shepard, J., Cusumano-Towner, M., Lei, J., and Abbeel, P. *Cloth grasp point detection based on multiple-view geometric cues with application to robot towel folding, 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, May 3–8, 2010.*

Schuster, M.J., Okerman, J., Nguyen, H., Rehg, J.M., and Kemp, C.C. *Perceiving clutter and surfaces for object placement in indoor environments, 2010 IEEE-RAS International Conference on Humanoid Robots, Nashville, TN, Dec. 6–8, 2010.*

Yamazaki, A., Yamazaki, K., Burdelski, M., Kuno, Y., and Fukushima, M. *Coordination of verbal and non-verbal actions in human-robot interaction at museums and exhibitions, Journal of Pragmatics 42, 9, Sept. 2010.*

Attamimi, M., Mizutani, A., Nakamura, T., Sugiura, K., Nagai, T., Iwahashi, N., Okada, H., and Omori, T. *Learning novel objects using out-of-vocabulary word segmentation and object extraction for home assistant robots, 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, May 3–8, 2010.*

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

© 2011 ACM 0001-0782/11/05 \$10.00

Data Optimization in Developing Nations

Artificial intelligence and machine learning could expand access to health care, improve the quality of education, and respond effectively to natural disasters in the developing world.

BY NOW, MANY scientists and CEOs have begun to seize the opportunities that lie within the exabytes of data being generated each day. Banks trawl data to detect criminal fraud, marketers to spot emerging trends, researchers to uncover new patterns, and governments to reduce crime and provide better services.

Most data analyses thus far have focused on developed societies. Yet, a growing community of computer scientists is calling for new applications that would harness these data-analysis methods to improve the lives of people in developing nations. Machine learning and artificial intelligence, they say, are perfectly poised to promote socioeconomic development, respond more effectively to natural disasters, expand access to health care, and improve the quality of education. Now, thanks to the efforts of Eric Horvitz, a distinguished scientist at Microsoft Research, and Nathan Eagle, a researcher who lives in Kenya and holds faculty appointments at the Massachusetts Institute of Technology (MIT) Media Lab and Northeastern University, a small but diverse group of computer scientists is banding together to share ideas and information, and to define itself as a community.

Interest about the developing world has been growing in the field of Information and Communication Technology for Development (ICT-D), which encompasses projects that range from managing the delivery of basic services like health care and education to developing network infrastructure, but ICT-D has rarely focused on opportunities to apply artificial intelligence or mine data from developing nations. Last year, ICT-D experts set out to rectify that situation with the formation of the ACM Special Interest Group on Global



Nathan Eagle (above), Eric Horvitz, and others are creating an Artificial Intelligence for Development community to address problems in economically developing countries.

Development (SIGDEV), which held its first conference at the University of London in December. What Horvitz, Eagle, and others aim to do is foster the creation of a subfield within ICT-D to address these deficiencies. The name they've proposed for it: Artificial Intelligence for Development, or AI-D.

It began two years ago at a Princeton University conference called Studying Society in a Digital World, which was organized by Edward W. Felton, director of the university's Center for Information Technology Policy. Eagle presented a paper about using large data sets—in this case, phone calls in Britain—to test American sociologist Mark Granovetter's "The Strength of Weak Ties" theory, which argues that innovation often travels most effectively via weak social connections. Did factors like the geographical distance between call-

ers correlate with socioeconomic indicators like income and education? As it turns out, they did: Regions with a higher volume of geographically diverse calls scored lower on the Index of Multiple Deprivation, a statistical study that covers factors like employment, crime, and health care. Horvitz was intrigued. "I'm passionate about machine intelligence and its applications," he explains. "And I realized there's a lot we can do to stimulate thought." Horvitz was president for the Association for the Advancement of Artificial Intelligence (AAAI); with Eagle's help, he set up an AAAI symposium titled Artificial Intelligence for Development at Stanford University, which took place last March.

"Our idea was that we have so much data, and the majority of it is being generated by people in the developed world," says Eagle. "There's

a real opportunity for us to repurpose that data and serve these underserved communities.”

The diverse set of projects presented at the Artificial Intelligence for Development symposium underscored his point. Much of the research was preliminary, but the initial results were promising. Shawndra Hill, an assistant professor in Operations and Information Management at the Wharton School of the University of Pennsylvania, who has also taught at Addis Ababa University (AAU), spoke of efforts to improve Ethiopia’s road safety. Ethiopia has the world’s highest rate of traffic fatalities, according to the World Health Organization, with a reported 114 deaths per 10,000 vehicles per year. By comparison, the U.K. has one death per 10,000 vehicles per year.

“The Ethiopian Traffic Enforcement Agency collects data on every accident that’s reported,” Hill explains. “Where did the accident happen, what did the intersection look like, what’s the road quality, was it raining, and so on.” Working with AAU lecturer Tibebe Beshah, Hill investigated the role of road-related factors in accident severity. The researchers tested classification models to predict the severity of more than 18,000 car accidents and used a projective adaptive resonance theory algorithm to identify the data’s significant patterns. One research finding: Severe physical injuries were more likely to occur on straight, flat roads than on all other types of roads in the same area.

“The methods don’t change,” says Hill. “You could do the same analysis with data from the United States.” In a country that has the highest rate of traffic fatalities in the world, however—and those accidents being among the nation’s leading causes of death—the potential socioeconomic impact is huge. In the future, Hill and her fellow researchers hope to develop new predictive models that combine road data with driver information, and develop a decision support tool for the Ethiopian Traffic Office.

At the Artificial Intelligence for Development symposium, Eagle and Horvitz presented research in which they deduced the impact of seismic activity in the Lac Kivu region of the Demo-

cratic Republic of the Congo from three years of mobile phone data in neighboring Rwanda. “By watching anomalous call behavior, we could infer the epicenter of the earthquake,” Horvitz explains. The researchers could then make inferences about which areas in the Lac Kivu region were likely to have suffered the greatest damage and be of higher priority for emergency assistance workers. Eagle has used the same data to better understand the dynamics of urban slums and model the effects of social networks on infectious disease outbreaks. And University of California, Berkeley postdoctoral research fellow Emma Brunskill spoke of using traveling salesman techniques to help community health workers in the developing world—some of whom can be responsible for up to 4,000 people—improve the efficiency and timing of their visits to patients in rural areas. The data analysis was exploratory, but Brunskill says she is encouraged by the potential of existing techniques. Another area she finds promising is education. Schools in developing nations often rely on a single computer per classroom. In experimental trials in Bangalore, India, Brunskill and a team of researchers built on foundational studies in multi-input interfaces to test the efficacy of an adaptive multi-user learning game. Initial trials suggested

“Our idea was that we have so much data, and the majority of it is being generated by people in the developed world,” says Nathan Eagle. “There’s a real opportunity for us to repurpose that data and serve these underserved communities.”

that customizing each student’s experience could increase her or his engagement by reducing the likelihood that a single student dominated the game.

Constraints, Costs, Challenges

While AI-D research methods may be the same as they are in mainstream Western science, other factors in developing nations are quite different. First and foremost are the technology constraints. Access to electricity, computers, and the Internet is limited in many areas. Language presents another barrier, as does cost. “The design considerations are much different,” says Lakshmi Subramanian, an assistant professor at the Courant Institute of Mathematical Sciences at New York University. Subramanian’s research includes the use of document classification and focused crawling methods to build offline educational portals, and computer vision techniques to detect diabetic retinopathy, the world’s leading cause of adult blindness. Yet, according to Subramanian, constraints are what make the problems interesting. “If you can only use SMS, what can you do? Turns out, you can do a lot, thanks to semantic compression and other tools,” he says. “In fact, we’ve built an SMS search engine in Kenya.”

Gaining access to useful data can also be a challenge. “There’s no culture of data like there is in the West,” says Hill. “Even businesses in Ethiopia aren’t collecting data like we are.” As a result, one of Horvitz and Eagle’s priorities is to create a central data repository to support new research projects. They began by compiling a list of useful resources at the AI-D symposium Web site, <http://www.ai-d.org>, from organizations like the World Bank, World Trade Organization, and UNICEF. They are also working with regional organizations, such as telephone companies, to share additional data.

“We’re trying to set up a Switzerland for data sets,” says Horvitz.

Beyond that, Horvitz and Eagle hope to get more computer scientists involved. Not surprisingly, in such a young field, there are differences of opinion about research, strategies, and direction. “There is a tension inherent in this area, as in the broader computing for development com-

munity, about whether you're trying to solve a problem or define a new research area," admits Brunskill. Many successful projects are trying to solve immediate societal problems, but researchers in the community are nonetheless convinced there are possibilities for serious long-term science.

"For example, I believe computer scientists can lead the way in combining data- and model-centric methods to design proactive plans that can mitigate the spread of diseases or the shortages of food and water following a disaster," Horvitz explains. "The idea is to use computational models and data from similar situations to make inferences about actions that promise to have a high expected value. Partial plans might be generated proactively with the goals of maximizing the survival of people who have been injured or are trapped. The details of such 'contingency plans' for transporting medications, food, and water could be instantiated in real time based on sensor data."

In the meantime, the field's lack of definition suits many scientists just fine. "Fundamental research in this domain is about understanding the ground realities," says Subramanian, who points out that in "the 1970s, you

"In computer science, you don't often have the sense that you're in a noble profession," says Eric Horvitz. "This is an interesting outlet for passionate people."

didn't have specialties like architecture or networking. You were trying to get things done."

"In computer science, you don't often have the sense that you're in a noble profession," says Horvitz. "This is an interesting outlet for passionate people." 

Further Reading

Beshah, T. and Hill, S.

Mining road traffic accident data to

improve safety: role of road-related factors on accident severity in Ethiopia, *Proceedings of AAAI Artificial Intelligence for Development (AI-D'10)*, Stanford, CA, March 22–24, 2010.

Brunskill, E. and Lesh, N.

Routing for rural health: optimizing community health worker visit schedules, *Proceedings of AAAI Artificial Intelligence for Development (AI-D'10)*, Stanford, CA, March 22–24, 2010.

Eagle, N. and Horvitz, E.

Artificial intelligence for development (AI-D), *Proceedings of the CCC Workshop on Computer Science and Global Development*, Berkeley, CA, Aug. 1–2, 2009.

Kapoor, A., Eagle, N. and Horvitz, E.

People, quakes, and communications: inferences from call dynamics about a seismic event and its influences on a population, *Proceedings of AAAI Artificial Intelligence for Development (AI-D'10)*, Stanford, CA, March 22–24, 2010.

Silberman, N., Ahrlich, K.,

Fergus, R., and Subramanian, L.

Case for Automated Detection of Diabetic Retinopathy, *Proceedings of AAAI Artificial Intelligence for Development (AI-D'10)*, Stanford, CA, March 22–24, 2010.

Leah Hoffmann is a technology writer based in Brooklyn, NY.

© 2011 ACM 0001-0782/11/05 \$10.00

Cybersecurity

Microsoft Dismantles Rustock Botnet

Microsoft has shut down what a spokesperson for the company's Digital Crimes Unit described as a notorious botnet that was sending some 30 billion junk email messages each day. Botnets typically consist of thousands of compromised computers remotely controlled for the purpose of executing denial-of-service attacks, spreading malware, or participating in other nefarious activities. In the case of Rustock, which reportedly consisted of hundreds of thousands of infected computers controlled by several command servers, the main activity was to send junk email.

Rustock's anonymous operators used the botnet to hawk low-cost pharmaceuticals, urge recipients to participate in fake lotteries, and otherwise clutter inboxes with unsolicited missives. Following the

shutdown of the botnet in March, email-monitoring agencies reported a significant downturn in junk email. As one example of the message volume Rustock was capable of achieving, a single Rustock computer that Microsoft researchers monitored leading up to the botnet shutdown was reportedly sending email at the rate of 10,000 pieces per hour.

The shutdown strategy involved a two-pronged approach: A legal initiative to seize the botnet's control servers and a technical initiative to block the infected client computers from establishing connections to new command servers. After an investigation by Microsoft's Digital Crimes Unit and successful legal pleadings, Microsoft's Operation b107 resulted in the simultaneous seizure by the U.S. Marshals Service of servers at five hosting

providers in the U.S.

Microsoft worked on the Rustock initiative with several organizations, including the Dutch High Tech Crime Unit, to dismantle command servers outside the U.S. and China's Computer Emergency Response Team to block the registration of domains that Rustock-infected computers could have contacted for future directives. Microsoft's Digital Crimes Unit is now working with Internet service providers to identify and cleanse the computers infected with the Rustock malware.

"It is impressive that Microsoft managed to get this operation working so smoothly," says Johannes Ullrich, chief research officer at the SANS Institute. According to Ullrich, the initiative's legal aspect shows maturity about the process of shutting down a botnet. "It's no

longer a bunch of guys deciding they don't like a botnet and taking technical measures to shut it down," Ullrich says.

Still, it remains to be seen whether this two-pronged approach, consisting of legal and technical components, will be more viable in the future than purely technical countermeasures. In assessing the long-term impact of such efforts, Ullrich likens botnets to home burglaries, which technical means, such as locks, and legal repercussions, such as jail time, make manageable but don't eliminate altogether. Ullrich says a similar approach may help mitigate the botnet threat, making it manageable.

"Look at it like home burglaries," he says. "Don't ask for absolute security but for reasonable security."

—Kirk L. Kroeker

ADVANCE YOUR CAREER WITH ACM TECH PACKS...

For Serious Computing Professionals.



Searching through technology books, magazines, and websites to find the authoritative information you need takes time.

That's why ACM created "Tech Packs."

- Compiled by **subject matter experts** to help serious computing practitioners, managers, academics, and students understand today's vital computing topics.
- Comprehensive **annotated bibliographies**: from ACM Digital Library articles, conference proceedings, and videos to ACM Learning Center Online Books and Courses to non-ACM books, tutorials, websites, and blogs.
- **Peer-reviewed** for relevance and usability by computing professionals, and **updated** periodically to ensure currency.

Access to ACM resources in Tech Packs is available to ACM members at <http://techpack.acm.org> or <http://learning.acm.org>.

Current topics include **Cloud Computing** and **Parallel Computing**. In development are Gaming, Globalization/Localization, Mobility, Security, and Software as a Service (SaaS).

Suggestions for future Tech Packs? Contact:

Yan Timanovsky
ACM Education Manager
timanovsky@hq.acm.org



Association for
Computing Machinery

Advancing Computing as a Science & Profession

Deus Ex Machina

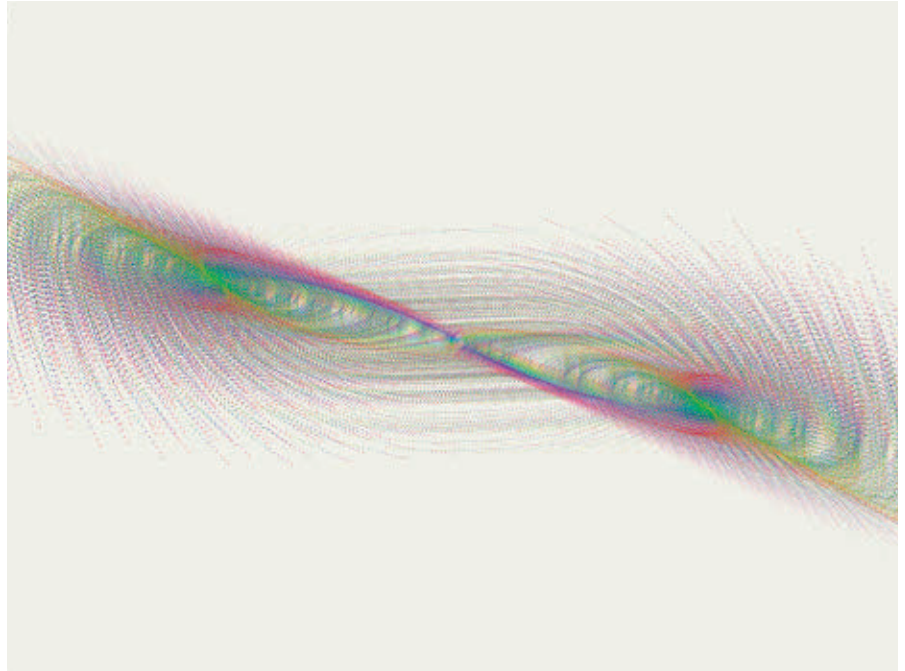
Computational metaphysics is helping philosophers answer age-old questions, such as whether God exists.

A FAMOUSLY TRICKY argument for the existence of God proposed by the British theologian Anselm in the 11th century recently got simpler with help from an automated reasoning engine. In a forthcoming paper in the *Australasian Journal of Philosophy*, Stanford philosophers Paul Oppenheimer and Edward Zalta discuss how they used a program called Prover9 to not only validate Anselm's ontological argument from its admittedly dubious premises, but also greatly reduced the number of premises necessary to reach that conclusion.

This result is one of the more interesting discoveries in the new field of computational metaphysics, which uses computers to reason through problems in metaphysics. "Lots of fields are using computers to explore outstanding questions, and that's true in philosophy no less than in other fields," says Zalta, a senior research scholar at Stanford University's Center for the Study of Language.

Philosophers have used computers before Oppenheimer and Zalta did, but its application is remarkable in metaphysics, a branch of philosophy dealing with the ultimate nature of reality. Lofty questions about existence, causation, and identity might seem too abstruse for automated reasoning; however, when formulated with mathematical precision, metaphysical propositions become ideal candidates for computer-assisted proofs in much the same way that mathematical theorems are, says Rutgers University philosopher Branden Fitelson, who's used automated reasoning in his specialties, logic and the philosophy of science.

When software is doing the philosopher's work of axiomatic reasoning—stepping logically from the premises to the desired conclusion—much of what's left for the philosopher is the task of translating the airy language of philosophy into a form the software



can work from (and then interpreting the program's output). But accomplishing that is a nontrivial process, says Fitelson. Since statements in metaphysics use second-order logic, for which there is no guarantee of a proof for valid claims, "you're outside the realm of being able to do things mechanically at all," Fitelson explains. To get around this problem of undecidability, metaphysicians who want the aid of computers must first translate higher-order claims into the first-order claims of classical logic. But that usually leads to complicated sets of formulas that are hard for humans to work with. What's more, philosophers must represent those formulas in the syntax of their automated reasoning system.

From there, by using tree search algorithms, the software can reliably find a proof or show a counterexample. And there's no beating the rigor that computers provide to philosophers, finding logical holes that might not otherwise be apparent. Because a computer stops once it hits a gap in the logic, for it to validate a proof the argument has to be air-

tight. "In philosophy you're not always sure that's true," says Fitelson, noting that metaphysics can be difficult to reason about with the kind of intuition one might apply to, say, geometry.

Zalta had no doubt when Anselm's premises were fed into Prover9 that it would find a valid proof. "However, when we looked at the actual proof the machine spit out, we saw that it didn't use all three premises!" Prover9 had found a way to derive Anselm's conclusion using just one premise.

Whether Anselm's argument is sound, as opposed to merely valid, depends on whether that premise itself is true—a question that philosophers will continue to debate. Nonetheless, having one premise gives would-be refuters a much clearer target. And, says Zalta, as philosophers develop more results using automated reasoning, the tools' use should become more widespread. ■

Based in San Francisco, **Marina Krakovsky** is the co-author of *Secrets of the Moneylab: How Behavioral Economics Can Improve Your Business*.

© 2011 ACM 0001-0782/11/05 \$10.00

ILLUSTRATION BY GWEN VANHEE

Web Science Meets Network Science

A pair of divergent scientific communities discusses their similarities and differences, and search for common ground.

EVER SINCE WORLD Wide Web inventor Sir Tim Berners-Lee announced the Web Science Research Initiative in 2006, researchers have been trying to map the boundaries of Web science, which spans a dizzying range of disciplines including computer science, economics, government, law, and psychology.

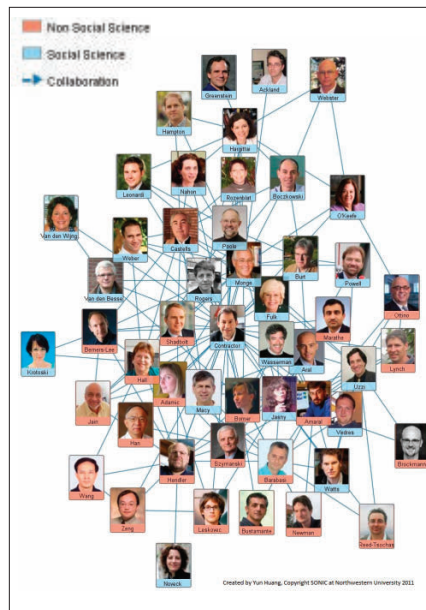
Complicating matters further has been the parallel evolution of a markedly similar-sounding field: Network science, whose devotees explore the characteristics of all types of networks, from neural networks to social networks to, yes, the Web.

Where do these two emerging fields overlap? Where do they diverge? These are some of the questions a group of scholars broached in the Third International Workshop on Network Theory, hosted last March at Northwestern University.

"In one sense, Web science is a subset of network science. In another sense, network science is a subset of Web science," says workshop co-chair Noshir Contractor, a professor of behavioral sciences at Northwestern.

Proponents of the former view argue that the Web is just one network among many that share certain common properties; for example, they are open, scale-free, and exhibit emergent properties like power laws. Proponents of the latter view tend to argue that the Web is fundamentally different from other networks in that it encompasses a broad range of human concerns that have little to do with a macro understanding of networks, such as issues of government policy, commerce, and human factors.

"In practice, Web science is focused on how we could do things better, while network science is more focused on how things work," says



Collaboration network map of the participants of the Northwestern University workshop.

Contractor, "but aspirationally, they are not different."

Given their overlapping areas of interest, it might seem surprising that many of the leading researchers in each field remained largely unaware of the others' work before they met for the first time at the Northwestern workshop.

"This was a coming together of two different communities," says Dame Wendy Hall, a professor of computer

science at the University of Southampton, who is one of the cofounders of the Web Science Research Initiative and now the managing director of its successor, the Web Science Trust.

Like so many good ideas, the idea for the workshop originated over drinks at a hotel bar. Hall remembers having a lively conversation with network theorist Manuel Castells during a meeting of the European Research Council. "We realized that we were coming at the same thing from different angles," Hall says. Soon afterward, Castells introduced Hall to Contractor, initiating a series of conversations that led to the Northwestern workshop.

The workshop organizers hoped to frame a new research agenda by leveraging the commonalities and distinctive contributions of Web science and network science, and to formulate questions of interest to both communities.

The two-day conference covered a wide range of broadly related topics such as debating the merits of network science's "pure" scientific approach vs. the more applied, engineering-oriented tactics of Web science; analyzing the effects of scale on network behaviors; exploring questions of causality, correlation, and inference; and discussing the possibility of a Web index, an idea currently being promoted by Berners-Lee.

Looking ahead, plenty of room exists for continuing dialogue between the two camps, who will almost certainly continue to probe each other's boundaries while searching for common ground.

"Is Web science a subset of network science or is it the same thing?" asks Hall. "The answer is, It doesn't matter." 

Alex Wright is a writer and information architect based in Brooklyn, NY.

© 2011 ACM 0001-0782/11/05 \$10.00

Where do these two emerging fields overlap? Where do they diverge?

ACM TechNews Goes Mobile

iPhone & iPad Apps Now Available in the iTunes Store

ACM TechNews—ACM's popular thrice-weekly news briefing service—is now available as an easy to use mobile apps downloadable from the Apple iTunes Store.

These new apps allow nearly 100,000 ACM members to keep current with news, trends, and timely information impacting the global IT and Computing communities each day.



TechNews mobile app users will enjoy:

- **Latest News:** Concise summaries of the most relevant news impacting the computing world
- **Original Sources:** Links to the full-length articles published in over 3,000 news sources
- **Archive access:** Access to the complete archive of TechNews issues dating back to the first issue published in December 1999
- **Article Sharing:** The ability to share news with friends and colleagues via email, text messaging, and popular social networking sites
- **Touch Screen Navigation:** Find news articles quickly and easily with a streamlined, fingertip scroll bar
- **Search:** Simple search the entire TechNews archive by keyword, author, or title
- **Save:** One-click saving of latest news or archived summaries in a personal binder for easy access
- **Automatic Updates:** By entering and saving your ACM Web Account login information, the apps will automatically update with the latest issues of TechNews published every Monday, Wednesday, and Friday

The Apps are freely available to download from the Apple iTunes Store, but users must be registered individual members of ACM with valid Web Accounts to receive regularly updated content.

<http://www.apple.com/iphone/apps-for-iphone/>

<http://www.apple.com/ipad/apps-for-ipad/>

ACM TechNews



Economic and Business Dimensions

Online Advertising, Behavioral Targeting, and Privacy

Studying how privacy regulation might impact economic activity on the advertising-supported Internet.

DATA ON THE online behavior of consumers has allowed companies to deliver online advertising in an extraordinarily precise fashion. For example, a Lexus dealership can target advertising so that its ads are shown only to people who have been recently browsing high-end cars on auto Web sites. Such behavioral targeting has obvious benefits to advertisers because fewer ad impressions are wasted. Instead, advertisers focus their resources on the consumers most likely to be influenced by the ads. For consumers, however, ads that are behaviorally targeted can appear unauthorized and even creepy. As a result there have been calls in the U.S. and elsewhere for new regulation to restrict the collection and use of online data for advertising purposes.

Unfortunately, there has been little empirical evidence about the consequences of such regulation for the future of advertisers and Web publishers



on the advertising-supported Internet. In a recent research paper² we begin to fill this gap by examining how earlier privacy regulation in Europe affected the performance of online display advertising in Europe relative to the U.S. and elsewhere. In this column we summarize the key findings and discuss how our results inform recent proposals for privacy regulation in the U.S.

Background

Advertisers spend \$8 billion each year on online display advertising and even more on search engine advertising. At the heart of this industry is the detailed collection, parsing, and analysis of consumer data, often without consumers' consent or knowledge. This data allows firms to target their advertising to specific groups who might be most influenced by advertising, and to measure how well the advertising then performs as they track the subsequent behavior of users who were exposed to an ad. There are growing concerns that such unregulated data collection may harm consumers. Specifically, the recent Federal Trade Commission (FTC) preliminary staff report, "Protecting Consumer Privacy in an Era of Rapid Change," (see <http://www.ftc.gov/os/2010/12/101201privacyreport.pdf>) identifies three groups of consumers who might be harmed in the current environment:

- Consumers troubled by the collection and sharing of their information.
- Consumers who have no idea that any of this information collection and sharing is taking place.
- Consumers—some teens for example—who may be aware of the sharing that takes place, but may not appreciate the risks it poses.

The FTC document then argues that there might be benefits to increased regulation. At the same time, the document makes it clear that another objective of any policy is "preserving the ability of companies to innovate, compete, and offer consumer benefits." Given this objective, it is important to understand how privacy regulation might impact economic activity on the advertising-supported Internet.

Our Study

We examined the effect of the EU Privacy and Electronic Communications

Online advertising became much less effective in Europe relative to elsewhere after the regulation was enacted.

Directive (2002/58/EC—sometimes known as the E-Privacy Directive) on online advertising in Europe. Specifically, we looked at how consumer responses to advertising changed in Europe after it came into effect, relative to changes in consumer responses to advertising in the U.S. and elsewhere.

Several provisions of the Privacy Directive limited the ability of companies to track user behavior on the Internet. These changes made it more difficult for the Lexus dealership in our earlier example to collect and use data about consumers' browsing behavior on other Web sites.

The interpretation of this E-Privacy Directive has been somewhat controversial as it relates to behavioral targeting. For example, it is not clear the extent to which a provision that requires companies who use invisible tracking devices such as Web bugs to use them only with the "knowledge" of consumers means that companies need explicitly to obtain opt-in consent. This is one of the reasons why, in the recent "Telecoms Reform Package," the EU amended the current regulation to require a user's explicit consent before placing a cookie on a computer. Therefore our estimates also capture business responses when there is some ambiguity over how privacy regulation should be interpreted.

To measure online advertising effectiveness, we use unusual data from a marketing research company that ran various "a/b" tests of online display ads across the world over eight years. The research company developed a straightforward methodology that permitted comparison of different advertising campaigns over time in order to allow advertisers to benchmark the

effectiveness of different ads. In this "a/b" test, some randomly selected people are exposed to the ad for a certain product, while others were exposed to a placebo ad, usually for a charity. The market research firm then surveyed these Web users about their likelihood of purchasing the advertised product. This allows a clean measurement of the effect of the ad: Because these people are randomly selected, any increase in expressed purchase intent toward the product for the group exposed to the ad relative to those who were not exposed can be attributed to advertising. We use data on 3.3 million of these survey responses for 9,596 different online display advertising campaigns conducted on hundreds of different Web sites across many countries.

Our research links changes in privacy protection in Europe to changes in the advertising-induced lift in purchase intentions. Specifically, we use regression analysis to compare and contrast:

- People who were randomly exposed to the ad and those who were not;
- The EU and elsewhere; and
- Before and after privacy regulation was enacted in Europe.

Our Results

We found that in Europe, after privacy protection was enacted, the difference in stated purchase intent between those who were exposed to ads and those who were not dropped by approximately 65%. There was no such change for countries outside Europe. In other words, online advertising became much less effective in Europe relative to elsewhere after the regulation was enacted.

One possible explanation for this result is that our estimates reflect a change in attitudes among Europeans toward targeted advertising, rather than something that can be causally attributed to the change in law. To examine this possibility, we looked at the behavior of Europeans on non-European Web sites and of non-Europeans on European Web sites. We found no drop in ad effectiveness for Europeans browsing non-European Web sites and a substantial drop in advertising effectiveness for non-Europeans browsing European Web sites. The drop does not appear to be a result of changing consumer attitudes in Eu-

rope. Instead, it suggests that, coincident with the timing of the enactment of European privacy regulation, advertising at Web sites in Europe became less effective.

This drop in ad effectiveness was not uniform across all types of Web sites and advertisements. In fact, many Web sites and ad formats were barely affected. We did not see any significant decline in ad effectiveness on Web sites that had specialized content, such as baby and automotive Web sites. It was Web sites that did not have content that was easily matched to a specific product (Web sites like CNN.com and Dictionary.com) that drove the measured drop in advertising effectiveness. We believe this result is driven by the difficulty of matching a customer to an ad on such Web sites because the Web site content does not reveal specific preferences. In contrast, it is easy to target advertising to a visitor at a Web site for parents of babies, even without data on browsing behavior.

Large, intrusive ads also showed little decline in effectiveness. Instead, the fall in ad effectiveness in Europe was largely driven by plain banner ads. This makes sense because more discreet ads rely on the inherent interest of a customer in that kind of product, rather than on a striking ad design.¹

Limitations

Before concluding, we clarify some of the limitations of this analysis:

► Our measures of advertising effectiveness are representative only for people who are willing to answer an online market survey. Though respondent demographics seem representative of the general population, they

Our results provide some of the first empirical evidence about the likely consequences of privacy regulation.

may be unusual in dimensions we do not observe. Therefore, a conservative interpretation of our results is that we measure a decline in a measurement of advertising effectiveness commonly used by the online advertising industry.

► The campaigns we study are representative of those launched by large firms who had substantial resources. We do not know how privacy regulation affected smaller firms. On the one hand, the costs of compliance may have been higher for such firms. On the other hand, small firms may have been less careful about compliance.

► The campaigns we study were placed directly by the advertiser on particular Web sites rather than being distributed through an advertising network—a common distribution channel for online advertising. It is possible that advertising networks were better able to invest resources to mitigate the effects of privacy regulation, and therefore experienced less of a decline.

► We do not know whether the change in advertising effectiveness affected advertising revenues. This would depend on substitution patterns between online and offline media.

► It is not clear whether our results generalize to Web sites that explicitly offer users control over their privacy settings, such as Facebook. Such proprietary opt-in Web sites may even benefit from regulation of this kind, if it means they are more efficient at delivering ads than their competitors.

What It Means

Notwithstanding these limitations, our results provide some of the first empirical evidence about the likely consequences of privacy regulation. This is important because of recent developments in the U.S. concerning regulation. In their preliminary staff report, the FTC made the following proposal: *The most practical method of providing such universal choice would likely involve the placement of a persistent setting, similar to a cookie, on the consumer's browser signaling the consumer's choices about being tracked and receiving targeted ads. Commission staff supports this approach, sometimes referred to as "Do Not Track."*

Obviously this persistent "opt-out" is a different approach from the EU regulation that we study. However,

Any decline in advertising effectiveness that results from the new regulation will not be borne equally by all Web sites.

our estimates suggest that one could reasonably expect a large drop in advertising effectiveness for consumers who choose to opt out of targeting. Therefore, the likely effects of the proposed regulation depend on the number of consumers who would ultimately choose to persistently opt out, which in turn will be driven by the specifics of the regulatory framework.

Crucially, our empirical findings suggest any decline in advertising effectiveness that results from the new regulation will not be borne equally by all Web sites. In the long run, this may change the kind of Web sites and firms that prosper on the advertising-supported Internet, perhaps leading to fewer free (ad-supported) general-interest Web sites. Our results also suggest that advertisers may move toward more visually arresting types of advertising in order to compensate for their inability to target. Therefore, the potential benefits to consumers of increased privacy should be weighed against the consumer benefits of a potentially broader advertising-supported Internet that has fewer visually distracting ads. **C**

References

1. Goldfarb, A. and Tucker, C. Online display advertising: Targeting and obtrusiveness. *Marketing Science*. Forthcoming.
2. Goldfarb, A. and Tucker, C. Privacy regulation and online advertising. *Management Science* 57, 1 (Jan. 2011), 57–71.

Avi Goldfarb (agoldfarb@rotman.utoronto.ca) is an associate professor of Marketing in the Rotman School of Management at the University of Toronto.

Catherine Tucker (cetucker@mit.edu) is Douglas Drane Career Development Professor of Information Technology and Management and an assistant professor of Marketing at the MIT Sloan School of Management, Cambridge, MA.

Copyright held by author.

Education

Reaching Learners Beyond our Hallowed Halls

Rethinking the design of computer science courses and broadening the definition of computing education both on and off campus.

THE VAST MAJORITY of our efforts in computing education revolve around formal learning environments. By virtue of where computing courses and programs are offered, much of our work is centered on the undergraduate curriculum in colleges and universities. Recently, increased attention has been given to computing education throughout elementary and secondary education in efforts to broaden participation in the field and address concerns about the student pipeline. However, by focusing exclusively on these formal settings, we may be missing an opportunity to reach the millions of people who find computing outside of academia's hallowed halls and are left to teach themselves through informal educational activities.

In this column, I focus on adults actively working in a traditionally non-computing discipline who, nonetheless, develop scripts or programs as part of their activities. These end-user programmers provide a canonical example of people engaged in informal computing education. Examples include accountants who write spreadsheet formulas and macros, biologists who create programs to simulate models or facilitate data analysis, and graphic designers who produce scripts to automate repetitive and tedious tasks. Disciplinary differences aside, a common characteristic of people in these examples is that their formal academic training is in something

These end-user programmers provide a canonical example of people engaged in informal computing education.

other than computing. Their knowledge of the computing fundamentals, and more specifically programming, is built largely from self-taught experiences and advice from peers.

Why should we be interested in these domains? At the very least, these informally trained individuals account for a substantial portion of the "computing pie." Estimates for the U.S. suggest there will soon be more than 13 million self-described programmers in the workplace, compared to fewer than three million professionally trained programmers.⁵ The difficulties of learning to program are well documented, and informal learners are left to grapple with these challenges in an environment very different from what best practices in education recommend. We have the same concerns for these programmers as we have for those who are professional software developers. For example, do they create correct, robust, and reusable

programs? Put simply, software written by informally trained programmers has a user base, and coding mistakes can cost valuable resources. The educational and training needs of this significant group of learners appear largely underserved by academic institutions today.

An Example in Graphic and Web Design

Over the course of the last five years, my colleagues and I have conducted a series of studies to better understand how to support the educational needs of a group of informally trained programmers. We have focused our attention on professional graphic and Web designers who actively write code in some aspect of their job.³ With educational backgrounds rooted in art and visual communication (and notably, very little STEM), the participants in our studies represent a unique outer bound of those who have a need for computing/programming later in life. It seems quite natural to require programming instruction for engineering majors, but rarely are similar requirements considered for art students.

One of the most striking observations about our study participants was the variety of their educational backgrounds. Even though those we interviewed shared similar job descriptions, they were trained in a wide variety of academic disciplines ranging from graphic design (as one might expect) to the humanities and the social sciences.



es. Further, very few of these designers had ever taken a course on programming or scripting as part of their secondary, post-secondary, or even professional development activities. An intriguing (but admittedly anecdotal) observation was that our participants appeared considerably more gender balanced and ethnically diverse than most demographic data reported for computer science.

Obviously this diversity poses a number of challenges for efforts to address the educational needs of these designers in the future. That aside, we took away two important lessons about current informal learning practices:

► **Example-Driven Learning.** The main driver for our participants to learn something new about programming derives directly from the needs of the designer's current project. Within that context, the designer actively seeks out examples related to the end goal in a somewhat piecemeal fashion. Ideally, the designer learns as he or she sees examples that make use of new programming features. Unfortunately, this doesn't always happen due to the unavailability of relevant examples, differences between the current goal and that of the example, and the lack of explanation accompanying examples. The explanations found rarely draw out the computer science ideas used, instead favoring specific practical de-

tails. Despite the inherent problems and inefficiencies in learning this way, the designers we interviewed consistently preferred learning from code examples to more general resources like books. Of utmost importance, then, is the instructive quality of the examples being used (that is, with good explanations) and how obviously relevant the examples are.

► **The Web as the Primary Resource.**

Closely related to the role of examples is a heavy reliance on the Internet as the first line of support. Our participants perceived the Internet as a complete resource—anything they might want must be in there somewhere. Given the low cost associated with searching, the Web is an attractive first option when

looking for information. However, search results related to programming questions can be difficult to interpret. When asked to read and modify a piece of program code, participants often spent significant amounts of time studying search results that pointed them to irrelevant code written in other programming languages. Sometimes a search might lead a user to a conceptually correct algorithm for their problem but implemented in an unfamiliar language. This dissimilarity may cause the example to be discarded altogether, or cause the learner difficulty in making the necessary adaptations to the current context. The primary issues in using the Web as a resource for informal learning about programming can be tied to difficulties in devising appropriate search terms and judging the relevance of the resulting examples.⁴

One underlying cause for these difficulties could be tied to our participants' lack of sufficient general, abstract knowledge of the computing and/or programming structures at play. When a search based on purely syntactic constructs fails, an expert would fall back to a more general conceptual term (such as "exception handling"). The ability to see similarities between algorithms implemented in different languages often requires an ability to abstract away from the concrete syntax. A recognized problem of highly

The educational and training needs of this significant group of learners appear largely underserved by academic institutions today.

context-oriented learning, like what informal learners do, is that the resulting knowledge can become context-bound and make it difficult to transfer to or from other similar settings.² Thus, there is a great opportunity for computing educators to explore how to foster conceptual knowledge growth in such communities while at the same time recognizing existing practices favoring example-driven learning.

In our work, we sought to balance these two issues by creating collections of annotated scripting examples. Each example project presents a solution to a realistic programming task for graphic designers (that is, using the same tools and language). For example, one project creates a script that manipulates image layers in a Photoshop document. The project outlines multiple versions of the script, moving from an iterative solution to a recursive one as a natural response to unexpected results discovered when the script is used on various images.

A key component of our projects distinguishing them from other materials a user might find on the Web is that each project includes a narrative description of the code's development, explaining relevant programming concepts (for example, recursion) as they are used in context. This narrative interleaves explanations with progressively more complete example code, and failure drives the narrative forward.

Essentially, each project serves as a case study that learners can review to access example code as well as instruction. Our evaluation of this resource has shown that these explicit connections to conceptual content can in fact lead to measurable learning gains for end-user programmers.³ We were able to promote learning by briefly articulating details about concepts as they were used in the code while not negatively impacting the overall usability of the examples for those that did not need the explanations.

Roles for Educators

Our work over the past several years has further convinced us of the importance of educators and others involved in formal educational or training institutions to actively consider the role informal learning is playing in the computing landscape. We see

We see several ways to increase involvement in efforts to support the vast population of adults engaging in informal computing education.

several ways to increase involvement in efforts to support the vast population of adults engaging in informal computing education.

First, educators can provide access to the curricular resources already being generated for our classes. Informal learners are often concerned with the reliability and credibility of examples, so our respected, well-known institutions have the opportunity to disseminate high-quality information about computing. MIT's OpenCourseWare initiative (<http://ocw.mit.edu/courses/>) is an excellent example. However, given the diversity of the professional domains engaged in end-user programming and their respective differences and contextual needs, it is unlikely that providing online materials alone will be adequate.

We should also actively lead the development of non-traditional educational opportunities offering context-specific courses or seminars that include conceptual instruction in computing. Rather than developing these in isolation, we must reach out and collaborate with groups of informal learners. Many end-user programming activities make use of scripting or programming affordances within commercial software packages, and most cities have active user groups for such software tools. Building collaborative partnerships with these groups to develop, distribute, and disseminate contextualized computing education has the potential to impact many of these previously underserved adult learners.

There remain many open opportunities in this space for computing education and human-computer interaction researchers alike. Among them:

- We need a better understanding of the information ecologies upon which various end-user programmers rely, so that we can provide useful information in ways that merge with their existing informal learning strategies.

- New programming environments should be designed that recognize the role of Web search and example code (like Blueprint⁴). Furthermore, they should be evaluated not only for whether they enable users to complete a script, but also for the degree to which they can promote learning of computing concepts.

- We can build on the roles that Web-based forums, online communities, and other forms of social media (for example, Twitter, Facebook) play in knowledge exchange among informal learners.

A final charge for educators pertains to the future generation of young professionals. We must rethink the design of our computer science courses to invite students from other disciplines across campus to gain a solid foundation in computing as it pertains to their fields of study. A fundamental aspect of this should be increased interdisciplinary and collaborative educational offerings with other STEM and non-STEM departments. By broadening our own definitions of computing education both on and off campus we might ensure that the next generation has the computational skills and general knowledge needed to succeed in their professions, whatever those professions might be. ■

References

1. Brandt, J. et al. Example-centric programming: Integrating Web search into the development environment. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, 2010, 513–522; <http://doi.acm.org/10.1145/1753326.1753402>
2. Bransford, J. et al. *How People Learn: Brain, Mind, Experience, and School*. National Academy Press, Washington, D.C., 2000.
3. Dorn, B. A case-based approach for supporting the informal computing education of end-user programmers. Doctoral dissertation, Georgia Institute of Technology, 2010.
4. Dorn, B. and Guzdial, M. Discovering computing: Perspectives of Web designers. In *Proceedings of the Sixth International Workshop on Computing Education Research (ICER '10)*. ACM, New York, NY, 2010, 23–30; <http://doi.acm.org/10.1145/1839594.1839600>
5. Scaffidi, C. et al. Estimating the numbers of end users and end user programmers. *IEEE Symposium on Visual Languages and Human-Centric Computing*, 2005, 207–214.

Brian Dorn (bdorn@hartford.edu) is an assistant professor of computer science and multimedia Web design and development at the University of Hartford in West Hartford, CT.

Copyright held by author.

Law and Technology

Bell Labs and Centralized Innovation

Replaying the long-term costs of monopolized innovation.

IN EARLY 1935, a man named Clarence Hickman had a secret machine, about six feet tall, standing in his office. Hickman was an engineer at Bell Labs, and his invention was, at the time, a device without equal on earth, way ahead of its time. Here's how it worked: in the event you called and Hickman was out, the machine would beep and a recording device would come on allowing the caller to leave a message.

What was truly interesting about Hickman's answering machine was not just the idea of a machine that answered calls, but rather, what was in its guts. For inside Hickman's machine was something new to the world: magnetic recording tape. Recall that before magnetic storage there were few low-cost means to store sound other than by pressing a record or making a piano roll. Over the long run, magnetic recording technology would not just herald audiocassettes and videotapes, but when used with the silicon chip, make computer storage a reality. Magnetic recording technology must be counted, in fact, as one of the most important inventions of the 20th century. For, from the 1980s onward, firms from Microsoft to Google—and by implication all the world—would become utterly dependent on magnetic storage, otherwise referred to as the hard disk.

Yet, there is something different about this story—the answering machine would not appear in American homes until the 1980s. What happened in the meantime, as we shall see, sheds



light on central questions of innovation in the 20th century that remain central in the 21st century. The history of the answering machine forces us to confront the costs and benefits of monopoly in the information industries. It is also a question of growing impor-

tance a technological age increasingly dominated by large firms like Google, Microsoft, and Facebook.

Bell Labs

That Bell Labs played a major role in inventing magnetic recording tape

ACM Transactions on Accessible Computing



This quarterly publication is a quarterly journal that publishes refereed articles addressing issues of computing as it impacts the lives of people with disabilities. The journal will be of particular interest to SIGACCESS members and delegates to its affiliated conference (i.e., ASSETS), as well as other international accessibility conferences.

www.acm.org/taccess
www.acm.org/subscribe



Association for
Computing Machinery

is, to any historian of technology, no surprise. Founded in 1925 for the specific purpose of improving telephony, Bell Labs made good on their mission (saving AT&T billions with inventions as simple as plastic insulation for telephone wires) and then some. By the 1920s the laboratories had effectively developed a mind of their own, carrying their work beyond better telephones and into basic research to become the world's preeminent corporate-sponsored scientific body. It was a scientific Valhalla, hiring definitely the right men (and later women) they may find and leaving them relatively free to pursue what interested them.

When scientists are given such freedom, they are able to do amazing things; Bell's scientists did cutting-edge work in fields as diverse as quantum physics and data theory. It was a Bell Labs employee named Clinton Davisson who would win a Nobel Prize in 1937 for demonstrating the wave nature of matter, an insight more typically credited to Einstein than to a telephone company employee. In total, Bell would collect seven Nobel Prizes, more than any other corporate laboratory, including one awarded in 1956 for its most renowned invention—the transistor—which made the computer possible. Other Bell Labs creations, while obscure to the general public, are certainly dear to *Communications* readers, including Unix and the C programming language.

In short, Bell Labs was a superb

The story of Bell Labs is, in many ways, the strongest case in support of the near-inevitable monopolies that emerge in the information industries.

force for good. It is the kind of thing, in fact, that gives monopoly a good name. In current-era usage, the word “monopoly” is a scary concept, one that few would dare endorse publicly. But AT&T was, in its time, a proud monopolist, and even a critic is forced to admit a system run by a beneficent monopolist had its advantages. While to some degree Bell Labs served AT&T's interests, it was also run, in part, out of a kind of noblesse oblige. For in a corporate setting, it is often difficult to imagine how funding theoretical quantum physics research can be of any immediate benefit to shareholder value. More to the point, it is very difficult to imagine a phone company today hiring someone to be their quantum physicist, without rules and with no boss.

The story of Bell Labs is, in many ways, the strongest case in support of the near-inevitable monopolies that emerge in the information industries. Yet, despite all of the undeniable glory of Bell Labs, when you look carefully at the history there emerge little cracks inside the resplendent façade of corporatism for the public good. For however many its breakthroughs, there was a technique through which the institution was very different from a research university. For when the interests of AT&T were at odds with the advancement of information, there was no doubt as to which good prevailed. And so, interspersed between Bell Labs' public triumphs were its secret discoveries, the skeletons within the imperial closet of AT&T. And here we clearly see the long-term costs of industrial rule by a single firm.

Let's return to Hickman's magnetic tape and the answering machine. In the U.S. and in most of the world, answering machines were not widely sold until the 1980s—almost 50 years after Hickman's invention. Why not? Well, soon after Hickman had demonstrated his invention, AT&T ordered its Labs to cease all research into magnetic tape. In fact, Hickman is virtually unknown to history: his research was so effectively suppressed and concealed that it came to light only in the 1990s, when a historian named Mark Clark found Hickman's laboratory notebook in the Bell archives. Magnetic tape would come to

America through imports of German technology after World War II.

Why on earth would management bury such a vital and commercially valuable discovery (one that, in fact, could have aided the war effort)? Here AT&T had, in-house, the chance to dominate a new and incredibly important market. What was it frightened of? The answer, rather surreal, comes from the corporate memoranda, also unearthed by Clark. As odd as it may sound, AT&T firmly believed magnetic tape and the telephone were fundamentally incompatible technologies. The widespread usage of magnetic recording technology, AT&T believed, would lead Americans to abandon the telephone.

In Bell's imagination, the very knowledge that it was possible to record a conversation would "greatly restrict using the telephone," with catastrophic consequences for its business. The firm specified two particular dangers. Businessmen might fear the possible use of a recorded conversation to undo a written contract. Second, AT&T estimated that the telephone was used for an enormous number (Clark quotes an estimate of approximately two-thirds of all calls) of obscene, indecent, or ethically dubious conversations. The very possibility of a recording, AT&T reasoned, would scare off any such users. Hence magnetic recording would "change the total nature of telephone conversations" and "render the telephone much less satisfactory and useful inside the vast majority of cases during which that's employed."

Here we see great problem with monopolized invention: The enlightened planner of the future can also, at times, prove a delusional paranoid. True, once magnetic tape arrived in America, there were several notorious examples—from Nixon to Lewinsky—where sordid secrets were exposed by it. But, amazingly enough, we all still use telephones. Yet in the 1930s it seemed safer to shut down an exhilarating line of research than to risk the Bell system.

The story of AT&T and the answering machine holds important lessons, both for how innovation happens, and the underlying questions of industrial structure. There has been, and per-

Bell Labs was never a place that could originate technologies that could, by the remotest possibility, threaten the Bell system itself.

haps will always be a strong allure to what can be termed "centralized" innovation as epitomized by Bell Labs. It is attractive to envision a planned, systematic means of finding the future, as directed by a great centralized intelligence.

In contrast, the alternative, an open, decentralized approach to innovation—hundreds or thousands of solo inventors or small firms—is superficially much less attractive. It seems so chaotic and underresourced that it is hard to imagine anything of real value being produced. And yet when you look carefully at the history of the communications and computing industries in particular, it is so often the outsider and even outcasts, working in attics or garages, who invent the "big ones." That, at least, is the story, a minimum, behind the telephone, radio broadcasting, the television, cable television, the personal computer, and so many of the Internet's most important firms.

Centralized systems of innovation are excellent for certain types of research. Yet they also have, as it were, one fatal flaw, one that we can see clearly in the story of AT&T and its Bell Labs. Yes, Bell Labs was great. But at the same time, Bell Labs was never a place that could originate technologies that could, by the remotest possibility, threaten the Bell system itself. The truly disruptive technologies—those that might even cast a shadow of uncertainty over the business model—were out of the question.

This is also why, for example, AT&T

never invented the Internet, even though it clearly had the chance. In the 1960s, men like scientist Paul Baran spent years trying to convince AT&T that packet-switching technologies were a step forward that would improve the network, but AT&T regarded the idea as preposterous. "Their attitude," Baran said in a later interview "was that they knew everything and nobody outside the Bell System knew anything. So here some idiot comes along and talks about something being very simple, who obviously doesn't know how the system works."

Packet networking and the recording machine are just two examples of technologies that AT&T, out of such fears, would for years suppress or fail to market: other examples include fiber optics, mobile telephones, digital subscriber lines, facsimile machines, speakerphones—the list goes on and on. These technologies, ranging from novel to revolutionary, were simply too daring for Bell's comfort. Without a reliable sense of ways they would affect the Bell system, AT&T and its heirs would deploy each with painfully slow caution, if at all.

Conclusion

Can we expect the same kind of problems in our contemporary Internet age? On the one hand, it seems difficult to imagine, with a seemingly never-ending parade of startups, that any technology could ever be marginalized for that long. On the other hand, as the power of the main Internet platforms increase—like Facebook, Google, and Apple—their respective interests in discouraging or co-opting certain lines of innovation may slowly increase. It is certainly too early to raise any sort of alarm. But we should never forget that the greatest threat to any dominant firm is a disruptive technology. As Joseph Schumpeter once said of all industries, when faced with anything truly new, "the forces of habit raise up and bear witness against the embryonic project." ■

Tim Wu (wu@pobox.com) is a professor of law at Columbia Law School and author of *The Master Switch: The Rise and Fall of Information Empires*. Knopf Doubleday Publishing Group, 2010.

© 2011 ACM 0001-0782/11/05 \$10.00

Interview

An Interview with Steve Furber

Steve Furber, designer of the seminal BBC Microcomputer System and the widely used ARM microprocessor, reflects on his career.

STEPHEN BYRAM FURBER is the ICL Professor of Computer Engineering in the School of Computer Science at the University of Manchester, U.K. Furber is renowned for his work at Acorn Computers Ltd., where he was a principal designer of the BBC Microcomputer System and the ARM microprocessor, both of which have achieved unique historical significance.

The BBC Micro platform was fundamental to computing in British education in the 1980s and directly led to the development of the ARM architecture. The ARM architecture is the most widely used 32-bit RISC architecture and the ARM processor's power efficiency—performing the same amount of work as other 32-bit processors while consuming one-tenth the amount of electricity—has resulted in the widespread dominant use of the ARM processor in mobile devices and embedded systems.

Furber is a Fellow of the Royal Academy of Engineering, of the Royal Society, the IEEE, the British Computer Society and the Institution of Engineering and Technology (IET), and was appointed Commander of the Order of the British Empire (CBE) in 2008.

Jason Fitzpatrick, a computer historian and the curator at the Centre for Computing History at Suffolk, U.K., conducted an extensive interview with Furber on behalf of the museum, which is dedicated to creating a permanent public exhibition telling the story

The BBC Micro was just the front end of something that was designed as a dual processor from the outset.

of the information age (see <http://www.computinghistory.org.uk/>.) A video of the interview is available at <http://www.computinghistory.org.uk/det/5438/Steve-Furber-Interview-17-08-2009/>; a condensed version of the interview is presented here.

I'd like to talk to you about your involvement with Acorn, and what it's led to today.

I was at the University [in Cambridge]; I read maths as an undergraduate and I went on to do a Ph.D. in aerodynamics. During my Ph.D. I got interested in aspects of flight, and then I heard about the formation of the Cambridge University Processor Group. I thought maybe I should join up with these guys and see if I could build myself a flight simulator or something like that. I was involved in the University Processor Group from its foundation although I wasn't actually

a founder. I went along to the very first meetings and started building computers for fun, which was fairly scary in those days because the components had to be ordered from California by mail order using a credit card. I was a student so credit cards were fairly scary then; using them internationally was even scarier. But we got the microprocessors. My first machine was based on the Signetics 2650, which not many people have heard of these days. It had a full kilobyte of static RAM main memory. I assembled the circuit board using Verowire, which is a little wiring pen where you hand-wired the things together; you soldered it, which melted the insulation and made the connections. I understand it gave off carcinogenic vapor, but it hasn't got me yet.

That's how I built these things. I built myself a small rack—I couldn't afford a commercial rack, so I made one and got the 2650 system going. In the Processor Group, enthusiasts exchanged notes with each other. I remember Sophie Wilson coming to my house for one meeting of the Processor Group, looking at my machine and poking away at it—finding faults in the memory and stuff like that. Then while I was still a Ph.D. student in the Engineering Department, Hermann Hauser came knocking on my door and explained that he and Chris Curry were thinking of starting a consultancy company in the microprocessor business. They had been looking to the University Processor Group as the source of

technical people who might be able to help them; he asked me if I was interested. I said, “Well, I’m just a hobbyist. I’ve been doing this for fun. But if you think I can help, I’m willing to give it a go.” That’s how I joined the embryonic Acorn, before it was Acorn.

Was it based inside Sinclair’s building at the time?

Yes, the first things we did were in the Science of Cambridge Building in King’s Parade. Chris Curry was set up running Science of Cambridge with Clive. Hermann and Chris did bits of Acorn work in there. In fact the first thing I did for Acorn was actually not for Acorn, it was for Science of Cambridge. I hand-built the prototype MK14; I got a circuit diagram and built one using Verowire, soldering in my front room. The MK14 was basically a copy of the National Semiconductor SC/MP development kit. They had taken what was a masked program ROM from the development kit and copied it into two fusible link PROMS for the MK14, and they managed to copy it wrong. So I debugged this thing in my front room. That was the first piece of work I did for them.

Then Chris and Hermann got a contract to do some development work on microprocessor controlled fruit machines, which were very new at that time. Up to that date fruit machines had all been controlled by relays and so on; this was an early attempt to do microprocessor stuff. We used two SC/MPs in a rack to control the fruit machine. In fact, the software for that was bootstrapped from the 2650 machine I built in the Processor Group; it was used as a dumb terminal into the SC/MP development kit, and we brought this fruit machine controller up. The main challenge in those days was to make these things robust. Very early on people had discovered if you just sparked electronic cigarette lighters next to the fruit machine, they would often pay out.

Yes, the program counter jumps off somewhere and anything can happen!

Yes. So that was when Sophie Wilson came in. She designed an FM receiver front end that would trigger whenever you flicked one of these cigarette lighters and cause the SC/MPS to reset; it would definitely not pay out. [laughs] That was the requirement—if

you interfered with it, it should definitely not pay out. The things were tested by plugging a mains adapter into the wall, plugging the fruit machine into one socket, and an arc welding transformer into the other. Somebody welded metal together while you operated the fruit machine to see if the thing was robust to sparking.

Fantastic! The feeling at that time was very much of the hobbyist. You just enjoyed doing that kind of thing, and the whole industry has pretty much come out of that. Is that fair to say?

Yes, that’s right. We are talking about the late 1970s before the IBM PC started, before the Apple II had appeared. There were some very basic box machines. I

think the Altair had probably appeared about this time in the States.

In the University Processor Group, the real men built computers with TTL. It was only the wimps like me that used newfangled microprocessors, which were kind of cheating because you got too much functionality in one package. But yes, microprocessors were just entering the public consciousness, so the MK14 from Science of Cambridge was an example of a microprocessor on a printed circuit board with a hexadecimal keypad and seven segment display; you could put assembly code into it and make it run. Sophie saw the MK14 and said something which she said many times—basically, “I could do better than that.”—and she went home over



Easter holiday and came back with a design she called the Hawk, which was 6502-based. Hermann looked at this and thought he could sell it; that became the Acorn System 1. The name Acorn was introduced originally just as a trading name. The company was called Cambridge Processor Unit Ltd.

If you look at those machines today, the System 1 and the MK14, they are what most people would describe now as unusable. But these things sold in a big way.

The System 1 and the MK 14 sold faster than people could put the kits together. I think the System 1 was mainly sold as a kit, so you got the parts and you had to solder it together. But there was lots of interest. It was really the only way the general public could get their hands on anything that looked like a computer at that time. Real computers cost a million pounds, lived in clean rooms, and were only touched by men in white coats; whereas these things you could buy for £100 or £200 and play with at home.

It was just the want to own and control one of these things. A lot of it was driven by science fiction...

Of course, the real science-fiction aspect is they got used as props in TV shows as well. So the Acorn System 1 was featured as the computer on “Blake’s 7.” There was quite a lot of competition between Acorn and Sinclair at the time. Clive Sinclair had proudly boasted that you could control a nuclear power station with his ZX81. Well, this was nothing compared with controlling a 21st century interstellar cargo ship [on the “Blake’s 7” television program] with an Acorn System 1.

You win, hands down.

That’s right! [laughs]

Going forward to the BBC Micro, obviously the BBC came to Acorn with the specification for a machine? How did that change things at Acorn? The Atom was out and it was selling well. Then all of a sudden you were shot into fame in the computing industry.

The BBC Micro was a huge phenomenon. Of course, when the BBC came their spec was a Z80 machine running CP/M. The BBC Micro was

neither Z80 nor CP/M, although a little bit later you could buy a Z80 second processor to run CP/M; we kind of met the spec in the end. But no, they were sufficiently convinced by what we could do with the 6502 that they moved the spec to the machine that Acorn had already begun to get on the drawing board. The Proton was always designed as a dual processor. The fact the BBC Micro had a second processor connection was actually because the BBC Micro was just the front end of something that was designed as a dual processor from the outset.

I remember that when the BBC was talking to Acorn—I wasn’t involved in the commercial discussions, I was just a techie—they were confident this machine would sell, and on the back of their programs we’d sell 12,000 of these machines. That was big numbers to Acorn. Not huge numbers—we’d probably sold several thousand Atoms—but it was really worth going for. Nobody imagined that that estimate would be off by a factor of a hundred—one and a half million were shipped in the end—because nobody really anticipated the wave of interest.

I really realized that this was a phenomenon when Sophie, Chris Turner, and I agreed to do a seminar at the IET in Savoy Place in London. It has a big central amphitheatre that sits about 500 or 600 people. They asked us to do a seminar on the BBC Micro. We went down there thinking this is a big room, I wonder if they’ll fill it? Three times the number of people they could get in the room turned up. People booked coaches from Birmingham; they had to be sent home because Health and Safety said they

couldn’t let them in. Nobody was expecting this, either.

The machine was first sold in January of 1982, so this may have been later in 1982.

And that was the first time you thought this is big?

Well, this is when you first felt the scale of public interest. People were prepared to hire a coach from Birmingham to hear this bunch of techies, who probably didn’t know how to speak in public, say something about this hobbyist computer thing. Of course, we always had the education market in mind, but this was much bigger than just the education market in terms of interest. We actually went on tour with this seminar. We gave it twice more at the IET to soak up demand. We did a tour of the U.K. and Ireland. Everywhere we went there was a big turnout. There was real, real interest.

What sort of people were coming to this?

A wide range of people. I think it is the same phenomenon as with the System 1 but on a bigger scale. It was a bunch of people who recognized that computers were about to come within their reach, when they’d been behind closed doors throughout past history. There were lots of companies building machines at the time. We’ve mentioned Sinclair, but if you go and look at the machines coming out, the 1980s was a real era of diversity. Wonderful quirky machines of all shapes and flavors, all coming out of companies a bit like Acorn: small start-ups; enthusiasts the public couldn’t perhaps fully trust. Unless you were a hobbyist and a real enthusiast yourself, you didn’t know who to trust.

Then the BBC put their name on this machine from Acorn. I think that really was the key to the success of the BBC Micro, even though by the standards of the competition it was a slightly expensive machine. It was slightly higher spec and that was partly the BBC’s requirements. The BBC imposed—no, imposed is the wrong word—encouraged us to go with a particular spec. The spec was all negotiated and agreed; there was no imposition. But they were tough negotiations. The BBC had a pretty clear idea of what they wanted. The fact that we pushed

We always had the education market in mind, but this was much bigger than just the education market in terms of interest.

the technology a bit—you know, we ran the memory a bit faster, we made an early use of semi-custom chips, we had these two ULAs that gave us quite a lot of excitement—all pushed the price of the machine up a bit. It was an expensive machine, but despite that, a much broader market was comfortable with it because it had the BBC name.

The machine itself was low level. If you connected something to the back, like a joy stick, it was easy to communicate with it because you weren't, as you would be with today's PC, so high up in the software stack. The BBC Micro was easy and anybody could read the user guide; if they had a little bit of technical know-how they could build their own electronics to go into the one megahertz bus and program it. It was easy. I'm quite sad we've lost some of that. I think it's much tougher now for teenagers with technical interest to say, "I want to build one of these things, plug it into my computer and make it do stuff."

Yes, I had a BBC Micro and connected up various different bits of equipment to it and had to program this thing down to the metal, talking directly to the chips; you're talking to the very pins that you just wired up something to. Like you say, that's completely gone. You have to go through layers and talk through Windows, etc. How does one get into hardware programming now? We're running so fast that you can't even make circuit boards to connect to it because the frequency just won't have it.

You're right; the problem isn't simply the stacks of software. The problem is the hardware now...the signal-level hardware is very difficult to work with and requires a lot of skill and knowledge. But of course, you could still build a slow interface. If you want an interface at a megahertz, then connecting to a megahertz is no harder now than it was in 1980. Actually, a megahertz means you can do a million things a second, which is quite a lot for many purposes. So how would people access that? I guess they get some of it through the Lego robotic systems and so on. But in a sense, a bit too much of that is prefabricated. It's a bit too ready-to-go. But you are talking at quite a low level to the microcontroller. There's no reason now why we can't make BBC-like products

The BBC imposed—no, imposed is the wrong word—encouraged us to go with a particular spec.

using microcontroller technology. In fact, you could probably make a BBC Micro in a single chip and sell it very cheaply. I can still run BBC Micro programs on my laptop. In fact I have BBC BASIC for Windows, which is a very faithful emulation. It's produced by Richard Russell, who is one of the people who worked for the BBC when we were negotiating with them in the 1980s. I have an Archimedes emulator that I run on the PC, and I have a BBC emulator that I run on the Archimedes emulator that runs on the PC.

Emulating emulation!

And it's still faster than the original BBC Micro because that's how much compute power we've got now. So you can still run the stuff, but you can't interface hardware to it so easily.

There's an interest in actually building something much more basic again around today's microcontroller technology. My guess the market for that is small but not insignificant.

I think the nearest thing you get at the moment is a PIC development kit for about £20; with a PC you can write PIC Assembly code, load it into the PIC, and do all sorts of stuff with it. I do know quite a few people who use PICs this way. That's probably the nearest you get. But with a PIC, in some sense you're going to a lower-level interface than on a BBC Micro.

To go back to the point about the price of the BBC, it was expensive. You had Spectrums, which were under half the price. Was a big part of the negotiations trying to keep the price down?

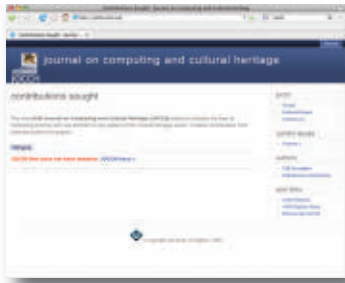
Yes. You may remember that we launched (I may get the numbers wrong here) the Model A at £239 and the B at £339, but really we couldn't make

them at that price. So it went up to £299, £399. We were trying hard to keep the price down. But really, how could you make the machine cheaper? If you look at a Spectrum, then of course everything about it is cheaper, including the keyboard. The BBC had by today's standards a stunningly high-quality keyboard. Today's keyboards are very cheap elastomeric. Every switch on the BBC Micro keyboard had a pair of gold wires that touched when they crossed. When you pushed the key down they sprang together and touched, so you weren't making the contact with the key press; you were removing the obstruction that was preventing the contact. And BBC keyboards were formidably robust. There were some manufacturing problems with them. There was a batch that were manufactured with the key switches about half a millimeter off the PCB; when you hammered on them the force got transmitted through to the solder joints on the back then the PCB tracks broke. But when they were made properly, they would last 10 years of kids thrashing them. The machine took a hammering. You know, mine still works.

They're fantastic machines. So what did the success of the BBC Micro do to Acorn?

Acorn grew rapidly. When the BBC contract was signed, I was still a research fellow at the University; my day job was doing aerodynamics. I guess Acorn at that stage employed maybe 30 people. I joined them full time in October 1981, and by 1983 the company had grown to 400 people, just to manage this stuff. There was no real sales activity involved because the stuff sold faster than you could make it—it just walked off the shelves. So it was really building the technical team up; and we had a strong manufacturing team. If you're going to make millions, you want to know something about procurement. Acorn didn't actually manufacture; it was all sub-contracted. We had to build skills in manufacturing. The BBC contract also required lots of exotic technology beyond the basic machine. The Prestel telesoftware receiver was a bit of unknown technology we had to make; the second processors including the Z80 running CPM. There were a lot of things to go around it, so we grew technical teams.

ACM Journal on Computing and Cultural Heritage



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

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



Association for
Computing Machinery

We also realized the price point was quite good for schools and professional users, but it was too high for hobbyist and most home users, so we developed the Electron, which was a cost-reduced BBC Micro - not an entirely happy story. There wasn't much wrong with the machine, but for Christmas 1983 when there would have been a huge market we couldn't get the electronics reliable enough. By 1984 when we cracked it and could make lots of them, the market had gone. We ended up with a quarter of a million of them in the warehouse that were eventually sold below cost.

But yes, the company went from a small, experimental start-up to big, established...well, is 400 big—medium-sized maybe?

What was the atmosphere like with the people that worked there? Or was the excitement lost in the numbers of people?

No, there was a core group who did the really ambitious technical stuff. We did begin to get the idea that we could take on anything. We knew there was competition. We were a bit over-focused on Sinclair as the competition. Standard problem: seeing the parochial competition and missing the real competition from much further away. Technically we felt very much on top of what we were trying to do, and we kept taking on bigger and bigger challenges. We felt we'd developed the Midas touch when it came to advanced technology; that's a lot of the background to the development of the ARM microprocessor. This was a very short period of time when you look back. The first sale of the BBC Micro was January 1982. The ARM design started in October 1983 and the first ARM chips were in our hands in April 1985. It was only four years from beginning to sell BBC Micros to having ARM chips in our hands.

Going on to the ARM chip, how did that come about? What started you guys working on a new type of processor?

The advanced R&D group, which I was in with Sophie and several other folk, was responsible for looking furthest out in terms of what the company was going to do for product. Hermann was very hands-on—he was always

very involved in this. We were already thinking the BBC Micro has been a big success; we need to build on this. We could put second processors on, which tided us over for a bit. But really we needed to be thinking about the next big machine. It was clear that we were going to step up from 8 bits. 16-bit processors were already around and going into some competing products.

We looked at the 16-bit processors that were around and we didn't like them. We built test bench prototype machines with these. We didn't like them for two reasons. Firstly, they were all going to 16 bits by adopting mini-computer-style instruction sets, which meant they had some very complex instructions. The anecdote I always remember is that the National Semiconductor 32016 had a memory-to-memory divide instruction that took 360 clock cycles to complete; it was running at 6 megahertz, so 360 clock cycles was 60 microseconds; it was not interruptible while the instruction was executing. Single density floppies, if you handled them chiefly with interrupts, give an interrupt every 64 microseconds, double density every 32. Hence you couldn't handle double density floppy disks. The complex instruction sets gave them very poor real-time response.

The second problem we had was that none of them kept up with memory. Commodity memory at the time was, as it still is today, DRAM—it was rather smaller DRAM chips—and this DRAM had a certain bandwidth. If you ran the DRAM at full spec, you get a certain bandwidth. We deduced from a number of experiments that compute power goes with memory bandwidth. But these microprocessors wouldn't even use the bandwidth that was there; they couldn't keep up with the memory. This struck us as the wrong answer.

So we were feeling unsatisfied with the microprocessors we could go out and buy when I'm pretty sure it was Hermann who dropped a couple of papers on our desks, which were early RISC papers from Berkeley and Stanford, where a grad class had designed a microprocessor that was competitive with the best industry could offer. We looked at this, and thought how would we really like a microprocessor to look? Sophie began tinkering with instruction set architecture, inspired consid-

erably by the Berkeley and Stanford RISC work, but also by what she understood of the 6502, and also what was needed to write a good BASIC interpreter. Sophie had written several BASIC interpreters by then for the Atom, for the BBC Micro, for the 32016 second processor, and so on. She sketched out an instruction set.

Then in October 1983 Sophie and I went to visit the Western Design Center in Phoenix, Arizona. They were designing a slightly extended 6502, the 24-bit address 6502 that became the 65C816. We went in expecting to find big, shiny American office buildings with lots of glass windows and fancy coffee machines. What we found was a bungalow in the suburbs of Phoenix. They hired school kids during the summer vacation to do some of the basic cell design. Yes, they'd got some big equipment, but they were basically doing this on Apple IIs. My strong memory is walking out of there saying to each other, "Well, if they can design a microprocessor, so can we."

We went back and from the tinkering that Sophie had been doing with instruction set design, which Hermann had entirely supported and approved of, we put the project on an official footing.

The other infrastructure aspect of this is that Andy Hopper from the Cambridge Computer Lab, who was a director at Acorn, had persuaded Hermann that if he was serious about staying in the computer business, he needed to get serious about chip design. Andy advised Hermann to get chip design tools from VSLI Technology, and Apollo workstations. They recruited IC designers, a group led by Robert Heaton. I can't remember precisely the order they came in, but Jamie Urquhart came early, Dave Howard, Harry Oldham—all names still associated with ARM. So we'd got these tools and the IC designers, but no chips to design. Sophie and I were thinking we should have a go at designing our own microprocessor.

We looked at this RISC stuff, and thought this is kind of obvious, this is a good idea. So we'll set off using these ideas and try to put something together. But it's clear that big industry has got far more resources; they're going to pick up on these ideas too, we're just going to get squashed underfoot by big indus-

try as it catches up. But if we set about doing this, we'll learn something, we'll understand something about what it takes to build a good microprocessor; and then we'll be better at recognizing a good one when we see it. We didn't expect this to go through. To us, building microprocessors was a black art. The big companies had hundreds of people, and it took them 10 revs of the chip before it started to work sensibly. It just looked like a black hole, and Acorn couldn't afford that size of black hole.

But we got on with it. It turned out there is no magic. Microprocessors are just a lump of logic, like everything else we'd designed, and there are no formidable hurdles. The RISC idea was a good one; it made things much simpler. Eighteen months later, after about 10 or 12 man-years of work, we had a working ARM in our hands, which probably surprised us as much as it surprised everybody else.

In July 1985, we'd had the processor on our bench running for a couple of months; we decided it was time to say something to the public. I rang a journalist and said, "We've been working on this microprocessor design and we've got it working." He said, "I don't believe you. If you'd been doing this, I'd have known.", and put the phone down. [laughs] We'd actually done this in considerable secrecy; the secrecy was so good that we couldn't even persuade people when we got the working silicon in our hands. In terms of timescale, this was all happening at exactly the time when Acorn was going bust and being rescued by Olivetti. I believe Olivetti wasn't told about the ARM when they bought Acorn. When they bought it, we thought, maybe it's time to own up:

My strong memory is walking out of there saying to each other, "Well, if they can design a microprocessor, so can we."

they didn't know what to do with it.

So we're talking about a chip now that is in something like 92% of mobile devices today?

Yes. Around the end of 2007, the ten-thousand-millionth ARM had been shipped, so there are more ARMs than people on the planet. I believe production is currently running at about 10 million a day. It is projected to rise to about one per person on the planet per year within two or three years.

They're mind-blowing numbers. Looking at all this and seeing how it's changed us as people—to have this computing power in our pockets has completely changed the way we are and the way we live our lives. And you played an absolute key part in that. So what does it feel like, to know that you played a big part in it?

It's kind of magic, isn't it? I mean it's largely serendipity. I spent some of my last two years at Acorn trying to work out how to build a business plan for a company that could take ARM out. Acorn's desktop PC business was not big enough to support proper processor development; we needed a bigger market, so I tried to work out how to spin out a company. I could never get the numbers to work. You have to sell millions before the royalties start paying the bills. We couldn't imagine selling millions of these things, let alone billions, which is where we are now. But a lot has happened to make that happen—it hasn't gone there on its own. When the company was spun out, Robin Saxby was brought in, and he and the team evolved this business model, which has been instrumental in its success. Had Apple not come knocking at the door wanting the ARM for the Newton, and Robin Saxby not been brought in to head it up...You know, there are lots of ifs.

If these things hadn't happened, we wouldn't be where we are today. But where are we today? I've been trying to work this out. I suspect there's more ARM computing power on the planet than everything else ever made put together. The numbers are just astronomical. 

Viewpoint

The Importance of Reviewing the Code

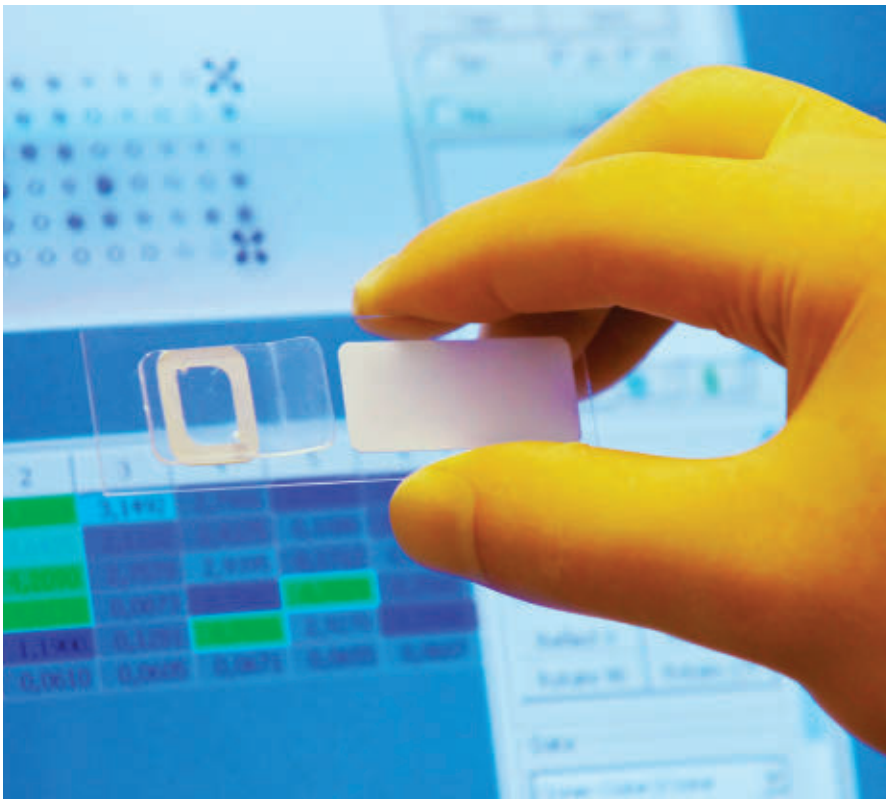
Highlighting the significance of the often overlooked underlying software used to produce research results.

CONTRIBUTORS TO JOURNALS, as well as researchers and politicians, are currently focused on such subjects as open access, data mining, and the growth of the Internet and information technologies together with their associated problems and solutions. At the same time, there is one extremely significant topic in scientific research and computing that is rarely addressed: the importance of the code.

Nowadays, the use of software is essential in many different research fields. It is possible to access a vast amount of research data thanks to the use of computers, software, and storage facilities. If you work in the field of geosciences—as I do—you probably rely on the use of satellite data collected for use by governmental or intergovernmental agencies that has undergone rigorous testing. Normally, there is a peer-reviewed paper to which you

can refer and that you can cite when you use the data. It is possible to create your own scripts and code in order to work with the data, study the results, and formulate a hypothesis about the cause(s) of a phenomenon. In some cases, you might also use software packages that have been developed and released by others, such as spreadsheets and statistical programs. You might also use the functions that are available in your commercially released high-level programming language that make your daily programming tasks easier. When you have computed your results you might use them to publish a paper. Yet how often do reviewers or editors ask about the software used during research? You might receive a large amount of criticism about the statistics, methods, and data when you submit papers for publication, but how often do you receive comments about the software—who cares about that?

Given the lack of comments on software, the issue arises as to whether we are systematically violating basic principles of the scientific method. One such principle is that experiments should be reproducible. Yet it is often the case that reviewers, editors, and other scientists who read your paper cannot reproduce your experiment because they do not have access to essential information about the software you used. In order to address this problem, we must think beyond merely citing the programs and their version numbers. Different programs or different versions of



It is arguable that when using proprietary software it is a question of faith to rely on the results, because it is not possible to check the code.

the same program can make the same computation in different ways, that is, by using different algorithms, some of which will yield results with different degrees of precision or accuracy. It is generally the case that people are simply too willing to believe the results of computations, especially in view of the frequency with which bugs are present in most commonly used programs. Indeed, it is arguable that when using proprietary software it is a question of faith to rely on the results, because it is not possible to check the code (see <http://www.gnu.org/philosophy/categories.html#ProprietarySoftware>).

In light of the foregoing, we may well ask whether we should call for software specifications and code reviewers in scientific publishing. In fact, publishing the software specifications should be a requirement for authors and journal editors. The author's own source code should be published, at least on the Internet, along with the research results, and that source code should be accessible to referees. This does not mean that reviewers should be required to study the code in detail before accepting a paper, because this would require too much work to be viable. However, having the source code available to those who are interested would be a big step forward. In fact, a relatively quick check of the software code by an expert would be beneficial and would encourage authors to place greater emphasis on the reliability of the software they use. This principle should clearly apply to code that one writes oneself. In addition, prepackaged software (whether commercial or not) should be tested, verified, and certified with its code filed and accessible, and checked in detail

by independent programmers or agencies. If such certification were available, it would suffice when submitting a paper for publication to indicate that certified software had been used.

In order to realize the state of affairs described here, the most desirable choice is to use free software (see <http://www.gnu.org/philosophy/free-sw.html>). Free software lets you go into the code and check it. Using free software also follows the spirit of science, in that scientists can disseminate any modifications they make to the code within the scientific community.

Clearly, the challenges involved in applying the framework described in this Viewpoint will vary between different fields of research. However, the amount of work entailed should not be seen as an excuse for not doing it. Furthermore, it can be argued that in some fields of study, the possibility of investigating a phenomenon using different approaches and theories, obtaining similar results, and testing similar hypotheses should be sufficient to render the type of software used unimportant. Yet to argue in this way would be to miss the point. What if the results differ? How do we explain the discrepancy? One possibility is that the difference lies in the software code used. Thus, doing things in the right way, by using free software, will bear fruit. At least it is something we should aspire to, along with what we could call the scientific ideal.

Acknowledgments

The author would like to thank Richard M. Stallman from the Free Software Foundation, Michael McIntyre from the Department of Applied Mathematics and Theoretical Physics at the University of Cambridge, Gerald J. Sussman from the Computer Science and Artificial Intelligence Laboratory at MIT, and Brian Gough and José E. Marchesi from the GNU Project for their useful comments and suggestions. 

Juan Antonio Añel (j.anel@uvigo.es) is Ángeles Alvario Researcher in the Environmental Physics Laboratory at the Universidade de Vigo at Ourense, Spain.

This Viewpoint was accepted for publication in February 2010; in the intervening time prior to publication other material addressing this topic has appeared in *Communications*.

Copyright held by author.

Calendar of Events

May 16–20

International Parallel and Distributed Processing Symposium, Anchorage, AK, Contact: Alan Sussman, Email: als@cs.umd.edu

May 16–20

5th International ICST Conference on Performance Evaluation Methodologies and Tools Communications, Paris, France, Contact: Lasaulce Samson, Email: Samson.lasaulce@lss.supelec.fr

May 16–20

The Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing, Paris, France, Sponsored: SIGMOBILE, Contact: Philippe Jacquet, Email: philippe.jacquet@inria.fr

May 19–21

Computer Personnel Research Conference, San Antonio, TX, Sponsored: SIGMIS, Contact: Cindy K. Riemenschneider, Email: c_riemenschneider@baylor.edu

May 21–28

International Conference on Software Engineering, Waikiki, Honolulu, Sponsored: SIGSOFT, Contact: Richard N. Taylor, Email: taylor@uci.edu

May 23–25

International Symposium on Technology and Society, Chicago, IL, Contact: Keith Miller, Email: miller.keith@uis.edu

May 23–26

5th International Conference on Pervasive Computing Technologies for Healthcare, Dublin, Ireland, Contact: John O'Donoghue, Email: john.odonoghue@ucc.ie

ACM, Advancing Computing as a Science and a Profession



Dear Colleague,

The power of computing technology continues to drive innovation to all corners of the globe, bringing with it opportunities for economic development and job growth. ACM is ideally positioned to help computing professionals worldwide stay competitive in this dynamic community.

ACM provides invaluable member benefits to help you advance your career and achieve success in your chosen specialty. Our international presence continues to expand and we have extended our online resources to serve needs that span all generations of computing practitioners, educators, researchers, and students.

ACM conferences, publications, educational efforts, recognition programs, digital resources, and diversity initiatives are defining the computing profession and empowering the computing professional.

This year we are launching Tech Packs, integrated learning packages on current technical topics created and reviewed by expert ACM members. The Tech Pack core is an annotated bibliography of resources from the renowned ACM Digital Library – articles from journals, magazines, conference proceedings, Special Interest Group newsletters, videos, etc. – and selections from our many online books and courses, as well as non-ACM resources where appropriate.

BY BECOMING AN ACM MEMBER YOU RECEIVE:

Timely access to relevant information

Communications of the ACM magazine • ACM Tech Packs • *TechNews* email digest • Technical Interest Alerts and ACM Bulletins • ACM journals and magazines at member rates • full access to the *acmqueue* website for practitioners • ACM SIG conference discounts • the optional ACM Digital Library

Resources that will enhance your career and follow you to new positions

Career & Job Center • The Learning Center • online books from Safari® featuring O'Reilly and Books24x7® • online courses in multiple languages • virtual labs • e-mentoring services • *CareerNews* email digest • access to ACM's 36 Special Interest Groups • an acm.org email forwarding address with spam filtering

ACM's worldwide network of more than 100,000 members ranges from students to seasoned professionals and includes many renowned leaders in the field. ACM members get access to this network and the advantages that come from their expertise to keep you at the forefront of the technology world.

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

Sincerely,

A handwritten signature in cursive script that reads "Alain Chesnais".

Alain Chesnais
President
Association for Computing Machinery



Association for
Computing Machinery

Advancing Computing as a Science & Profession



Association for
Computing Machinery

Advancing Computing as a Science & Profession

membership application & digital library order form

Priority Code: AD10

You can join ACM in several easy ways:

Online <http://www.acm.org/join> **Phone** +1-800-342-6626 (US & Canada)
+1-212-626-0500 (Global) **Fax** +1-212-944-1318

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

Special rates for residents of developing countries:

<http://www.acm.org/membership/L2-3/>

Special rates for members of sister societies:

<http://www.acm.org/membership/dues.html>

Please print clearly

Name

Address

City

State/Province

Postal code/Zip

Country

E-mail address

Area code & Daytime phone

Fax

Member number, if applicable

Purposes of ACM

ACM is dedicated to:

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

I agree with the Purposes of ACM:

Signature

ACM Code of Ethics:

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

choose one membership option:

PROFESSIONAL MEMBERSHIP:

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

STUDENT MEMBERSHIP:

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

All new ACM members will receive an
ACM membership card.
For more information, please visit us at www.acm.org

Professional membership dues include \$40 toward a subscription to *Communications of the ACM*. Student membership dues include \$15 toward a subscription to *XRDS*. Member dues, subscriptions, and optional contributions are tax-deductible under certain circumstances. Please consult with your tax advisor.

RETURN COMPLETED APPLICATION TO:

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

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

Satisfaction Guaranteed!

payment:

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

☐ Visa/MasterCard ☐ American Express ☐ Check/money order

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

☐ ACM Digital Library (\$99) \$ _____

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

Total Amount Due \$ _____

Card #

Expiration date

Signature

Article development led by [acmqueue](http://acmqueue.queue.acm.org)
queue.acm.org

Finding a lasting solution to the leap seconds problem has become increasingly urgent.

BY POUL-HENNING KAMP

The One-Second War

THANKS TO A secretive conspiracy working mostly below the public radar, your time of death may be a minute later than presently expected. But don't expect to live any longer, unless you happen to be responsible for time synchronization in a large network of computers, in which case this coup will lower your stress level a bit every other year or so.

We're talking about the abolishment of leap seconds, a crude hack added 40 years ago to paper over the fact that planets make lousy clocks compared with quantum mechanical phenomena.

Timekeeping used to be astronomers' work, and the trouble it caused was very academic. To the rural

population, sunrise, midday, and sunset were plenty precise for all relevant purposes.

Timekeeping became a problem for non-astronomers only when ships started to navigate where they could not see land. Finding your latitude is easy: measure the height of the midday sun over the horizon, look at the table in your almanac, done. Finding your longitude is possible only if you know the time of day precisely, and the sun will not tell you that unless you know your longitude.

If you know your longitude, however, the sun will tell you the time very precisely. Using that time, you can make tables of other nonsolar astronomical events—for example, the transits of the





moons of Jupiter, which can then be used to estimate time from that longitude.

This is why Greenwich Observatory in the U.K. and the U.S. Naval Observatory were funded by their respective admiralities. The British empire staked some money on this question, and while the astronomers won on dirty play, the audience vastly preferred John Harrison's chronometers because you did not need to see the transits of the moons of Jupiter to know what time it was. Harrison's chronometer just told you, any time you wanted to know.⁴

Ever since, astronomers have lost ground as "time lords."

Time zones, made necessary by

transcontinental railroads, reduced the number of necessary observatories to nearly nothing. Previously, every respectable city, with or without a university, had somebody whose job it was to figure out proper time. With time zones and a telegraph, you could service all of the United States from the Naval Observatory.

The next loss was the length a second, which astronomers had defined as "1/31,556,925.9747 of the tropical year in 1900," neither a very practical nor very reproducible definition.

Louis Essen's atomic clock won that battle, and SI (International System of Units) seconds became 9,192,631,770 periods of hyperfine radiation from a

cesium-133 atom. A new time scale was created to count these seconds.

Civil time was still kept using a different and varying length of a second, depending on what astronomers had measured the earth's rotation to for each year.

Having variable-length seconds did not work for anybody, not even the astronomers, so in 1970 it was decided to use SI seconds and do full-second step adjustments—leap seconds—starting January 1, 1972.² In practice, this works by astronomers sending the rest of the world a telegram twice a year to tell us how long the last minute of June and December will be: 59, 60, or 61 seconds.

There is a certain irony in the fact

that the UTC (*Universal Time Coordinated*) time scale depends on the rotation of one particular rock in the less fashionable western part of the galaxy. I am pretty sure that, should humans ever colonize other rocks, leap seconds will not be in the luggage.

How Leap Seconds Became a Problem

Until the advent of big synchronized networks of computers, leap seconds bothered nobody. Many computers used the frequency of the electrical grid to count time, and most had their time initially set from somebody's wristwatch. The number of people who actually cared probably numbered fewer than two dozen worldwide.

Therefore, Unix didn't bother with leap seconds. In the `time_t` definition from Unix, all minutes have 60 seconds, all hours 3,600 seconds, and all days 86,400 seconds. This definition carried over to Posix and The Open Group where it is presumably gold-plated for all eternity.

Then something shifted deep under the surface of the earth. We can only guess what it might have been, but there was no need for leap seconds for seven straight years: from the end of 1998 to the end of 2005. This was, more or less, the time when the Internet happened and everybody bought PCs with Windows. Most of the people who hacked Perl to implement the dot-com revolution had never heard of leap seconds.

This is what Microsoft had to say on the subject of leap seconds: "[...]after the leap second occurs, the NTP (Network Time Protocol) client that is running Windows Time service is one second faster than the actual time."³

Unix systems running NTP will paper over the leap second, but there is no standard that says how this should be done. Your system might do one of the scenarios shown in the accompanying figure.

Or it might do something entirely different. Some systems have resorted to slowing down the clock by 1/3600th for the last hour before the leap second, hoping that nobody notices that seconds suddenly are 277 microseconds long.

That's in theory. In practice it depends on the systems getting notice of the leap second and handling it as intended. In this context *systems* are also the NTP servers from which the rest of the computers get their time: at the 2008 leap second, more than one in seven in the public NTP pool servers got it wrong.

The Effort to "Fix" Leap Seconds

By early 2005 when the first leap second in seven years finally began to look likely, some people started to worry about a "Y2K-lite" event. Some bright person inside the U.S. military-industrial complex thought, "Wait a minute, why do we need leap seconds in the first place?" and proposed to the ITU-R (International Telecommunication Union, Radiocommunication Sector) that they be abolished, preferably before December 2005.

Nice try, but one should never underestimate the paper tiger in a UN organization.

The December 2005 leap second came, Armageddon did not, but it was painfully obvious to everybody who paid attention that there were massive amounts of software that needed fixing, before leap seconds would not cause trouble. Even the HBG time signal from the Swiss time reference system did it wrong.

Another leap second occurred in December 2008, and the situation had not changed in any measurable way, but at least the Swiss got it right this time.

Since then the proposal, known to insiders as TF.460-7, has been the subject of "further study" in "Study Group 7A,"

and all sorts of secret scientific brotherhoods, from AAU to CCTF, have had their chance to weigh in. Many have, but few have clear-cut positions.

What is the Problem with Leap Seconds?

The problem is that more systems care about time at the second level.

Air Traffic Control systems perform anti-collision tests many times a second because a plane moves 300 meters in a second. A one-second hiccup in input data from the radar is not trivial in a tightly packed airspace around a major airport.

Medical products and semiconductors are produced in time-critical processes in complex continuous production facilities. On December 8, 2010, a 70-msec power glitch hit a Toshiba flash chip manufacturing facility, and 20% of the products scheduled to ship in January and February 2011 had to be scrapped: "Once the line is stopped, we can't just resume production," said Toshiba spokesman Hiroko Yamazaki.⁵

Technically, there is no problem with leap seconds that we IT professionals cannot tolerate. We just have to make sure that all computers know about leap seconds and that all programs, operating systems, and applications know how to deal with them.

The first part of that problem is we have only six months to tell all computers and software about leap seconds, because that is all the warning we get from the astronomers. In practice, we often have 10 months' notice; for example, we were told on February 2 that there will be no leap second in December of this year.¹

Unfortunately, this advantage is negated by some time signals—for example, the DCF77 signal from Germany, announcing the leap second only one hour ahead of time.

The other part of the problem—changing `time_t` to know about leap seconds—has nasty results: time is suddenly not a fixed radix quantity anymore. How much code finds the current day by `d = t/86400` or tests if two events are further apart than a minute by `if (t1 >= t2 + 60)?` Nobody knows. How much of such code needs to be fixed if we change the `time_t` definition? Nobody knows.

The Y2K experience indicates it

Sensitivities in leap seconds.

23:59:57	23:59:57	23:59:57
23:59:58	23:59:58	23:59:58
23:59:59	23:59:59	23:59:59
23:59:59	00:00:00	(halt for 1 sec)
00:00:00	00:00:00	00:00:00
00:00:01	00:00:01	00:00:00

would be expensive to find out, because relative to Y2K, the questions are a lot harder than “2 digits or 4 digits.”

How do we tell if code that does `s += 3600` intends this to mean “one hour from now” or “same time, next hour?” The original programmer did not expect there to be any difference, so the documentation will not tell us.

The Cost of Uncertainty

The next time Bulletin C tells us to insert a leap second, probably in 2012, a lot of people will have to kick into action. Any critical bits installed since December 2008 and any bits older than that that failed to “do the right thing” with the December 2008 leap second will need to be pondered, and a plan made for what to do: test, fix, hope, or shut down.

Unsurprisingly, many plants and systems simply give up trying to predict what their multivendor heterogeneous systems will do with a leap second, and they sidestep the issue by moving or scheduling planned maintenance downtime to cover the leap second. For them, that is the cheapest way to make sure that no robot arms get out of sync with the assembly line and that no space-shuttle computers hiccup while in space.

I’m told from usually reliable sources that the entire U.S. nuclear deterrent is in “a special mode” for one hour on either side of a leap second and that the cost runs into “two-digit million dollars.”

But What Do Leap Seconds Actually Do?

Leap seconds make sure the sun is due south at noon by adjusting noon to happen when the sun is due south at the reference location. This very important job is handled by the International Earth Rotation Service (IERS).

Leap seconds are not a viable long-term solution because the earth’s rotation is not constant: tides and internal friction cause the planet to lose momentum and slow down the rotation, leading to a quadratic difference between earth rotation and atomic time. In the next century we will need a leap second every year, often twice every year; and 2,500 years from now we will need a leap second every month.

On the other hand, if we stop plugging leap seconds into our time scale,



There is no problem with leap seconds that we IT professionals cannot tolerate. We just have to make sure all computers know about leap seconds and that all programs, operating systems, and applications know how to deal with them.



noon on the clock will be midnight in the sky some 3,000 years from now, unless we fix that by adjusting our time zones.

Actually, the sun is not due south at noon, and certainly not with a second’s precision, for more than an infinitesimal number of people who are probably totally unaware of it. Our system of one-hour-wide time zones means that only those who live exactly on a longitude divisible by 15 have the chance, provided that their governments have not put them in a different time zone. For example, all of China is one time zone, despite the 75–120E span of longitude.

Of the remaining few lucky people, many are out of luck during the part of the year when their government has decided to have daylight saving time—although that could possibly put a select few of those who lost on the first criterion back in luck during that part of the year. Finally, it is really only a couple of times a year that the sun is precisely due south, for interesting orbital and geophysical reasons.

The people who really do care about UTC time being synchronized to earth rotation are those who use UTC time as an estimator for earth rotation: those who point things on earth at things in the sky—in other words, astronomers and their telescopes, and satellite operators and their antennae. Actually, that should more accurately be *some* of those people: many of them have long since given up on using UTC as an earth rotation estimator, because the ± 1 -second tolerance is not sufficient for their needs. Instead, they pick up Bulletin A or B from the IERS FTP server, which gives daily values with microsecond precision.

The Cost-Benefit Equation

Most of those involved on the “Abolish Leap Seconds” side of the debate claim a cost-benefit equation that essentially says: “cost of fixing all computers to deal correctly with leap seconds = infinity” over “benefits of leap seconds = next to nothing.” QED: case closed.

The vocal leaders of the “Preserve the Leap Seconds” campaign (not to be confused with the “Campaign for Real Time”) have a different take on the equation: “cost of unknown consequences of decoupling civil time from earth rotation = [a lot...infinity]” over

“programmers should fix their past mistakes for free.” QED: case closed.

Not a lot of common ground there, and not a lot of data supporting either proposition, although Y2K experience, as well as the principles of a capitalist economy, dictate that getting programmers to handle leap seconds correctly will be expensive.

A Possible Compromise?

Warner Losh, a fellow time-and-computer nerd, and I both have extensive hands-on experience with leap-second handling in critical systems, and we have tried to suggest a compromise on leap seconds that would vastly reduce the costs and risks involved: schedule the darn things 20 years in advance instead of only six months in advance.

If we know when leap seconds are to occur 20 years in advance, we can code them into tables in our operating systems, and suddenly 99.9% of our computers will do the right thing when leap seconds happen, because they know when they will happen. The remaining 0.1% of the systems, involving ready, cold spares on shelves, autonomous computers on the South Pole, and similar systems, get 20 years to update stored tables rather than six months to do so.

The astronomical flip side of this proposal is that the difference between earth rotation and UTC time would likely exceed the current one-second tolerance limit, at least until geophysicists get a better understanding of the currently not understood fluctuations in earth rotation.

The IT flip side is that we would still have a variable radix time scale: most minutes would be 60 seconds, but a few would be 61 seconds, and code that really cares about time intervals would have to do the right thing instead of just adding 86,400 seconds per day.

So far, nobody has tried, or if they tried, they failed to inject this idea into the official standards process in ITU-R. It is not clear to me that it would even be possible to inject this idea unless a national government, seconded by another, officially raises it at the ITU plenary assembly.

What Happens Next?

Proposal TF-460-7 to abolish leap seconds will come up for plenary vote at the

ITU-R in January 2012, and if it, modulo amendments, collects a supermajority of 70% of the votes, leap seconds would cease beginning in approximately 2018.

If the proposal fails to gain 70% of the votes, then leap seconds will continue, and we had better start fixing computers to deal properly, or at least more predictably, with them.

As I understand the voting rules of ITU-R, only country representatives can vote, one vote per country. If my experience is anything to go by, finding out who votes on behalf of your country and how they intend to vote may not be immediately obvious to the casually inquiring citizen.

The Philosophical Issues

One of my Jewish friends explained to me that all the rules Jews must follow are not meant to make sense; they are meant to make life so difficult that you never take it for granted. In the same spirit, Van Halen used brown M&Ms to test for lack of attention, and I use leap seconds: if a system has not documented and tested what happens on leap seconds, I don't trust it to get anything else right, either.

But Linus Torvalds' observation that “95% of all programmers think they are in the top 5%, and the rest are certain they are above average” should not be taken lightly: very few programmers have any idea what the difference is between “wall-clock time” and “interval time,” and leap seconds are way past rocket science for them. (For example, Posix defines only a `pthread_cond_timedwait()`, which takes wall-clock time but not an interval-time version of the call.)

When a large fraction of the world economy is run by the creations of lousy programmers, and when embedded systems are increasingly capable of killing people, do we raise the bar and demand that programmers pay attention to pointless details such as leap seconds, or do we remove leap seconds?

As an old-timer in the IT business, I'm firmly for the first option: we should always strive to do things better, and do them right, and pointless details makes for good checkboxes. As a frequent user of technological marvels built by the lowest bidder, however, the second option is not unattractive—particularly when the pilots tell us they “have to turn

the entire plane off and on again before we can start all the motors.”

As a time-nut, a small and crazy fraternity that thinks running an atomic clock in your basement is a requirement for a good life (let me know if you need a copy of my 400GB recording of the European VLF spectrum during a leap second...), I would miss leap seconds. They are quaint and interesting, and their present rate of one every couple of years makes for a wonderful chance to inspire young nerds with tales of wonders in physics and geophysics.

But once every couple of years is not nearly often enough to ensure that IT systems handle them correctly.

I wish we could somehow get the 20-year horizon compromise on the table next January, but failing that, if the choice is only between keeping leap seconds or abolishing leap seconds, they will have to go—before they kill somebody through bad standards writing and bad programming. 

Related articles on queue.acm.org

Principles of Robust Timing over the Internet

Julien Ridoux, Darryl Veitch
<http://queue.acm.org/detail.cfm?id=1773943>

You Don't Know Jack about Network Performance

Kevin Fall, Steve McCanne
<http://queue.acm.org/detail.cfm?id=1066069>

Fighting Physics: A Tough Battle

Jonathan M. Smith
<http://queue.acm.org/detail.cfm?id=1530063>

References

1. International Earth Rotation and Reference Systems Service. Information on UTC-TAI; <http://data.iers.org/products/16/14433/orig/bulletinc-041.txt>.
2. International Earth Rotation and Reference Systems Service. Relationship between TAI and UTC; http://hpiers.obspm.fr/eop-pc/earthor/utc/TAI-UTC_tab.html.
3. Microsoft. How the Windows Time service treats a leap second (2006). (November 1); <http://support.microsoft.com/kb/909614>.
4. Sobel, D. *Longitude*. Walker and Company, 2005.
5. Williams, M. Power glitch hits Toshiba's flash memory production line. *ComputerWorld* (Dec. 2010); http://www.computerworld.com/s/article/9200738/Power_glitch_hits_Toshiba_s_flash_memory_production_line.

Poul-Henning Kamp (phk@FreeBSD.org) has programmed computers for 26 years and is the inspiration behind bikeshed.org. His software has been widely adopted as “under the hood” building blocks in both open source and commercial products. His most recent project is the Varnish HTTP accelerator, which is used to speed up large Web sites such as Facebook.



Web apps are cheaper to develop and deploy than native apps, but can they match the native user experience?

BY ANDRE CHARLAND AND BRIAN LEROUX

Mobile Application Development: Web vs. Native

A FEW SHORT years ago, most mobile devices were, for want of a better word, “dumb.” Sure, there were some early smartphones, but they were either entirely email focused or lacked sophisticated touch screens that could be used without a stylus. Even fewer shipped with a decent mobile browser capable of displaying

anything more than simple text, links, and maybe an image. This meant if you had one of these devices, you were either a businessperson addicted to email or an alpha geek hoping that this would be the year of the smartphone. Then Apple changed everything with the release of the iPhone, and our expectations for mobile experiences were completely reset.

The original plan for third-party iPhone apps was to use open Web technology. Apple even released tooling for this in its Dashcode project.⁴ Fast-forward three years and native apps are all the rage, and—usually for performance reasons—the mobile Web is being unfavorably compared.

There are two problems with this line of thinking. First, building a different app for each platform is very expensive if written in each native language. An indie game developer or startup may be able to support just one device, likely the iPhone, but an IT department will have to support the devices that its users have that may not always be the latest and greatest. Second, the performance argument that native apps are faster may apply to 3D games or image-processing apps, but there is a negligible or unnoticeable performance penalty in a well-built business application using Web technology.

For its part, Google is betting on Web technology to solve the platform

fragmentation. Vic Gundotra, VP of engineering at Google, claimed that “even Google was not rich enough to support all of the different mobile platforms from Apple’s App Store to those of the BlackBerry, Windows Mobile, Android, and the many variations of the Nokia platform,”⁶ and this was before HP webOS, MeeGo, and other platforms emerged.

In this article we discuss some of the strengths and weaknesses of both Web and native approaches, with special attention to areas where the gap is closing between Web technologies and their native counterparts.

Native Code vs. Web Code

Implementing a software app begins with code. In the case of native code, most often these days the developer typically writes in a C dialect, as in the case of the iPhone. In our work at Nitobi (<http://nitobi.com/>) and on PhoneGap (<http://www.phonegap.com/>), we have had plenty of experience wrestling with the various mobile platforms from a native development perspective.

Of course, for various market or organizational reasons most developers or teams must support apps on multiple smart platforms. Want to write an app in native code and hit every single mobile operating system? No problem if your team has the skill sets shown in the accompanying table.

What makes things even more complicated are the differences among the actual platform SDKs (software development kits). There are different tools, build systems, APIs, and devices with different capabilities for each platform. In fact, the only thing these operating systems have in common is that they all ship with a mobile browser that is accessible programmatically from the native code.

Each platform allows us to instantiate a browser instance, chromeless, and interact with its JavaScript interface from native code. From within that Webview we can call native code from JavaScript. This is the hack that became known as the PhoneGap technique pioneered by Eric Oesterle, Rob Ellis, and Brock Whitten for the first iPhone OS SDK at iPhoneDevCamp in 2008. This approach was later ported to Android, BlackBerry, and then to the rest of the platforms PhoneGap

supports. PhoneGap is an open source framework that provides developers with an environment where they can create apps in HTML, CSS, and JavaScript and still call native device features and sensors via a common JS API. The PhoneGap framework contains the native-code pieces to interact with the underlying operating system and pass information back to the JavaScript app running in the Webview container. Today there is support for geolocation, accelerometer, and more.

What is native code exactly? Usually it’s compiled, which is faster than interpreted languages such as JavaScript. Webviews and browsers use HTML and CSS to create user interfaces with varying degrees of capability and success. With native code, we paint pixels directly on a screen through proprietary APIs and abstractions for common user-interface elements and controls.

In short, we’re pitting JavaScript against compiled languages. These days, JavaScript is holding its own. This isn’t surprising—JavaScript virtual machine technology is the new front line for the browser wars. Microsoft, Google, Apple, Opera, and Mozilla are all iterating furiously to outperform competing implementations.⁵ Right now, by some benchmarks (<http://arewefastyet.com/>), Mozilla’s SpiderMonkey is closing in on Google’s V8 engine. JavaScriptCore by Apple, found in most WebKit browsers (which is on most mobile devices), is somewhere in-between. The bottom line is that heavy spending by all the major players is fueling this JavaScript arms race. The benchmark by Ars Technica shown in Figure 1 is an example of how these companies are marketing themselves.

JavaScript is rapidly getting faster—so fast, in fact, that HP Palm webOS 2.0 rewrote its services layer from Java to the extremely popular node.js platform (<http://nodejs.org/>), which is built on Google’s V8 engine to obtain better performance at a lower CPU cost (and therefore longer battery life). The trend we’re seeing is the Web technology stack running at a low level, and it’s in production today on millions of devices.

User Interface Code

Things aren’t as pretty when it comes to the user interface. Most native platforms have wonderful abstractions for

common user-interface controls and experiences. No two platforms have the same, or even similar, user-interface paradigms, let alone APIs to instantiate and access them. The Web platform is consistent, for the most part, but the number of built-in or SDK-included controls is limited. You have to roll your own. Sometimes the differences among browsers can cause pain, but—at least in the modern smartphone world—most devices sport the very capable WebKit rendering engine, and only small differences prevail.

Unfortunately for the Web, those small differences are becoming a big deal. For example, on iOS, the CSS position property does not properly support a value of “fixed.” (This was a problem in Android, but has been corrected in the latest Android 2.2 code.) BlackBerry operating systems earlier than version 6.0 sport a completely arcane browser for which there has been much suffering and toil at unfathomable cost to Web developer sanity. Fortunately, RIM has addressed a lot of this in 6.0, and in general, things are getting better.

Some operating systems include something called hardware acceleration. The iOS stack famously supports this concept in CSS transforms, which is how some Web frameworks achieve silky smooth transitions between view states. It’s a technique first uncovered in Dashcode. It was painstakingly reverse engineered by David Kaneda, pioneered in jQTouch (<http://jqtouch.com/>), and released later in Sencha Touch (<http://www.sencha.com/>). Both are incredible Web projects and examples of what can be done when developers push the boundaries.

When we first started tapping into these next-generation mobile browsers, no framework worked properly across devices. Today there are more than 20 mobile frameworks, and support is being rapidly added to existing DOM (Document Object Model) libraries—not the least of which is John Resig’s jQuery (<http://jquery.com/>) and jQuery Mobile (<http://jquerymobile.com/>); that code is improving and adding support for more devices every day. With tools like these, it’s getting easier and easier to support multiple targets from a single Web-oriented code base.

Rapid execution and beautiful user interfaces aren’t the whole story when

contrasting the Web technology stack to native code. Web technology lives in a sandbox, which is also a jail from lower-level APIs that native code can access—APIs that can access device storage, sensors, and data. But this gap is being bridged, too. Most mobile browsers support geolocation today, for example, and iOS recently added Accelerometer and a slew of other HTML5 APIs. Given that the W3C has a Device API Working Group (<http://www.w3.org/2009/dap/>), it's likely we will be seeing many more APIs reach the browser in the near future. If the near future isn't soon enough, you can use PhoneGap (<http://docs.phonegap.com/>) to access these APIs today.

Of course, the Web technology stack (HTML/CSS/JS) is itself implemented in native code. The distance between the native layer and the browser is just one compile away. In other words, if you want to add a native capability to a browser, then you can either bridge it or recompile the browser to achieve that capability. *If a browser does not support a native capability, it's not because it can't or that it won't; it just means it hasn't been done yet.*

User Experience: Context vs. Implementation

Another area that has a big effect on both native and Web mobile application development is *user experience*, the term we use for the overall experience a user has with a software application. User experience can even extend outside the app. For example, we can use push notifications to wake up an application under certain conditions, such as a location change, or to spawn a new purpose-built application to handle different application aspects. Obviously, a successful user experience is crucial for successful application adoption.

Generally speaking, a mobile software project user experience can be divided into two primary categories:

- The *context*—elements that must be understood but cannot be changed or controlled. These include hardware affordances, platform capabilities and UI conventions, and the environment in which your application is used.
- The *implementation*—elements that can be controlled in an application, such as performance, design, and

integration with platform features such as accelerometer data or notifications.

The context in which your application will be used affects users' expectations. The context for a single application may be radically different from one user to the next, even on a single platform. We're not really talking about a context; we're actually talking about *multiple* contexts. Let's look at the things that define the contexts to which a successful mobile application must adapt.

Hardware. The Android device ecosystem (Figure 2) is a fantastic example of this variety of contexts, with devices varying significantly in terms of display (physical size, color depth, screen resolution, pixel density, aspect ratio); input (trackball, touchscreen, physical keyboard, microphone, camera); and

capability (processing power, storage, antennae, and so on.).

The combination of these properties greatly impacts how your application will appear, and the range of possible ways the user might choose to interact with it. If a particular combination doesn't exist today, it very well could tomorrow. A successful application must account for the habits associated with all of these hardware devices.

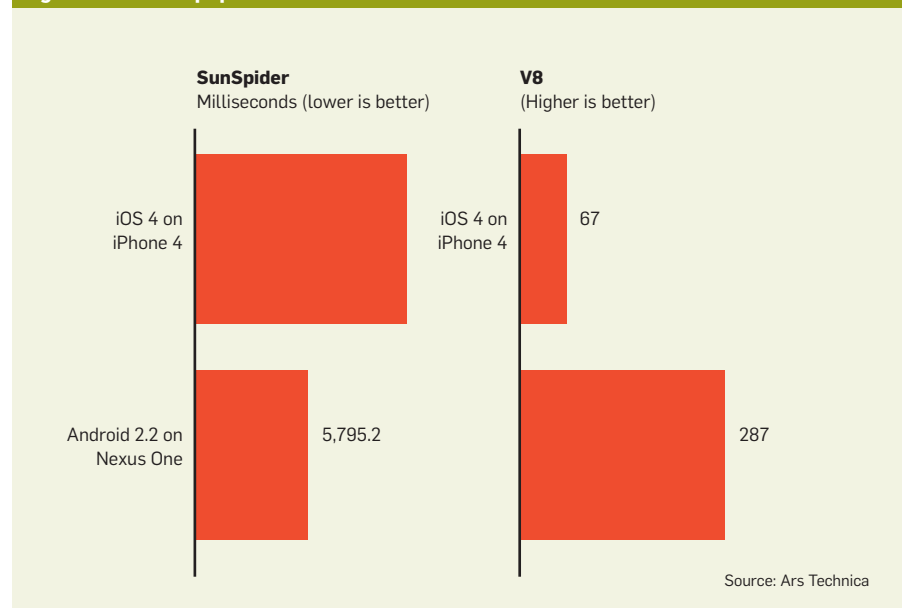
Platform conventions. Each platform has its own user-interface conventions, typically described in a human interface guideline doc and evidenced in the operating-system interface. The variety of mobile Web browsers provides a prime example of how different these conventions can be:

A common user expectation is the ability to “go back” in the browser. iOS

Required skill sets for nine mobile OSs.

Mobile OS Type	Skill Set Required
Apple iOS	C, Objective C
Google Android	Java (Harmony flavored, Dalvik VM)
RIM BlackBerry	Java (J2ME flavored)
Symbian	C, C++, Python, HTML/CSS/JS
Windows Mobile	.NET
Window 7 Phone	.NET
HP Palm webOS	HTML/CSS/JS
MeeGo	C, C++, HTML/CSS/JS
Samsung bada	C++

Figure 1. JavaScript performance: Android 2.2 vs. iOS 4.



satisfies this with a virtual button; Android and BlackBerry devices rely on a physical hardware back button; webOS uses a back button and a back gesture. Whatever the method, users expect that they will be able to “go back” in your application.

Users also expect context menus. In the default Android and BlackBerry browser, the context menu is accessed through a physical button found at the bottom of the screen, close to the natural position of the thumbs. On iOS and webOS the context menu is accessed through a persistent virtual tab bar positioned close to the thumb at the bottom of the screen. The persistent tab bar at the bottom of the screen on devices other than iOS and webOS often produces a poor experience because users can easily accidentally hit their context menu or back buttons, causing an app to close unexpectedly. These are limitations with which both native and Web apps must contend.

Developers must consider approaches that make good sense for both data and users. HTML5 does support the concept of a menu element so a common abstraction is possible here, but the work has yet to be done.

Environment is the greatest wild card of all! Is it daytime or nighttime?

Is the user standing or sitting? Standing still or in motion? One or two hands free? In a busy place? The variables are endless.

Where does that leave us? Expectations borne out of the context are not inherently cross platform. Both native and Web implementations must provide designs and code that support these expectations. The good news for Web developers is that they can fall back on a familiar paradigm in the Web technology stack to satisfy user expectations.

Implementation. To produce the best possible user experience, implementations must deliver designs and code that support expectations set out by a user’s particular context.

Performance: The Hobgoblin of Software Development

Without a doubt, performance is a cornerstone of a great user experience. Like security, it is the most misunderstood and oft-used scapegoat of the software developer. It’s not uncommon to hear developers reject ideas with a flippant, “We can’t do that, it will negatively impact performance.” Rarely quantified and frequently cited, performance is the hobgoblin of software development. How do we quan-

tify performance? Latency is a form of performance. Execution, the time an operation takes to perform, is another. We’ll address these separately.

Latency is a huge consideration in the mobile world. Be it a native or a Web application, there is a performance penalty to downloading an app and the data it consumes or publishes through the network. Obviously, the smaller the payload, the faster the app.

Using JavaScript Object Notation (JSON)-formatted data is a good idea as it tends to result in a smaller data payload compared with an equivalent XML payload, depending on how the XML is formatted. On the other hand, XML data can make sense when returning HTML fragments that are to be inserted into a Web page rather than returning JSON-formatted data that, while smaller over the wire, needs to be converted to an HTML fragment using JavaScript. Your mileage will vary. Benchmarking is the only way to know for sure.

Another latency issue can be the initialization of code. Once we actually get the code into memory, it still needs to be parsed. There can be a noticeable performance penalty in this process. We can fake it and enhance the perception of performance with determinate or indeterminate progress indicators.

Execution time is, of course, a key facet of performance. When interpreting code (as we do for the Web with JavaScript), the more there is to interpret, the longer the execution time. Here the Web technology stack has some catching up to do. JavaScript, for all its leaps in performance, is still slower than native counterparts. On the other hand, the time it takes a programmer to write comparable logic in a native compiled language on multiple mobile devices may be worth the time penalty for execution; however, this will certainly require more maintenance than one code base written in JavaScript that can run on multiple devices, maybe some tweaks per platform. Less code often leads to less and easier maintenance.

That said, the benefit of less code doesn’t matter to the end user, who expects a responsive interface. The developer tradeoff is a larger code base—often vastly larger, considering support for multiple native platforms. In the world of native code, the main challenge is reimplementing to multiple



Figure 2. Variety of contexts across Android devices.

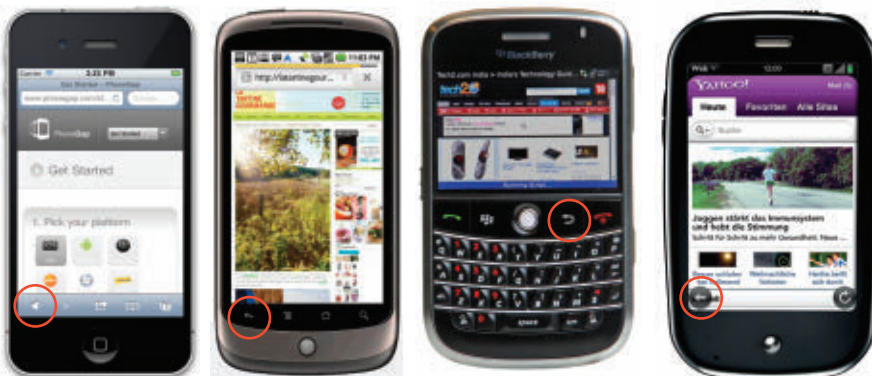


Figure 3. The variety of mobile Web browsers.

targets. In the world of the Web, the main challenge is limiting your footprint as much as possible to produce a responsive user experience. That's not to say that one user interface can suffice in all environments, rather that the majority of the application logic is in one code base and then specific device-specific UI idioms can be implemented with conditional code. You therefore might want to implement slightly different functionality and user experiences appropriate to the expectations of the users of a particular device. For example, Android and BlackBerry devices have physical back and menu buttons, whereas an iOS device does not.

Another key point to remember is that even though the mobile industry is quickly converging on WebKit as the de facto standard for HTML rendering engines, every device and operating system has a slightly different flavor of WebKit. This means you should expect development to be similar to cross-browser Web development today. Thankfully, there are many libraries such as jQuery Mobile, Sencha Touch, and SproutCore that seek to address this.

All of this discussion of latency and execution of code means taking a tough look at the business goals of your application development initiative. Favoring data over decor is the most pragmatic approach. Gradients, drop shadows, bevels, embossing, highlights, rounded corners, and Perl noise do not make an application useful or usable—they don't fulfill a business requirement—but they do impact performance. CSS gradients, in particular, are real devils for performance in the mobile world. You need to decide what your objective is: looking neat or providing a useful interface for data publishing and acquisition. You win some of these capabilities on some platforms with optimized (often hardware-accelerated) pixel painting with native code. It's not that these effects are impossible to achieve, but they should be used judiciously and only when they enhance and do not distract from the user experience. It is possible to deliver a great user experience that succeeds in the market; it just requires proper mobile Web development techniques and good user-experience skills that take into account the constraints of the environment.

Lovely Bounces and Beautiful Design

Of course, beautiful design matters. From aesthetics to intangibles such as the structure of a good program, software designers must commit to great design and to building on solid practices already in place. Scrolling via kinetic physics, lovely bounces, easing, and so forth create reactive interfaces that feel real and are a delight to use. This is an area where native controls are particularly good.

We have yet to solve the problem of native scrolling satisfactorily with Web technology.¹ There have been many attempts: iScroll (<http://cubiq.org/iscroll>), TouchScroll (<http://uxebu.com/blog/2010/04/27/touchscroll-a-scrolling-layer-for-webkit-mobile/>), GloveBox (<https://github.com/purplecabbage/GloveBox>), Sencha (<http://www.sencha.com/>), and jQuery Mobile (<http://jquerymobile.com/>). All of these address the scrolling issue but do not solve it as well as a native device. Even the Google mobile team is working on releasing a solution for this problem.³ Without a doubt, this is the most common complaint the PhoneGap team hears, but we're one bug fix in WebKit away from it being a nonissue. The Google Mobile team has recently released its solution and code for WebKit-based browsers and platforms.²

Here's the rundown. The Web technology stack has not achieved the level of performance we can attain with native code, but it's getting close. We're confident that Web technologies will become indistinguishable from native experiences. In the meantime, Web developers must focus on delivering data while working diligently on improving the decor.

Looking to the Future

As much as native and Web are pitted against one another in this debate, the likely outcome is a hybrid solution. Perhaps we'll see computing as inherently networked and (this is my sincere hope) free for anyone to access. We already see signs of a native Web: WebGL recently proved that in-browser 3D gaming is possible, even running *Quake III* (<http://media.tojicode.com/q3bsp/>).

In the meantime, software makers must balance the Web-vs.-native de-

bate based on an application's primary objectives, development and business realities, and the opportunities the Web will provide in the not-so-distant future. The good news is that until all of this technology makes it into the browser, hacks such as PhoneGap can help bridge the divide. I encourage developers not simply to identify software development trends but to implement them! If the Web doesn't fulfill a capability your particular application requires, you're presented with an exciting opportunity to contribute and close the Web/native chasm in the process. ■

Related articles on queue.acm.org

Case Study: UX Design and Agile: A Natural Fit?

<http://queue.acm.org/detail.cfm?id=1891739>

Enterprise-Grade Wireless

Bruce Zene and Andrew Toy

<http://queue.acm.org/detail.cfm?id=1066065>

Energy Management on Handheld Devices

Marc A Viredaz, Lawrence S. Brakmo,

William R. Hamburgen

<http://queue.acm.org/detail.cfm?id=957768>

References

1. Ecker, C. Ars iPad application redux: where we're going, 2010; <http://arstechnica.com/apple/news/2010/11/ars-application-redux-where-were-going.ars>.
2. Fioravanti, R. Implementing a fixed-position iOS Web application, 2010; http://code.google.com/mobile/articles/webapp_fixed_ui.html.
3. Google Mail Blog. Gmail in Mobile Safari; now even more like a native app, 2010; <http://blogs.msdn.com/b/ie/archive/2010/11/17/html5-and-real-world-site-performance-seventh-ie9-platform-preview-available-for-developers.aspx>.
4. Lee, W-M. Build Web apps for iPhone using Dashcode, 2009; <http://mobiforge.com/developing/story/build-web-apps-iphone-using-dashcode>.
5. MSDN, IEBlog. HTML5, and real-world site performance: Seventh IE9 platform preview available for developers, 2010; <http://blogs.msdn.com/b/ie/archive/2010/11/17/html5-and-real-world-site-performance-seventh-ie9-platform-preview-available-for-developers.aspx>.
6. Nuttall, C. App stores are not the future, says Google. FT Tech Hub, 2009; <http://blogs.ft.com/fttechhub/2009/07/app-stores-are-not-the-future-says-google>.

Andre Charland is the co-founder and CEO at Nitobi Inc. He's been at the forefront of Web 2.0 software development for almost a decade and is an expert on the next-generation Web. He is an advocate for usability and user experience and speaks regularly about how to keep users engaged and active on Web sites or Web-based application. He is the co-author of *Enterprise Ajax* (Prentice Hall) and lead blogger for O'Reilly's InsideRIA.com.

Brian LeRoux is the lead architect at Nitobi Software with the prestigious title SPACELORD. He also has the dubious distinction of being the creator of wtfs.com and crookfordfacts.com. He is also responsible for leading the direction on the PhoneGap free software project that has the ambitious goal to provide a Web platform complete with Device APIs for nearly all smartphone operating systems.

© 2011 ACM 0001-0782/11/05 \$10.00

Article development led by **acmqueue**
queue.acm.org

A Ruby on Rails app highlights some serious, yet easily avoided, security vulnerabilities.

BY PATRICK MCKENZIE

Weapons of Mass Assignment

IN MAY 2010, during a news cycle dominated by users' widespread disgust with Facebook privacy policies, a team of four students from New York University published a request for \$10,000 in donations to build a privacy-aware Facebook alternative. The software, Diaspora, would allow users to host their own social networks and own their own data. The team promised to open-source all the code they wrote, guaranteeing the privacy and security of users' data by exposing the code to public scrutiny. With the help of front-page coverage from the *New York Times*, the team ended up raising more than \$200,000. They anticipated launching the service to end users in October 2010.

On September 15, Diaspora (<https://joindiaspora.com/>) released a “pre-alpha developer preview” of its source code (<https://github.com/diaspora/diaspora>). I took a look at it, mostly out of curiosity, and was struck by numerous severe security errors. I spent the next day digging through the code locally and trying to

get in touch with the team to address them, privately. The security errors were serious enough to jeopardize the goals of the project.

This article describes the mistakes that compromised the security of the Diaspora developer preview. Avoiding such mistakes via better security practices and better choice of defaults will make applications more secure.

Diaspora is written against Ruby on Rails 3.0, a popular modern Web framework. Most Rails applications run as very long-lived processes within a specialized Web server such as Mongrel or Thin. Since Rails is not thread-safe, typically several processes will run in parallel on a machine, behind a threaded Web server such as Apache or nginx. These servers serve requests for static assets directly and proxy dynamic requests to the Rails instances.

Architecturally, Diaspora is designed as a federated Web application, with user accounts (seeds) collected into separately operated services (pods), in a manner similar to email accounts on separate mail servers. The primary way end users access their Diaspora accounts is through a Web interface. Pods communicate with each other using encrypted XML messages.

Unlike most Rails applications, Diaspora does not use a traditional database for persistence. Instead, it uses the MongoMapper ORM (object-relational mapping) to interface with MongoDB, which its makers describe as a “document-oriented database” that “bridges the gap between key/value stores and traditional relational databases.” MongoDB is an example of what are now popularly called NoSQL databases.

While Diaspora's architecture is somewhat exotic, the problems with the developer preview release stemmed from very prosaic sources.

Security in Ruby on Rails

Web application security is a very broad and deep topic, and is treated in detail in the official Rails security guide (<http://guides.rubyonrails.org/security.html>)



and the Open Web Application Security Project (OWASP) list of Web application vulnerabilities (http://www.owasp.org/index.php/Top_10_2007), which would have helped catch all of the issues discussed in this article. While Web application security might seem overwhelming, the errors discussed here are elementary and can serve as an object lesson for those building public-facing software.

A cursory analysis of the source code of the Diaspora prerelease revealed on the order of a half-dozen critical errors, affecting nearly every class in the system. There were three main genres, detailed below. All code samples pulled from Diaspora's source at launch (note: I have forked the Diaspora pub-

lic repository on GitHub and created a tag so that this code can be examined: https://github.com/patio11/diaspora/tree/diaspora_launch) were reported to the Diaspora team immediately upon discovery, and have been reported by the team as fixed.

Authentication ≠ Authorization: The User Cannot Be Trusted

The basic pattern in the following code was repeated several times in Diaspora's code base: security-sensitive actions on the server used parameters from the HTTP request to identify pieces of data they were to operate on, without checking that the logged-in user was actually authorized to view or operate on that data.

```
#photos_controller.rb
def destroy
  @album = Album.find_by_id
  params[:id]
  # No authorization check.
  @album.destroy
  flash[:notice] = "Album
  #{@album.name} deleted."
  respond_with :location =>
    albums_url
end
```

For example, if you were logged in to a Diaspora seed and knew the ID of any photo on the pod, changing the URL of any destroy action visible to include the ID of any other user's photo would let you delete that second photo. Rails makes such exploits very

easy, since URLs to actions are trivially easy to guess, and object IDs “leak” all over the place. Do not assume that an object ID is private.

Diaspora, of course, does attempt to check credentials. It uses Devise, a library that handles authentication, to verify that you get to the destroy action only if you are logged in. As shown in the previous code example, however, Devise does not handle authorization—checking to see that you are, in fact, permitted to do the action you are trying to do.

Impact. When Diaspora shipped, an attacker with a free account on any Diaspora node had, essentially, full access to any feature of the software vis-à-vis someone else’s account. That is quite a serious vulnerability, but it combines with other vulnerabilities in the system to allow attackers to commit more subtle and far-reaching attacks than merely deleting photos.

How to avoid this scenario. Check authorization prior to sensitive actions. The easiest way to do this (aside from using a library to handle it for you) is to take your notion of a logged-in user and access user-specific data only through that. For example, Devise gives all actions access to a `current_user` object, which is a stand-in for the currently logged-in user. If an action needs to access a photo, it should call `current_user.photos.find(params[:id])`. If a malicious user has subverted the `params` hash (which, since it comes directly from an HTTP request, must be considered “in the hands of the enemy”), that code will find no photo (because

of how associations scope to the `user_id`). This will instantly generate an ActiveRecord exception, stopping any potential nastiness before it starts.

Mass Assignment Will Ruin Your Day

We have learned that if we forget authorization, then a malicious user can do arbitrary bad things to people. In the example in Figure 1, since the user update method is insecure, an attacker could meddle with their profiles. But is that all we can do?

Unseasoned developers might assume that an update method can only update things on the Web form prior to it. For example, the form shown in Figure 2 is fairly benign, so one might think that all someone can do with this bug is deface the user’s profile name and email address:

This is dangerously wrong.

Rails by default uses something called mass update, where `update_attributes` and similar methods accept a hash as input and sequentially call all accessors for symbols in the hash. Objects will update both database columns (or their MongoDB analogs) and will call `parameter_name=` for any `:parameter_name` in the hash that has that method defined.

Impact. Let’s take a look at the `Person` object in the following code to see what mischief this lets an attacker do. Note that instead of updating the profile, `update_profile` updates the `Person`: Diaspora’s internal notion of the data associated with one

human being, as opposed to the login associated with one email address (the `User`). Calling something `update_profile` when it is really `update_person` is a good way to hide the security implications of such code from a reviewer. Developers should be careful to name things correctly.

This means that by changing a `Person`’s `owner_id`, one can reassign the `Person` from one account (`User`) to another, allowing one not only to deny arbitrary victims their use of the service, but also to take over their accounts. This allows the attacker to impersonate them, access their data at will, and so on. This works because the “one” method in MongoDB picks the first matching entry in the DB it can find, meaning that if two `Persons` have the same `owner_id`, the owning `User` will nondeterministically control one of them. This lets the attacker assign your `Person#owner_id` to be his `#owner_id`, which gives the attacker a 50-50 shot at gaining control of your account.

It gets worse: since the attacker can also reassign his own data’s `owner_id` to a nonsense string, this delinks his personal data from his account, which will ensure that his account is linked with the victim’s personal data.

It gets worse still. Note the `serialized_key` column. If you look deeper into the `User` class, that is its serialized public/private encryption key pair. Diaspora seeds use encryption when talking with each other so the prying eyes of Facebook can’t read users’ status updates. This is Diaspora’s core selling point. Unfortunately, an attacker can use the combination of unchecked authorization and mass update *silently to overwrite the user’s key pair*, replacing it with one the user generated. Since the attacker now knows the user’s private key, regardless of how well implemented Diaspora’s cryptography is, the attacker can read the user’s messages at will. This compromises Diaspora’s core value proposition to users: that their data will remain safe and in their control.

This is what kills most encryption systems in real life. You don’t have to beat encryption to beat the system; you just have to beat the weakest link in the chain around it. That almost certainly isn’t the encryption

Figure 1. Weaknesses in user update method.

```
#users_controller.rb
def update
  @user = User.find_by_id params[:id] # <-- No authorization check.
  prep_image_url(params[:user])

  @user.update_profile params[:user] # <-- Pass untrusted input to @user then...
  respond_with(@user, :location => root_url)
end

#user.rb
def update_profile(params)
  if self.person.update_attributes(params) # <-- insert input directly to DB.
    #omitted for clarity
  end
end
```

algorithm—it is probably some inadequacy in the larger system added by a developer in the mistaken belief that strong cryptography means strong security. Crypto is not soy sauce for security.

This attack is fairly elementary to execute. It can be done with a tool no more complicated than Firefox with Firebug installed: add an extra parameter to the form, switch the submit URL, and instantly gain control of any account you wish. Of particular note to open source software projects and other scenarios where the attacker can be assumed to have access to the source code, this vulnerability is very visible: the controller in charge of authorization and access to the user objects is a clear priority for attackers because of the expected gains from subverting it. A moderately skilled attacker could find this vulnerability and create a script to weaponize it in a matter of minutes.

How to avoid this scenario. This particular variation of the attack could be avoided by checking authorization, but that does not by itself prevent all related attacks. An attacker can create an arbitrary number of accounts, changing the `owner_id` on each to collide with a victim's legitimate user ID, and in doing so successfully delink the victim's data from his or her login. This amounts to a denial-of-service attack, since the victim loses the utility of the Diaspora service.

After authentication has been fixed, write access to sensitive data should be limited to the maximum extent practical. A suitable first step would be to disable mass assignment, which should always be turned off in a public-facing Rails app. The Rails team presumably keeps mass assignment on by default because it saves many lines of code and makes the 15-minute blog demo nicer, but it is a security hole in virtually all applications.

Luckily, this is trivial to address: Rails has a mechanism called `attr_accessible`, which makes only the listed model attributes available for mass assignment. Allowing only safe attributes to be mass-assigned (for example, data you would expect the end users to be allowed to update, such as their names rather than their keys)

prevents this class of attack. In addition, `attr_accessible` documents programmers' assumptions about security explicitly in their application code: as a whitelist, it is a known point of weakness in the model class, and it will be examined thoroughly by any security review process.

This is extraordinarily desirable, so it's a good idea for developers to make using `attr_accessible` compulsory. This is easy to do: simply call `ActiveRecord::Base.attr_accessible(nil)` in an initializer, and all Rails models will automatically have mass assignment disabled until they have it explicitly enabled by `attr_accessible`. Note that this may break the functionality of common Rails gems and plugins, because they sometimes rely on the default. This is one way in which security is a problem of the community.

An additional mitigation method, if your data store allows it, is to explicitly disallow writing to as much data as is feasible. There is almost certainly no legitimate reason for `owner_id` to be reassignable. ActiveRecord lets you do this with `attr_readonly`. MongoMapper does not currently support this feature, which is one danger of using bleeding-edge technologies for production systems.

NoSQL Doesn't Mean No SQL Injection

The new NoSQL databases have a few decades less experience getting exploited than the old relational databases we know and love, which means that countermeasures against well-understood attacks are still im-

mature. For example, the canonical attack against SQL databases is SQL injection: using the user-exposed interface of an application to craft arbitrary SQL code and execute it against the database.

```
def self.search(query)
  Person.all('$where' =>
    "function()
    { return this.diaspora_handle.match(/#{query}/i) ||
    this.profile.first_name.
    match(/#{query}/i) ||
    this.profile.last_name.
    match(/#{query}/i); }")
    #Permits code injection to
    MongoDB.
end
```

Impact. The previous code snippet allows code injection into MongoDB, effectively allowing an attacker full read access to the database, including to serialized encryption keys. Observe that because of the magic of string interpolation, the attacker can cause the string including the JavaScript to evaluate to virtually anything the attacker desires. For example, the attacker could inject a carefully constructed JavaScript string to cause the first regular expression to terminate without any results, then execute arbitrary code, then comment out the rest of the JavaScript.

We can get one bit of data about any particular person out of this find call—whether the person is in the result set or not. Since we can construct the result set at will, however, we can make that a very significant bit. JavaScript can take a string and convert it

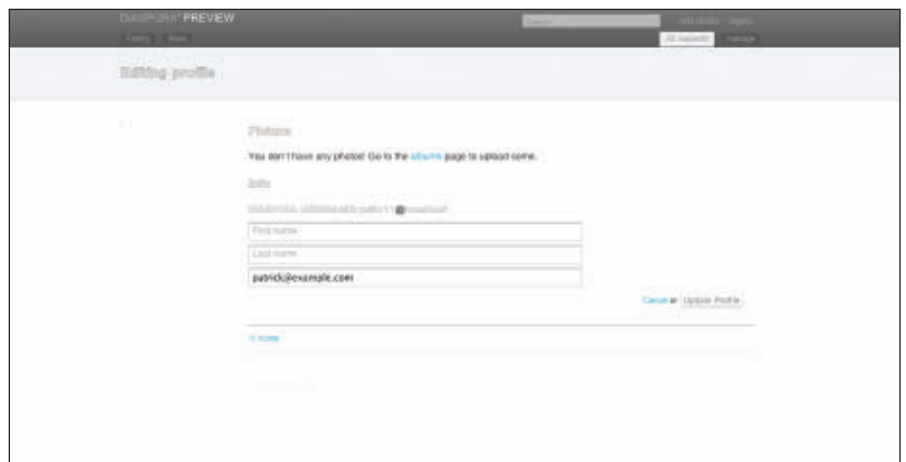


Figure 2. (Hardly) benign update method.

to a number. The code for this is left as an exercise for the reader. With that JavaScript, the attacker can run repeated find queries against the database to do a binary search for the serialized encryption key pair:

“Return Patrick if his serialized key is more than 2^{512} . OK, he isn’t in the result set? Alright, return Patrick if his key is more than 2^{256} . He is in the result set? Return him if his key is more than $2^{256} + 2^{255}$”

A key length of 1,024 bits might strike a developer as likely to be very secure. If we are allowed to do a binary search for the key, however, it will take only on the order of 1,000 requests to discover the key. A script executing searches through an HTTP client could trivially run through 1,000 accesses in a minute or two. Compromising the user’s key pair in this manner compromises all messages the user has ever sent or will ever send on Diaspora, and it would leave no trace of intrusion aside from an easily overlooked momentary spike in activity on the server. A more patient attacker could avoid leaving even that.

This is probably not the only vulnerability caused by code injection. It is very possible that an attacker could execute state-changing JavaScript through this interface, or join the Person document with other documents to read out anything desired from the database, such as user password hashes. Evaluating whether these attacks are feasible requires in-depth knowledge of the internal workings

of MongoDB and the Ruby wrappers for it. Typical application developers are insufficiently skilled to evaluate parts of the stack operating at those levels: it is essentially the same as asking them whether their SQL queries would allow buffer overruns if executed against a database compiled against an exotic architecture. Rather than attempting to answer this question, sensible developers should treat any injection attack as allowing a total system compromise.

How to avoid this scenario. Do not interpolate strings in queries sent to your database. Use the MongoDB equivalent of prepared statements. If your database solution does not have prepared statements, then it is insufficiently mature to be used in public-facing products.

Be Careful When Releasing Software to End Users

One could reasonably ask whether security flaws in a developer preview are an emergency or merely a footnote in the development history of a product. Owing to the circumstances of its creation, Diaspora never had the luxury of being both publicly available but not yet exploitable. As a highly anticipated project, Diaspora was guaranteed to (and did) have publicly accessible servers available within literally hours of the code being available.

People who set up servers should know enough to evaluate the security consequences of running them. This was not the case with the Diaspora

preview: there were publicly accessible Diaspora servers where any user could trivially compromise the account of another user. Moreover, even if one assumes the server operators understand what they are doing, their users and their users’ friends who are invited to join “The New Secure Facebook” are not capable of evaluating their security on Diaspora. They trust that, since it is on their browser and endorsed by a friend, it must be safe and secure. (This is essentially the same process through which they joined Facebook prior to evaluating the privacy consequences of that action.)

The most secure computer system is one that is in a locked room, surrounded by armed guards, and powered off. Unfortunately, that is not a feasible recommendation in the real world: software needs to be developed and used if it is to improve the lives of its users. Could Diaspora have simultaneously achieved a public-preview release without exposing end users to its security flaws? Yes. A sensible compromise would have been to release the code with the registration pages elided, forcing developers to add new users only via Rake tasks or the Rails console. That would preserve 100% of the ability of developers to work on the project and for news outlets to take screenshots—without allowing technically unsophisticated people to sign up on Diaspora servers.

The Diaspora community has taken some steps to reduce the harm of prematurely deploying the software, but they are insufficient. The team curates a list of public Diaspora seeds (<https://github.com/diaspora/diaspora/wiki/>), including a bold disclaimer that the software is insecure, but that sort of passive posture does not address the realities of how social software spreads: friends recommend it to friends, and warnings will be unseen or ignored in the face of social pressure to join new sites.

Could Rails Have Prevented These Issues?

Many partisans for languages or frameworks argue that “their” framework is more secure than alternatives and that some other frameworks are by nature insecure. Insecure code can

Figure 3. A mischievous ‘Person.’

```
#Person.rb
class Person
  #omitted for clarity
  key :url, String
  key :diaspora_handle, String, :unique => true
  key :serialized_key, String #Public/private key pair for encryption.

  key :owner_id, ObjectId #Extraordinarily security sensitive because...

  one :profile, :class_name => 'Profile'
  many :albums, :class_name => 'Album', :foreign_key => :person_id
  belongs_to :owner, :class_name => 'User' #... changing it reassigns account
  ownership!

end

#User.rb
one :person, :class_name => 'Person', :foreign_key => :owner_id
```

be written in any language: indeed, given that the question “Is this secure or not?” is algorithmically undecidable (it trivially reduces to the halting problem), one could probably go so far as to say it is flatly impossible to create any useful computer language that will always be secure.

That said, defaults and community matter. Rails embodies a spirit of convention over configuration, an example of what the team at 37signals (the original authors of Rails) describes as “opinionated software.” Rails conventions are pervasively optimized for programmer productivity and happiness. This sometimes trades off with security, as in the example of mass assignment being on by default.

Compromises exist on some of these opinions that would make Rails more secure without significantly impeding the development experience. For example, Rails could default to mass assignment being available in development environments, but disabled in production environments (which are, typically, the ones that are accessible by malicious users). There is precedent for this: for example, Rails prints stack traces (which may include sensitive information) only for local requests when in production mode, and gives less informative (and more secure) messages if errors are caused by nonlocal requests.

No amount of improving frameworks, however, will save programmers from mistakes such as forgetting to check authorization prior to destructive actions. This is where the community comes in: the open source community, practicing developers, and educators need to emphasize security as a process. There is no technological silver bullet that makes an application secure: it is made more secure as a result of detailed analysis leading to actions taken to resolve vulnerabilities.

This is often neglected in computer science education, as security is seen as either an afterthought or an implementation detail to be addressed at a later date. Universities often grade like industry: a program that operates successfully on almost all of the inputs scores almost all of the possible points. This mind-set, applied to security, has catastrophic results: the

A cursory analysis of the source code of the Diaspora prerelease revealed on the order of a half-dozen critical errors, affecting nearly every class in the system.

attacker has virtually infinite time to interact with the application, sometimes with its source code available, and see how it acts upon particular inputs. In a space of uncountable infinities of program states and possible inputs, the attacker may need to identify only one input for which the program fails to compromise the security of the system.

It would not matter if everything else in Diaspora were perfectly implemented; if the search functionality still allowed code injection, that alone would result in total failure of the project’s core goals.

Is Diaspora Secure after the Patches?

Security is a result of a process designed to produce it. While the Diaspora project has continued iterating on the software, and is being made available to select end users as of the publication of this article, it is impossible to say that the architecture and code are definitely secure. This is hardly unique to Diaspora: almost all public-facing software has vulnerabilities, despite huge amounts of resources dedicated to securing popular commercial and open source products.

This is not a reason to despair, though: every error fixed or avoided through improved code, improved practices, and security reviews offers incremental safety to the users of software and increases its utility. We can do better, and we should start doing so. 

Related articles on queue.acm.org

A Conversation with Jason Hoffman
<http://queue.acm.org/detail.cfm?id=1348587>

Browser Security: Lessons from Google Chrome

Charles Reis, Adam Barth, Carlos Pizano
<http://queue.acm.org/detail.cfm?id=1556050>

Cybercrime 2.0: When the Cloud Turns Dark

Niels Provos, Moheeb Abu Rajab, Panayiotis Mavrommatis
<http://queue.acm.org/detail.cfm?id=1517412>

Patrick McKenzie (patrick@bingocardcreator.com) is the founder of Kalzumeus, a small software business in Ogaki, Japan. His main products—Bingo Card Creator (<http://www.bingocardcreator.com>) and Appointment Reminder (<http://www.appointmentreminder.org>)—are both written in Ruby.

© 2011 ACM 0001-0782/11/05 \$10.00

DOI:10.1145/1941487.1941506

The brain's electrical signals enable people without muscle control to physically interact with the world.

BY DENNIS J. MCFARLAND AND JONATHAN R. WOLPAW

Brain-Computer Interfaces for Communication and Control

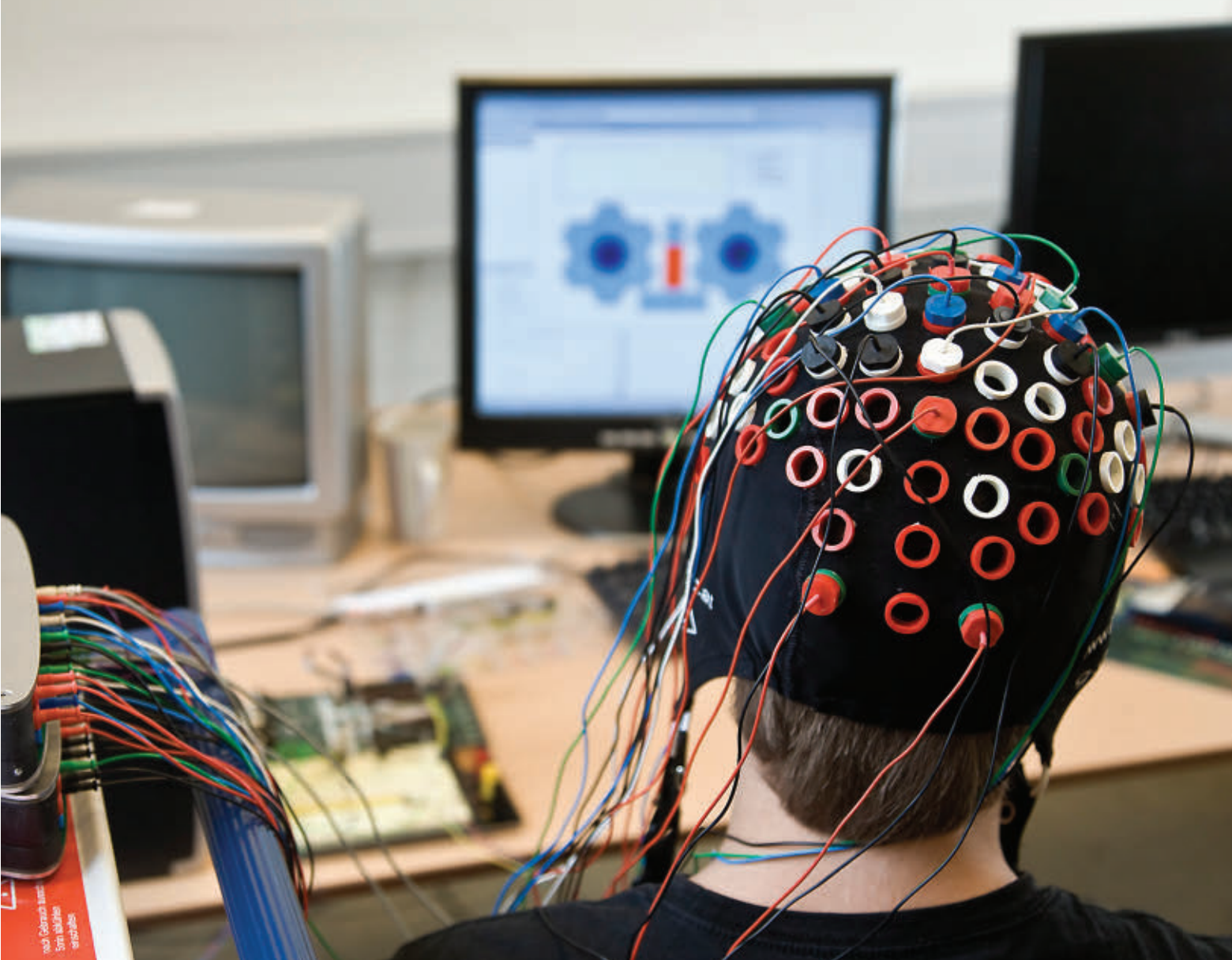
BRAIN ACTIVITY PRODUCES electrical signals detectable on the scalp, on the cortical surface, or within the brain. Brain-computer interfaces (BCIs) translate these signals into outputs that allow users to communicate without participation of peripheral nerves and muscles³⁶ (see Figure 1). Because they do not depend on neuromuscular control, BCIs provide options for communication and control for people with devastating neuromuscular disorders (such as amyotrophic lateral sclerosis, or ALS, brainstem stroke, cerebral palsy, and spinal cord injury). The central purpose of BCI research and development is to enable these users to convey their wishes to caregivers, use word-processing programs and other software, and even control a robotic arm or neuroprosthesis. Speculation has suggested that BCIs could be useful even to people with lesser, or no, motor impairment.

It has been known since the pioneering work of Hans Berger more than 80 years ago that the brain's electrical activity can be recorded noninvasively through electrodes on the surface of the scalp.²³ Berger observed that a rhythm of about 10Hz was prominent on the posterior scalp and reactive to light. He called it the alpha rhythm. This and other observations showed the electroencephalogram (EEG) could serve as an index of the gross state of the brain. Despite Berger's careful work many scientists were initially skeptical, with some suggesting that the EEG might represent some sort of artifact. However, subsequent work demonstrated conclusively that the EEG is indeed produced by brain activity.²³

Electrodes on the surface of the scalp are at some distance from brain tissue, separated from it by the coverings of the brain, skull, subcutaneous tissue, and scalp. As a result, the signal is considerably degraded, and only the synchronized activity of large numbers of neural elements can be detected, limiting the resolution with which brain activity can be monitored. Moreover, scalp electrodes pick up activity from sources other than the brain, including environmental noise (such as 50Hz or 60Hz activity from power lines) and biological noise (such as activity from the heart, skeletal muscles, and eyes). Nevertheless, since the time of Berger, many studies have used the EEG to gain insight into brain function, with many of them using averaging to separate EEG from superimposed electrical noise.

» key insights

- **Brain-computer interfaces provide a new communication-and-control option for individuals for whom conventional methods are ineffective.**
- **Current BCI technology is slow, benefiting only those with the most severe disabilities.**
- **Research may greatly expand the number of people who would benefit from the technology.**



BCIs are a direct communication pathway between the brain and external devices. EEG measurements at the Danish Master's program in Medicine & Technology; <http://www.medicin-ing.dk/kandidat/en>.

EEG research reflects two major paradigms: evoked potentials and oscillatory features. Evoked potentials are transient waveforms, or brief perturbations in the ongoing activity, that are phase-locked to an event (such as a visual stimulus). They are typically analyzed by averaging many similar events in the time-domain. Although oscillatory features in an EEG may occur in response to specific events, they are usually not phase-locked and typically studied through spectral analysis. Historically, most EEG studies have examined phase-locked evoked potentials. Both these major paradigms have been applied in BCIs.³⁶

The term “brain-computer interface” can be traced to Jacques Vidal of the University of California, Los Angeles who devised a BCI system in the

1970s based on visual evoked-potentials.³⁴ His users viewed a diamond-shape red checkerboard illuminated with a xenon flash. By attending to different corners of the flashing checkerboard, they could generate right, up, left, and down commands, enabling them to move through a maze presented on a graphics terminal. An IBM360 mainframe digitized the data, and an XDS Sigma 7 computer controlled the experimental events. Users first provided data to train a stepwise linear discriminant function, then navigated the maze online in real time. Thus, Vidal³⁴ used signal-processing techniques to realize real-time analysis of the EEG with minimal averaging. The waveforms showed by Vidal³⁴ suggested his BCI used EEG activity in the timeframe of the N100-P200 components, with the N and P indicating

negative and positive peaks, and the numbers indicating the approximate latency in msec.

Vidal's achievement was an interesting demonstration of proof of principle. In the early 1970s, it was far from practical, given that it depended on a time-shared system with limited processing capacity. Vidal³⁴ also included in his system online removal of ocular artifacts to prevent them from being used for control. A decade earlier, Edmond Dewan⁶ of the Air Force Cambridge Research Lab, Bedford MA, instructed users to explicitly use eye movements to modulate their brain waves, showing that subjects could learn to transmit Morse code messages using EEG activity associated with eye movement.

The fact that both Vidal's and Dewan's BCIs depended on eye move-

ment made them somewhat less interesting from a scientific or clinical point of view, since they required actual muscle control or eye movement, simply using EEG to reflect the resulting gaze direction.

Varieties of BCI Signals

Farwell and Donchin⁷ reported the first

use of a P300-based spelling device (see Figure 2b) in which a positive potential around 300msec after an event significant to the subject is considered a “cognitive” potential since it is generated in tasks where the subject discriminates among stimuli. Farwell’s and Donchin’s⁷ users viewed a 6×6 matrix of the letters of the alpha-

bet and several other symbols, focusing attention on the desired selection, as the rows and columns of the matrix were repeatedly flashed to elicit visual evoked potentials. Farwell and Donchin⁷ found their users were able to spell the word “brain” through the P300 spelling device; in addition, they did an offline comparison of detection algorithms, finding the stepwise linear discriminant analysis was generally best. The fact that the P300 potential reflects attention rather than simply gaze-direction implied this BCI did not depend on muscle, or eye-movement, control, thus representing a significant advance. Several groups have since further developed this BCI method.¹³

Wolpaw et al.³⁸ reported the first use of sensorimotor rhythms (SMRs) for cursor control (see Figure 2a), or EEG rhythms that change with movement or imagination of movement and are spontaneous in the sense they do not require specific stimuli to occur. People learned to vary their SMRs to move a cursor to hit one of two targets on the top or bottom edge of a video screen. Cursor movement was controlled by SMR amplitude (measured by spectral analysis). A distinctive feature of this task is that it required users to rapidly switch between two states to select a particular target. The rapid bidirectional nature of the Wolpaw et al.³⁸ paradigm made it distinct from prior studies that produced long-term uni-

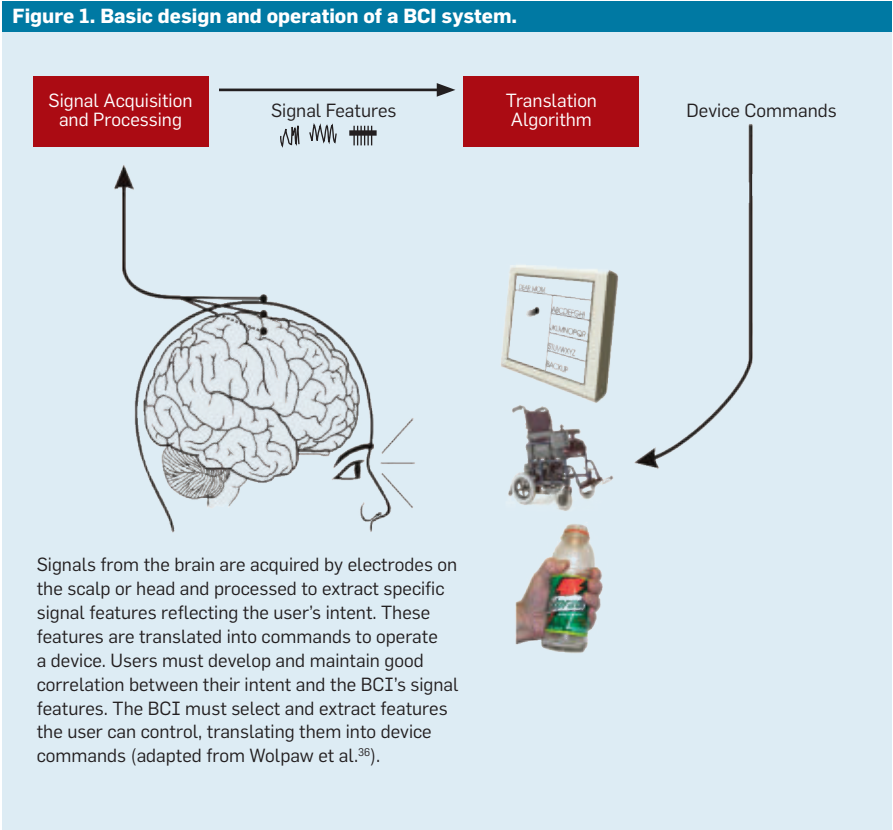
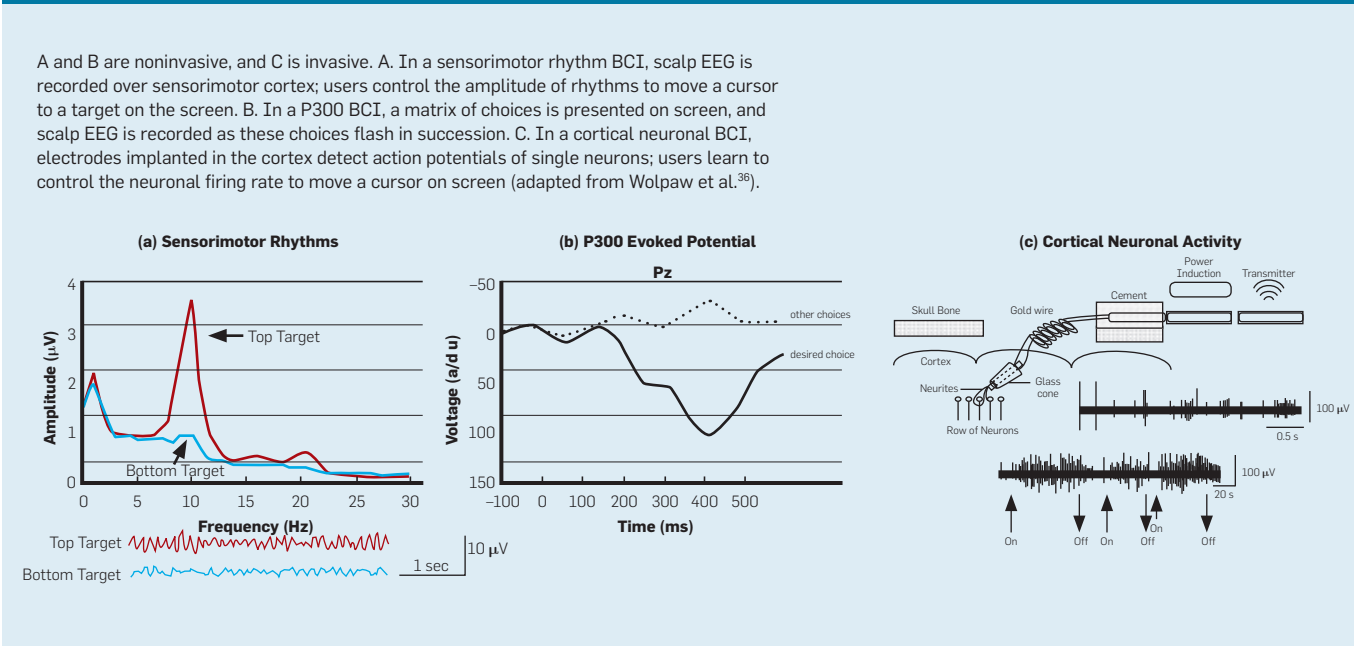


Figure 2. Current human BCI systems.



directional changes in brain rhythms; for example, users were required to maintain an increase in the size of an EEG rhythm for minutes at a time. In a series of subsequent studies, this group showed that the signals controlling the cursor were actual EEG activity and that covert muscle activity did not mediate this EEG control.^{18,31}

These initial SMR results were subsequently replicated by others^{21,24} and extended to multidimensional control.³⁷ These P300 and SMR BCI studies together showed that noninvasive EEG recording of brain signals can serve as the basis for communication-and-control devices.

A number of laboratories have explored the possibility of developing BCIs using single-neuron activity detected by microelectrodes implanted in the cortex^{12,30} (see Figure 2c). Much of the related research has been done in non-human primates, though trials have also been done with humans.¹² Other studies have shown that recordings of electrocorticographic (ECoG) activity from the surface of the brain can also provide signals for a BCI¹⁵; to date they indicate that invasive recording methods can also serve as the basis for BCIs. Meanwhile, important issues concerning their suitability for long-term human use have yet to be resolved.

Earlier studies demonstrating operant conditioning of single units in the motor cortex of primates,⁹ hippocampal theta rhythm of dogs,² and sensorimotor rhythm in humans²⁹ showed brain activity could be trained with operant techniques. However, these studies were not demonstrations of BCI systems for communication and control since they required subjects to increase brain activity for periods of many minutes, showing that brain activity could be tonically altered in a single direction through training. However, communication-and-control devices require that users be able to select from at least two distinct alternatives; that is, there must be at least one bit of information per selection. Effective communication-and-control devices require users to rapidly switch between multiple alternatives.

In addition to electrophysiological measures, researchers have also demonstrated the feasibility of magneto-

encephalography (MEG),²⁰ functional magnetic resonance imaging (fMRI),²⁸ and near-infrared systems (fNIR).⁴ Current technology for recording MEG and fMRI is both expensive and bulky, making it unlikely for practical applications in the near term; fNIR is potentially cheaper and more compact. However, both fMRI and fNIR are based on changes in cerebral blood flow, an inherently slow response.¹¹ Electrophysiological features represent the most practical signals for BCI applications today.

System Design

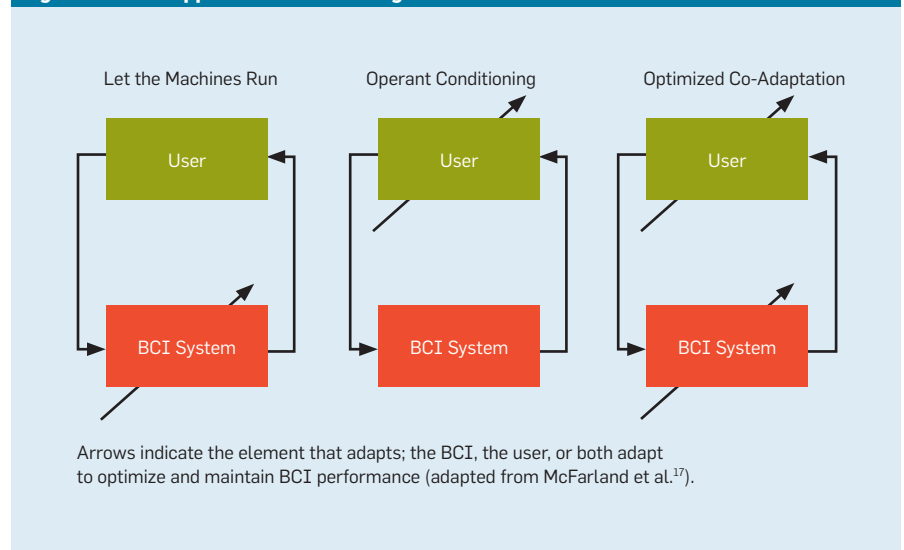
Communication-and-control applications are interactive processes requiring users observe the results of their effort to maintain good performance and correct mistakes. For this reason, BCIs must run in real time and provide real-time feedback to users. While many early BCI studies satisfied this requirement,^{24,38} later studies were often based on offline analyses of pre-recorded data¹; for example, the Lotte et al.¹⁶ review of studies evaluating BCI signal-classification algorithms found most used offline analyses. Indeed, the current popularity of BCI research is probably due in part to the ease of offline analyses are performed on publicly available data sets. While such offline studies may help guide actual online BCI investigations, there is no guarantee that offline results will generalize to online performance. Users' brain signals are often affected by BCI outputs that are in turn determined

by the algorithm the BCI is using. It is thus not possible to predict results precisely from offline analyses that cannot account for these effects.

Blankertz et al.³ identified several trends in the results of a BCI data-classification competition. Most winning entries used linear classifiers, the most popular being Fisher's discriminant and linear support vector machines (SVMs). The winning entries for data sets with multichannel oscillatory features often used common spatial patterns. In their review of the literature on BCI classifiers, Lotte et al.¹⁶ concluded that SVMs are particularly efficient, attributing the efficiency to their regularization property and immunity to the curse of dimensionality. They also concluded that combinations of classifiers seem efficient, noting a lack of comparison of classifiers within the same study using otherwise identical parameters.

Muller and Blankertz²¹ advocated a machine-learning approach to BCIs in which a statistical analysis of a calibration measurement is used to train the system. The goal is to develop a "zero-training" method providing effective performance from the first session, contrasting it with one based on training users to control specific features of brain signals.³⁸ A system that can be used without extensive training is appealing since it requires less initial effort on the part of both the BCI user and the system operator. Operation of such a system is based on the as-yet uncertain premise that users can

Figure 3. Three approaches to BCI design.



repeatedly and reliably maintain the specified correlations between brain signals and intent. Figure 3 outlines three different conceptualizations of where adaptation might take place to establish and maintain good BCI performance: In the first, the BCI adapts to the user; in the second, the user adapts to the BCI; and, in the third, user and system adapt to each other.

A number of BCI systems are designed to detect user performance of specific cognitive tasks. Curran et al.³

suggested that cognitive tasks (such as navigation and auditory imagery) might be more useful in driving a BCI than motor imagery. However, sensorimotor rhythm-based BCIs may provide several advantages over systems that depend on complex cognitive operations; for example, the structures involved in auditory imagery are also likely to be driven by auditory sensory input. Wolpaw and McFarland³⁷ reported that with extended practice users report motor imagery is no lon-

ger necessary to operate a sensorimotor rhythm-based BCI. As is typical of many simple motor tasks, performance becomes automatized through extended practice. Automatized performance may be less likely to interfere with mental operations users might wish to engage in concurrent with their BCI use; for example, composing a manuscript is much easier if the writer does not need to think extensively about each individual keystroke.

As noted, EEG recording may be contaminated by non-brain activity (such as line noise and muscle activity); see Fatourechi et al.⁸ for a review. Activity recorded from the scalp represents the superposition of many signals, some originating in the brain, some elsewhere. These signals include potentials generated by retinal dipoles, or eye movement and blinks, and facial muscles. It is noteworthy that mental effort is often associated with changes in eye-blink rate and muscle activity.³⁵ BCI users might generate these artifacts without being aware of what they are doing simply by making facial expressions associated with effort.

Facial muscles can generate signals with energy in the spectral bands used as features in an SMR-based BCI¹⁸ Muscle activity can also modulate SMR activity; for example, users can move their right hands in order to desynchronize the mu rhythm over the left hemisphere. This sort of mediation of the EEG through peripheral muscle movements was a concern in the early days of BCI development. As noted earlier, Dewan⁶ trained users to send Morse code messages using occipital alpha rhythms modulated by voluntary movements of eye muscles. For this reason, Vaughan et al.³³ recorded EMG from 10 distal limb muscles, while BCI users used central mu or beta rhythms to move a cursor to targets on a video screen. EMG activity was very low in these well-trained users. Most important, the correlations between target position and EEG activity could not be accounted for through EMG activity. Similar studies have been done with BCI users moving a cursor in two dimensions,³⁷ showing that SMR modulation does not require actual movements or muscle activity.

Figure 4. BCI2000 design consists of four modules: operator, source, signal processing, and application.

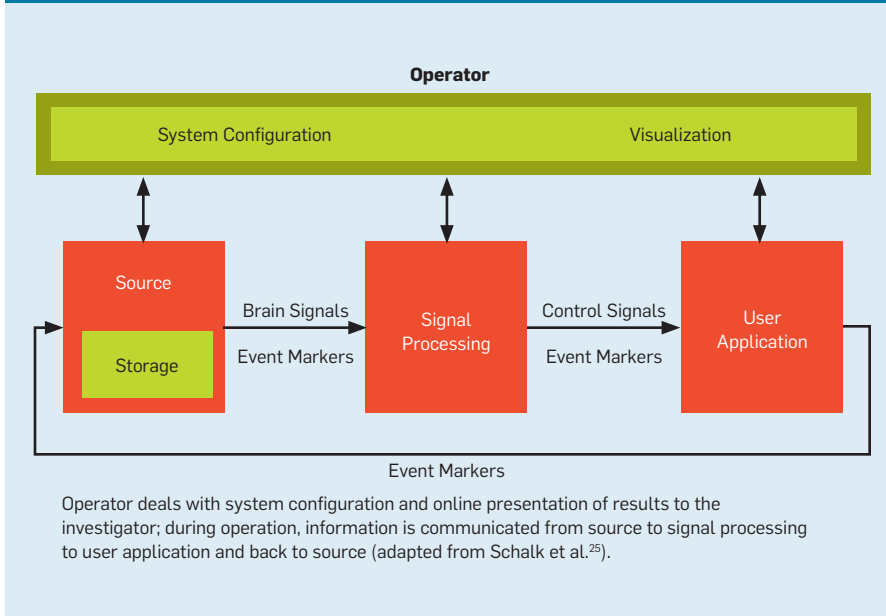


Figure 5. Hardware in the Wadsworth Center's home BCI system, including 16-channel electrode cap for signal recording, solid-state amplifier, laptop, and additional monitor as user display.

Applications

Several recent BCI spelling systems are based on different EEG signals, including the mu rhythm^{22,26} and the P300.³¹ The Mu rhythm systems made use of machine-learning paradigms that minimized training, with users of both mu-based systems reportedly averaging between 2.3–7 characters/minute²² and 2.85–3.38 characters/minute.²⁶ The P300 system averaged 3.66 selections/minute.³¹ Townsend et al.²⁴ noted the reported rate depends on how the figure is computed, but study authors do not always provide details. Omitting time between trials increased Townsend et al.³¹ results from 3.66 to 5.92 characters/second. In any case, these systems perform within a similar general range. At current BCI character rates, only users with limited options could benefit.

BCI systems have also been developed for control applications; for example, several groups have shown that human subjects can use their EEG activity to drive a simulated wheelchair.^{10,14} Bell et al.¹ showed the P300 could be used to select among complex commands to a partially autonomous humanoid robot; for a review of the use of BCI for robotic and prosthetic devices see McFarland and Wolpaw.¹⁹

Several commercial concerns recently produced inexpensive devices purported to measure EEG. Both Emotiv and Neurosky developed products with a limited number of electrodes that do not use conventional gel-based recording technology²⁷ and are intended to provide input for video games. Not clear is the extent to which they use actual EEG activity, as opposed to scalp-muscle activity or other non-brain signals. Given the well-established prominence of EMG activity in activity recorded from the head, it seems likely that such signals account for much of the control these devices provide.²⁷

Conclusion

In a review of the use of BCI technology for robotic and prosthetic devices, McFarland and Wolpaw¹⁹ concluded that the major problem facing BCI applications is how to provide fast, accurate, reliable control signals, as well as other uses of BCIs. Current BCI systems that operate using actual brain activity can provide communication-and-control



Sensorimotor rhythm-based BCIs may provide several advantages over systems that depend on complex cognitive operations.



options of practical value mainly for people severely limited in their motor skills and thus have few other options. Widespread use of BCI technology by individuals with little or no disability is unlikely in the short-term and would require much greater speed and accuracy than has so far been demonstrated in the scientific literature.

Noninvasive and invasive methods would both benefit from improved recording methods. Current invasive methods do not deal adequately with the need for long-term performance stability. The brain's complex reaction to an implant is still imperfectly understood and might impair long-term performance. Noninvasive EEG electrodes require some level of skill in the person placing them, as well as in periodic maintenance to ensure sufficiently good contact with the skin; more convenient and stable electrodes are under development. Improved methods for extracting key EEG features and translating them into device control, as well as user training, would also help improve BCI performance.

Recent developments in computer hardware provide compact portable systems that are extremely powerful. Use of digital electronics has also led to improved size and performance of EEG amplifiers. Thus it is no longer necessary to use a large time-shared mainframe, as it was with Vidal³⁴; standard laptops easily handle the vast majority of real-time BCI protocols. Signal-processing and machine-learning algorithms have also been improved. Coupled with discovery of new EEG features for BCI use and development of new paradigms for user training, such improvements are gradually increasing the speed and reliability of BCI communication and control, developments facilitated by the BCI2000 software platform.²⁵

BCI2000 is a general-purpose research-and-development system incorporating any brain signal, signal-processing method, output device, and operating protocol. BCI2000 consists of a general standard for creating interchangeable modules designed according to object-oriented principles (see Figure 4), including a source module for signal acquisition, signal-processing module, and user-application module. Configuration

and coordination of these modules is accomplished through a fourth operator module; several source modules, signal-processing modules, and user applications have been created for the BCI2000 standard (see <http://www.bci2000.org/BCI2000/Home.html>).

The Wadsworth Center recently began developing a system for home use by individuals with severe motor impairments.³² Its basic hardware (see Figure 5) consists of a laptop computer with 16-channel EEG acquisition, a second screen placed in front of the user, and an electrode cap; software is provided by the BCI2000 general-purpose system.²⁵ The initial users had late-stage ALS, with little or no voluntary movement, and found conventional assistive communication devices inadequate for their needs. The P300-based matrix speller is used for these applications due to its relatively high throughput for spelling and simplicity of use. A 49-year-old scientist with ALS has used this BCI system on a daily basis for approximately three years, finding it superior to his eye-gaze system (a letter-selection device based on eye-gaze direction) and using it from four to six hours per day for email and other communication purposes.³²

How far BCI technology will go and how useful it will be depend on future research developments. However, it is apparent that BCIs can serve the basic communication needs of people whose severe motor disabilities prevent them from using conventional augmentive communications devices, all of which require muscle control.

Acknowledgments

This work was supported in part by grants from the National Institutes of Health HD30146 (NCMRR, NICHD) and EB00856 (NIBIB & NINDS) and the James S. McDonnell Foundation. We thank Chad Boulay and Peter Brunner for their comments on the manuscript. C

References

1. Bell, C.J., Shenoy, P., Chalodhorn, R., and Rao, R.P.N. Control of a humanoid robot by a noninvasive brain-computer interface in humans. *Journal of Neural Engineering* 5, 2 (June 2008), 214–220.
2. Black, A.H., Young, G.A., and Batenchuk, C. Avoidance training of hippocampal theta waves in flaxedilized dogs and its relation to skeletal movement. *Journal of Comparative and Physiological Psychology* 70, 1

- (Jan. 1970), 15–24.
3. Blankertz, B., Müller, K.-R., Krusienski, D.J., Schalk, G., Wolpaw, J.R., Schlogl, A., Pfurtscheller, G., Millan, J., Schröder, M., and Birbaumer, N. The BCI competition III: Validating alternative approaches to actual BCI problems. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 14, 2 (June 2006), 153–159.
4. Coyle, S.M., Ward, T.E., and Markham, C.M. Brain-computer interface using a simplified functional near-infrared spectroscopy system. *Journal of Neural Engineering* 4, 3 (Sept. 2007), 219–226.
5. Curran, E., Sykacek, P., Stokes, M., Roberts, S.J., Penny, W., Johnsrude, I., and Owen, A. Cognitive tasks for driving a brain-computer interface: A pilot study. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 12, 1 (Mar. 2003), 48–54.
6. Dewan, E.M. Occipital alpha rhythm eye position and lens accommodation. *Nature* 214, 5092 (June 3, 1967), 975–977.
7. Farwell, L.A. and Donchin, E. Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalography and Clinical Neurophysiology* 70, 6 (Dec. 1988), 510–523.
8. Fatourech, M., Bashashati, A., Ward, R.K., and Birch, G.E. EMG and EOG artifacts in brain-computer interface systems: A survey. *Clinical Neurophysiology* 118, 3 (Mar. 2007), 480–494.
9. Fetz, E.E. Operant conditioning of cortical unit activity. *Science* 163, 870 (Feb. 28, 1969), 955–958.
10. Galan, F., Nuttin, M., Lew, E., Ferrez, P.W., Vanacker, G., Phillips, J., and Millan, J.D.R. A brain-actuated wheelchair: Asynchronous and noninvasive brain-computer interfaces for continuous control of robots. *Clinical Neurophysiology* 119, 9 (Sept. 2008), 2159–2169.
11. He, B. and Liu, Z. Multimodal functional neuroimaging: Integrating functional MRI and EEG/MEG. *IEEE Reviews in Biomedical Engineering* 1 (Nov. 2008), 23–40.
12. Hochberg, L.R., Serruya, M.D., Friehs, G.M., Mukand, J.A., Saleh, M., Caplan, A.H., Branner, A., Penn, D.R.D., and Donoghue, J.P. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature* 442, 7099 (July 13, 2006), 164–171.
13. Krusienski, D.J., Sellers, E.W., McFarland, D.J., Vaughan, T.M., and Wolpaw, J.R. Toward enhanced P300 speller performance. *Journal of Neuroscience Methods* 167, 1 (Jan. 15, 2008), 15–21.
14. Leeb, R., Friedman, D., Müller-Putz, G.R., Scherer, R., Slater, M., and Pfurtscheller, G. Self-paced (asynchronous) BCI control of a wheelchair in virtual environments: A case study with a tetraplegic. *Computational Intelligence and Neuroscience* 79642 (2007).
15. Leuthardt, E.C., Schalk, G., Wolpaw, J.R., Ojemann, J.G., and Moran, D.W. A brain-computer interface using electrocorticographic signals in humans. *Journal of Neural Engineering* 1, 2 (June 2004), 63–71.
16. Lotte, F., Congedo, M., Lecuyer, A., Lamarche, F., and Arnaldi, B. A review of classification algorithms for EEG-based brain-computer interfaces. *Journal of Neural Engineering* 4, 2 (June 2007), 1–13.
17. McFarland, D.J., Krusienski, D.J., and Wolpaw, J.R. Brain-computer interface signal processing at the Wadsworth Center: Mu and sensorimotor beta rhythms. *Progress in Brain Research* 159 (2006), 411–419.
18. McFarland, D.J., Sarnacki, W.A., Vaughan, T.M., and Wolpaw, J.R. Brain-computer interface (BCI) operation: Signal and noise during early training sessions. *Clinical Neurophysiology* 116 (2005), 56–62.
19. McFarland, D.J. and Wolpaw, J.R. Brain-computer interface operation of robotic and prosthetic devices. *Computer* 41, 10 (Oct. 2008), 48–52.
20. Mellinger, J., Schalk, G., Braun, C., Preissl, H., Rosenstiel, W., Birbaumer, N., and Kubler, A. An MEG-based brain-computer interface (BCI). *Neuroimage* 36, 3 (July 1, 2007), 581–593.
21. Müller, K.-R., and Blankertz, B. Towards noninvasive brain-computer interfaces. *IEEE Signal Processing Magazine* 23, 1 (Sept. 2006), 125–128.
22. Müller, K.-R., Tangermann, M., Dornhege, G., Krauledat, M., Curio, G.T., and Blankertz, B. Machine learning for real-time single-trial EEG-analysis: From brain-computer interfacing to mental-state monitoring. *Journal of Neuroscience Methods* 167, 1 (Jan. 15, 2008), 82–90.
23. Neidermeyer, E. Historical aspects. In *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields, Fifth Edition*, E. Neidermeyer and F. Lopes da Silva, Eds. Lippincott Williams and Wilkins, Philadelphia, 2005, 1–15.
24. Pfurtscheller, G., Flotzinger, D., and Kalcher, J. Brain-computer interface: A new communication device for handicapped persons. *Journal of Microcomputer Applications* 16, 3 (July 1993), 293–299.
25. Schalk, G., McFarland, D.J., Hinterberger, T., Birbaumer, N., and Wolpaw, J.R. BCI2000: A general-purpose brain-computer interface (BCI) system. *IEEE Transactions on Biomedical Engineering* 51 (2004), 1034–1043.
26. Scherer, R., Müller, G.R., Neuper, C., Graimann, B., Pfurtscheller, G. An asynchronously controlled EEG-based virtual keyboard: Improvement of the spelling rate. *IEEE Transactions on Biomedical Engineering* 51, 6 (June 2004), 979–984.
27. Singer, E. Brain games. *Technology Review* 111, 4 (July/Aug. 2008), 82–84.
28. Sitaram, R., Caria, A., Veit, R., Gaber, T., Rota, G., Kuebler, A., and Birbaumer, N. fMRI brain-computer interface: A tool for neuroscientific research and treatment. *Computational Intelligence and Neuroscience* (2007).
29. Sterman, M.B., MacDonald, L.R., and Stone, R.K. Biofeedback training of sensorimotor EEG in man and its effect on epilepsy. *Epilepsia* 15, 3 (Sept. 1974), 395–416.
30. Taylor D.A., Tillery S., and Schwartz, A.B. Direct cortical control of 3D neuroprosthetic devices. *Science* 296, 5574 (June 7, 2002), 1829–1832.
31. Townsend, G., LaPallo, B.K., Boulay, C.B., Krusienski, D.J., Frye, G.E., Hauser, C.K., Schwartz, N.E., Vaughan, T.M., Wolpaw, J.R., and Sellers, E.W. A novel P300-based brain-computer interface stimulus presentation paradigm: Moving beyond rows and columns. *Clinical Neurophysiology* 121, 7 (July 2010), 1109–1120.
32. Vaughan, T.M., McFarland, D.J., Schalk, G., Sarnacki, W.A., Krusienski, D.J., Sellers, E.W., and Wolpaw, J.R. The Wadsworth BCI research and development program: At home with BCI. *IEEE Transactions on Rehabilitation Engineering* 14, 2 (June 2006), 229–233.
33. Vaughan, T.M., Miner, L.A., McFarland, D.J., and Wolpaw, J.R. EEG-based communication: Analysis of concurrent EMG activity. *Electroencephalography and Clinical Neurophysiology* 107, 6 (Dec. 1998), 428–433.
34. Vidal, J.J. Real-time detection of brain events in EEG. *Proceedings of the IEEE* 65, 5 (May 1977), 633–641.
35. Whitham, E.M., Lewis, T., Pope, K.J., Fitzbibbon, S.P., Clark, C.R., Loveless, S., DeLosAngeles, D., Wallace, A.K., Broberg, M., and Willoughby, J.O. Thinking activates EMG in scalp electrical recordings. *Clinical Neurophysiology* 119, 5 (May 2008), 1166–1175.
36. Wolpaw, J.R., Birbaumer, N., McFarland, D.J., Pfurtscheller, G., and Vaughan, T.M. Brain-computer interfaces for communication and control. *Clinical Neurophysiology* 113, 6 (June 2002), 767–791.
37. Wolpaw, J.R. and McFarland, D.J. Control of a two-dimensional movement signal by a noninvasive brain-computer interface. *Proceedings of the National Academy of Science* 101, 51 (Dec. 21, 2004), 17849–17854.
38. Wolpaw, J.R., McFarland, D.J., Neat, G.W., and Forneris, C.A. An EEG-based brain-computer interface for cursor control. *Electroencephalography and Clinical Neurophysiology* 78, 3 (Mar. 1991), 252–259.

Dennis J. McFarland (mcfarland@wadsworth.org) is a research scientist in the Laboratory of Neural Injury and Repair at the Wadsworth Center of the New York State Department of Health, Albany, NY.

Jonathan R. Wolpaw (wolpaw@wadsworth.org) is a research physician in the Laboratory of Neural Injury and Repair in the Wadsworth Center of the New York State Department of Health, Albany, NY.

Energy efficiency is the new fundamental limiter of processor performance, way beyond numbers of processors.

BY SHEKHAR BORKAR AND ANDREW A. CHIEN

The Future of Microprocessors

MICROPROCESSORS—SINGLE-CHIP COMPUTERS—are the building blocks of the information world. Their performance has grown 1,000-fold over the past 20 years, driven by transistor speed and energy scaling, as well as by microarchitecture advances that exploited the transistor density gains from Moore's Law. In the

next two decades, diminishing transistor-speed scaling and practical energy limits create new challenges for continued performance scaling. As a result, the frequency of operations will increase slowly, with energy the key limiter of performance, forcing designs to use large-scale parallelism, heterogeneous cores, and accelerators to achieve performance and energy efficiency. Software-hardware partnership to achieve efficient data orchestration is increasingly critical in the drive toward energy-proportional computing.

Our aim here is to reflect and project the macro trends shaping the future of microprocessors and sketch in broad strokes where processor design is going. We enumerate key research challenges and suggest promising research directions. Since dramatic changes are coming, we also seek to inspire the research community to in-

vent new ideas and solutions address how to sustain computing's exponential improvement.

Microprocessors (see Figure 1) were invented in 1971,²⁸ but it's difficult today to believe any of the early inventors could have conceived their extraordinary evolution in structure and use over the past 40 years. Microprocessors today not only involve complex micro-

» key insights

- **Moore's Law continues but demands radical changes in architecture and software.**
- **Architectures will go beyond homogeneous parallelism, embrace heterogeneity, and exploit the bounty of transistors to incorporate application-customized hardware.**
- **Software must increase parallelism and exploit heterogeneous and application-customized hardware to deliver performance growth.**

architectures and multiple execution engines (cores) but have grown to include all sorts of additional functions, including floating-point units, caches, memory controllers, and media-processing engines. However, the defining characteristics of a microprocessor remain—a single semiconductor chip embodying the primary computation (data transformation) engine in a computing system.

Because our own greatest access and insight involves Intel designs and data, our graphs and estimates draw heavily on them. In some cases, they may not be representative of the entire industry but certainly represent a large fraction. Such a forthright view, solidly grounded, best supports our goals for this article.

20 Years of Exponential Performance Gains

For the past 20 years, rapid growth in microprocessor performance has been enabled by three key technology drivers—transistor-speed scaling, core microarchitecture techniques, and cache memories—discussed in turn in the following sections:

Transistor-speed scaling. The MOS transistor has been the workhorse for decades, scaling in performance by nearly five orders of magnitude and providing the foundation for today's unprecedented compute performance. The basic recipe for technology scaling was laid down by Robert N. Dennard of IBM¹⁷ in the early 1970s and followed for the past three decades. The scaling recipe calls for reducing transistor

dimensions by 30% every generation (two years) and keeping electric fields constant everywhere in the transistor to maintain reliability. This might sound simple but is increasingly difficult to continue for reasons discussed later. Classical transistor scaling provided three major benefits that made possible rapid growth in compute performance.

First, the transistor dimensions are scaled by 30% (0.7x), their area shrinks 50%, doubling the transistor density every technology generation—the fundamental reason behind Moore's Law. Second, as the transistor is scaled, its performance increases by about 40% (0.7x delay reduction, or 1.4x frequency increase), providing higher system performance. Third, to keep the electric field constant, supply voltage is reduced by 30%, reducing energy by 65%, or power (at 1.4x frequency) by 50% (active power = CV^2f). Putting it all together, in every technology generation transistor integration doubles, circuits are 40% faster, and system power consumption (with twice as many transistors) stays the same. This serendipitous scaling (almost too good to be true) enabled three-orders-of-magnitude increase in microprocessor performance over the past 20 years. Chip architects exploited transistor density to create complex architectures and transistor speed to increase frequency, achieving it all within a reasonable power and energy envelope.

Core microarchitecture techniques. Advanced microarchitectures have deployed the abundance of transistor-integration capacity, employing a dizzying array of techniques, including pipelining, branch prediction, out-of-order execution, and speculation, to deliver ever-increasing performance. Figure 2 outlines advances in microarchitecture, showing increases in die area and performance and energy efficiency (performance/watt), all normalized in the same process technology. It uses characteristics of Intel microprocessors (such as 386, 486, Pentium, Pentium Pro, and Pentium 4), with performance measured by benchmark SpecInt (92, 95, and 2000 representing the current benchmark for the era) at each data point. It compares each microarchitecture advance with a design without the ad-

Figure 1. Evolution of Intel microprocessors 1971–2009.

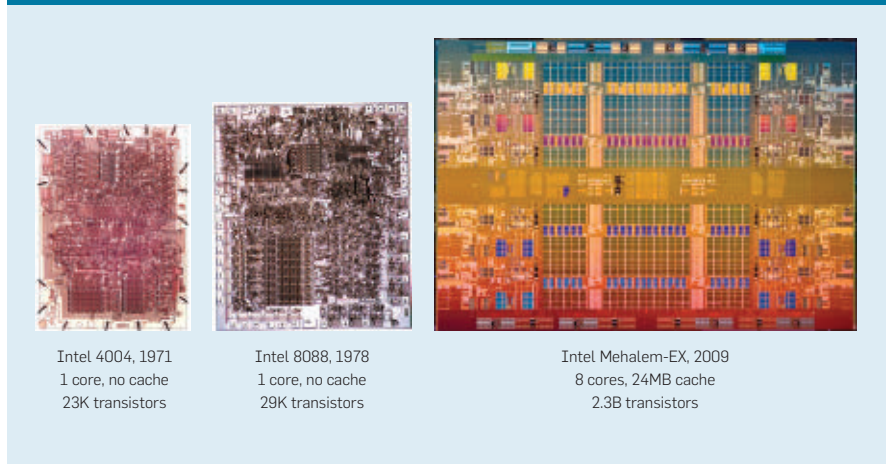
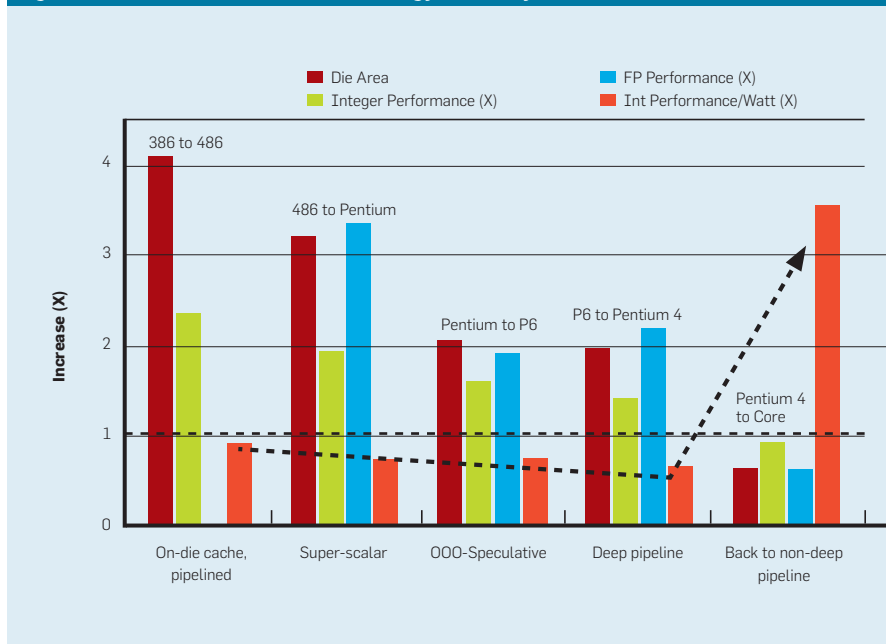


Figure 2. Architecture advances and energy efficiency.



vance (such as introducing an on-die cache by comparing 486 to 386 in 1 μ technology and superscalar microarchitecture of Pentium in 0.7 μ technology with 486).

This data shows that on-die caches and pipeline architectures used transistors well, providing a significant performance boost without compromising energy efficiency. In this era, superscalar, and out-of-order architectures provided sizable performance benefits at a cost in energy efficiency. Of these architectures, deep-pipelined design seems to have delivered the lowest performance increase for the same area and power increase as out-of-order and speculative design, incurring the greatest cost in energy efficiency. The term “deep pipelined architecture” describes deeper pipeline, as well as other circuit and microarchitectural techniques (such as trace cache and self-resetting domino logic) employed to achieve even higher frequency. Evident from the data is that reverting to a non-deep pipeline reclaimed energy efficiency by dropping these expensive and inefficient techniques.

When transistor performance increases frequency of operation, the performance of a well-tuned system generally increases, with frequency subject to the performance limits of other parts of the system. Historically, microarchitecture techniques exploiting the growth in available transistors have delivered performance increases empirically described by Pollack’s Rule,³² whereby performance increases (when not limited by other parts of the system) as the square root of the number of transistors or area of a processor (see Figure 3). According to Pollack’s Rule, each new technology generation doubles the number of transistors on a chip, enabling a new microarchitecture that delivers a 40% performance increase. The faster transistors provide an additional 40% performance (increased frequency), almost doubling overall performance within the same power envelope (per scaling theory). In practice, however, implementing a new microarchitecture every generation is difficult, so microarchitecture gains are typically less. In recent microprocessors, the increasing drive for energy efficiency has

caused designers to forego many of these microarchitecture techniques.

As Pollack’s Rule broadly captures area, power, and performance trade-offs from several generations of microarchitecture, we use it as a rule of thumb to estimate single-thread performance in various scenarios throughout this article.

Cache memory architecture. Dynamic memory technology (DRAM) has also advanced dramatically with Moore’s Law over the past 40 years but with different characteristics. For example, memory density has doubled nearly every two years, while performance has improved more slowly (see Figure 4a). This slower improvement in cycle time has produced a memory bottleneck that could reduce a system’s overall performance. Figure 4b outlines the increasing speed disparity, growing from 10s to 100s of processor clock cycles per memory access. It has lately flattened out due to the flattening of processor clock frequency.

Unaddressed, the memory-latency gap would have eliminated and could still eliminate most of the benefits of processor improvement.

The reason for slow improvement of DRAM speed is practical, not technological. It’s a misconception that DRAM technology based on capacitor storage is inherently slower; rather, the memory organization is optimized for density and lower cost, making it slower. The DRAM market has demanded large capacity at minimum cost over speed, depending on small and fast caches on the microprocessor die to emulate high-performance memory by providing the necessary bandwidth and low latency based on data locality. The emergence of sophisticated, yet effective, memory hierarchies allowed DRAM to emphasize density and cost over speed. At first, processors used a single level of cache, but, as processor speed increased, two to three levels of cache hierarchies were introduced to span the growing speed gap between

Figure 3. Increased performance vs. area in the same process technology follows Pollack’s Rule.

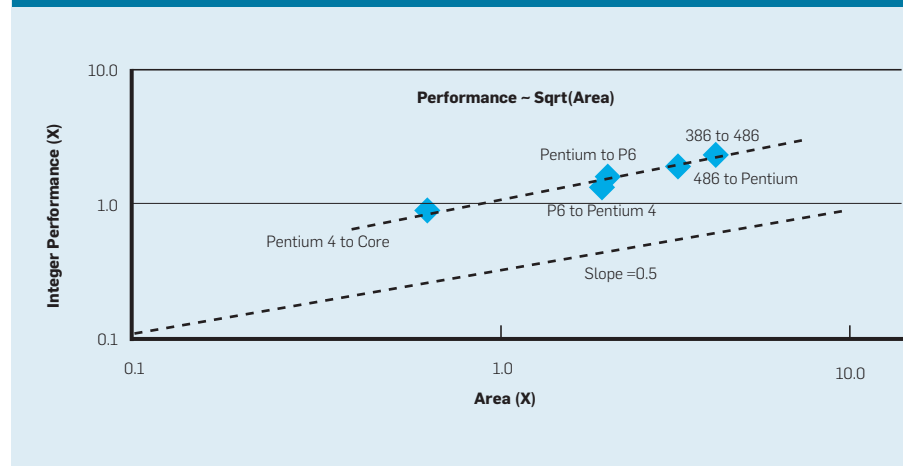
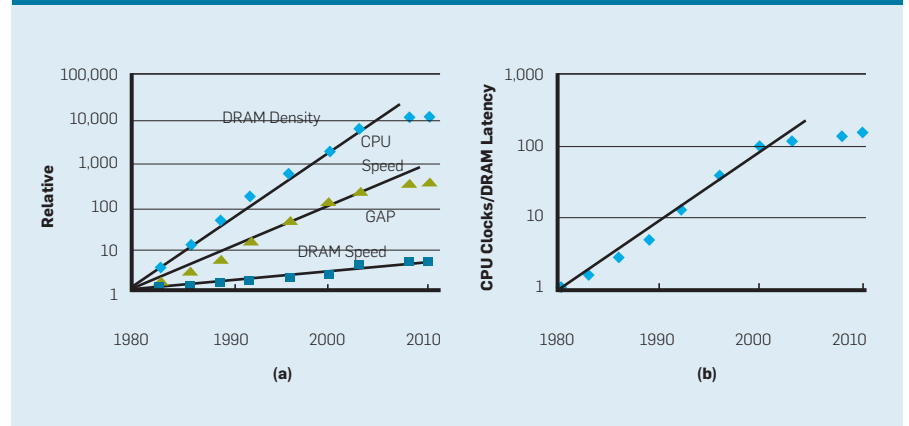


Figure 4. DRAM density and performance, 1980–2010.



processor and memory.^{33,37} In these hierarchies, the lowest-level caches were small but fast enough to match the processor's needs in terms of high bandwidth and low latency; higher levels of the cache hierarchy were then optimized for size and speed.

Figure 5 outlines the evolution of on-die caches over the past two decades, plotting cache capacity (a) and percentage of die area (b) for Intel microprocessors. At first, cache sizes increased slowly, with decreasing die

area devoted to cache, and most of the available transistor budget was devoted to core microarchitecture advances. During this period, processors were probably cache-starved. As energy became a concern, increasing cache size for performance has proven more energy efficient than additional core-microarchitecture techniques requiring energy-intensive logic. For this reason, more and more transistor budget and die area are allocated in caches.

The transistor-scaling-and-micro-

architecture-improvement cycle has been sustained for more than two decades, delivering 1,000-fold performance improvement. How long will it continue? To better understand and predict future performance, we decouple performance gain due to transistor speed and microarchitecture by comparing the same microarchitecture on different process technologies and new microarchitectures with the previous ones, then compound the performance gain.

Figure 6 divides the cumulative 1,000-fold Intel microprocessor performance increase over the past two decades into performance delivered by transistor speed (frequency) and due to microarchitecture. Almost two-orders-of-magnitude of this performance increase is due to transistor speed alone, now leveling off due to the numerous challenges described in the following sections.

The Next 20 Years

Microprocessor technology has delivered three-orders-of-magnitude performance improvement over the past two decades, so continuing this trajectory would require at least 30x performance increase by 2020. Micropro-

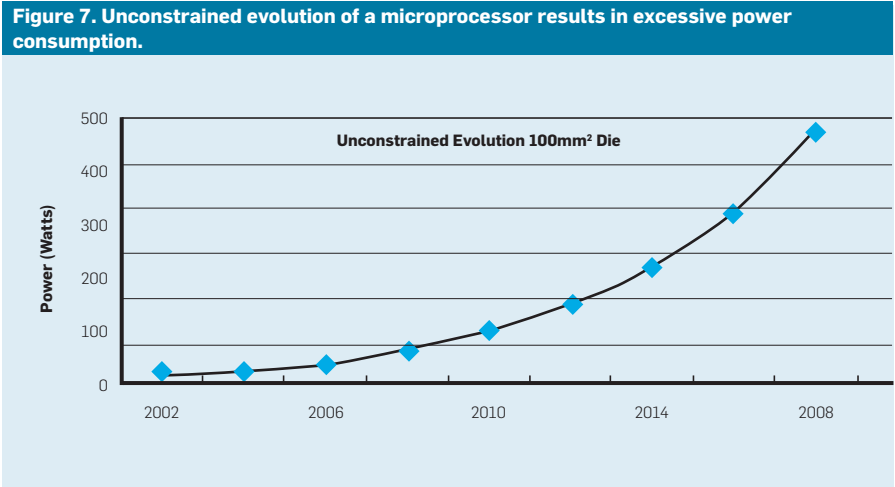
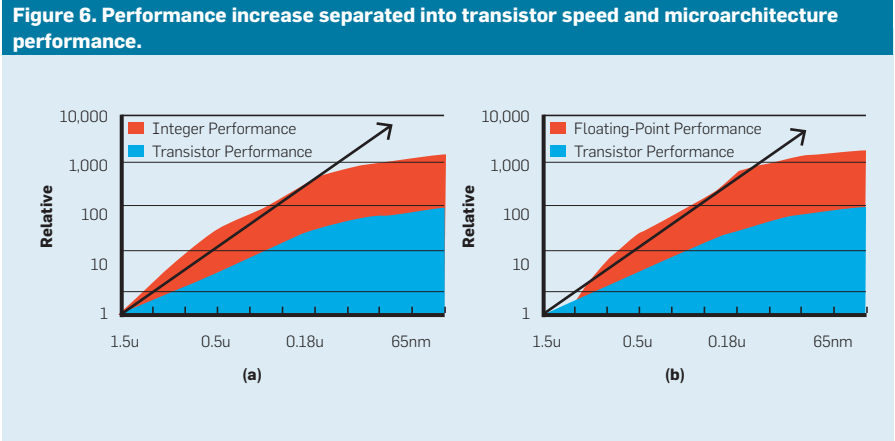
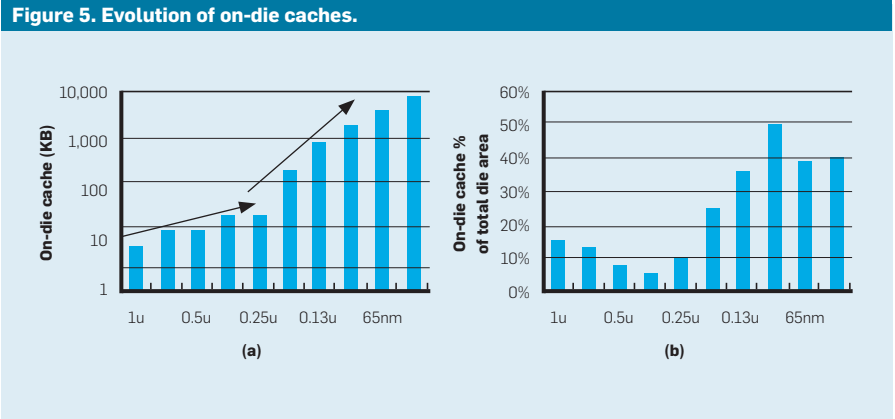


Table 1. New technology scaling challenges.

Decreased transistor scaling benefits: Despite continuing miniaturization, little performance improvement and little reduction in switching energy (decreasing performance benefits of scaling) [ITRS].
Flat total energy budget: package power and mobile/embedded computing drives energy-efficiency requirements.

Table 2. Ongoing technology scaling.

Increasing transistor density (in area and volume) and count: through continued feature scaling, process innovations, and packaging innovations.
Need for increasing locality and reduced bandwidth per operation: as performance of the microprocessor increases, and the data sets for applications continue to grow.

cessor-performance scaling faces new challenges (see Table 1) precluding use of energy-inefficient microarchitecture innovations developed over the past two decades. Further, chip architects must face these challenges with an ongoing industry expectation of a 30x performance increase in the next decade and 1,000x increase by 2030 (see Table 2).

As the transistor scales, supply voltage scales down, and the threshold voltage of the transistor (when the transistor starts conducting) also scales down. But the transistor is not a perfect switch, leaking some small amount of current when turned off, increasing exponentially with reduction in the threshold voltage. In addition, the exponentially increasing transistor-integration capacity exacerbates the effect; as a result, a substantial portion of power consumption is due to leakage. To keep leakage under control, the threshold voltage cannot be lowered further and, indeed, must increase, reducing transistor performance.¹⁰

As transistors have reached atomic dimensions, lithography and variability pose further scaling challenges, affecting supply-voltage scaling.¹¹ With limited supply-voltage scaling, energy and power reduction is limited, adversely affecting further integration of transistors. Therefore, transistor-integration capacity will continue with scaling, though with limited performance and power benefit. The challenge for chip architects is to use this integration capacity to continue to improve performance.

Package power/total energy consumption limits number of logic transistors. If chip architects simply add more cores as transistor-integration capacity becomes available and operate the chips at the highest frequency the transistors and designs can achieve, then the power consumption of the chips would be prohibitive (see Figure 7). Chip architects must limit frequency and number of cores to keep power within reasonable bounds, but doing so severely limits improvement in microprocessor performance.

Consider the transistor-integration capacity affordable in a given power envelope for reasonable die size. For regular desktop applications the pow-

Death of 90/10 Optimization, Rise of 10×10 Optimization

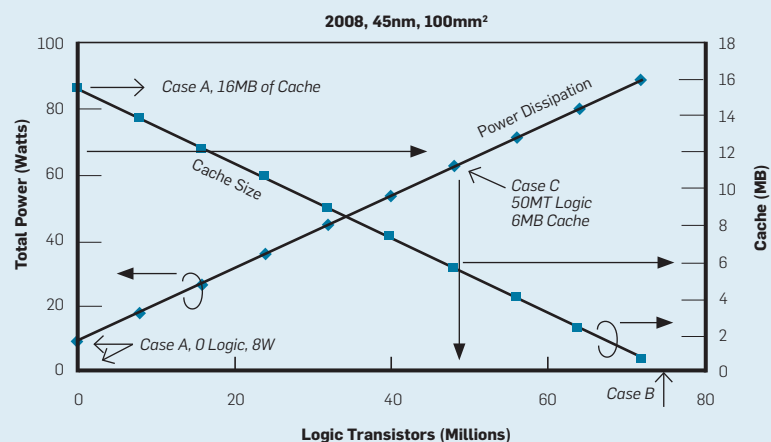
Traditional wisdom suggests investing maximum transistors in the 90% case, with the goal of using precious transistors to increase single-thread performance that can be applied broadly. In the new scaling regime typified by slow transistor performance and energy improvement, it often makes no sense to add transistors to a single core as energy efficiency suffers. Using additional transistors to build more cores produces a limited benefit—increased performance for applications with thread parallelism. In this world, 90/10 optimization no longer applies. Instead, optimizing with an accelerator for a 10% case, then another for a different 10% case, then another 10% case can often produce a system with better overall energy efficiency and performance. We call this “10×10 optimization,”¹⁴ as the goal is to attack performance as a set of 10% optimization opportunities—a different way of thinking about transistor cost, operating the chip with 10% of the transistors active—90% inactive, but a different 10% at each point in time.

Historically, transistors on a chip were expensive due to the associated design effort, validation and testing, and ultimately manufacturing cost. But 20 generations of Moore’s Law and advances in design and validation have shifted the balance. Building systems where the 10% of the transistors that can operate within the energy budget are configured optimally (an accelerator well-suited to the application) may well be the right solution. The choice of 10 cases is illustrative, and a 5×5, 7×7, 10×10, or 12×12 architecture might be appropriate for a particular design.

er envelope is around 65 watts, and the die size is around 100mm². Figure 8 outlines a simple analysis for 45nm process technology node; the x-axis is the number of logic transistors integrated on the die, and the two y-axes are the amount of cache that would fit and the power the die would consume. As the number of logic transistors on the die increases (x-axis), the size of the cache decreases, and power dissipation increases. This analysis assumes average activity factor for logic and

cache observed in today’s microprocessors. If the die integrates no logic at all, then the entire die could be populated with about 16MB of cache and consume less than 10 watts of power, since caches consume less power than logic (Case A). On the other hand, if it integrates no cache at all, then it could integrate 75 million transistors for logic, consuming almost 90 watts of power (Case B). For 65 watts, the die could integrate 50 million transistors for logic and about 6MB of cache (Case C).

Figure 8. Transistor integration capacity at a fixed power envelope.



This design point matches the dual-core microprocessor on 45nm technology (Core2 Duo), integrating two cores of 25 million transistors each and 6MB of cache in a die area of about 100mm².

If this analysis is performed for future technologies, assuming (our best estimates) modest frequency increase 15% per generation, 5% reduction in supply voltage, and 25% reduction of

capacitance, then the results will be as they appear in Table 1. Note that over the next 10 years we expect increased total transistor count, following Moore's Law, but logic transistors increase by only 3x and cache transistors increase more than 10x. Applying Pollack's Rule, a single processor core with 150 million transistors will provide only about 2.5x microarchitecture performance improvement over today's 25-million-transistor core, well shy of our 30x goal, while 80MB of cache is probably more than enough for the cores (see Table 3).

The reality of a finite (essentially fixed) energy budget for a microprocessor must produce a qualitative shift in how chip architects think about architecture and implementation. First, energy-efficiency is a key metric for these designs. Second, energy-proportional computing must be the ultimate goal for both hardware architecture and software-application design. While this ambition is noted in macro-scale computing in large-scale data centers,⁵ the idea of micro-scale energy-proportional computing in microprocessors is even more challenging. For microprocessors operating within a finite energy budget, energy efficiency corresponds directly to higher performance, so the quest for extreme energy efficiency is the ultimate driver for performance.

In the following sections, we outline key challenges and sketch potential approaches. In many cases, the challenges are well known and the subject of significant research over many years. In all cases, they remain critical but daunting for the future of microprocessor performance:

Organizing the logic: Multiple cores and customization. The historic measure of microprocessor capability is the single-thread performance of a traditional core. Many researchers have observed that single-thread performance has already leveled off, with only modest increases expected in the coming decades. Multiple cores and customization will be the major drivers for future microprocessor performance (total chip performance). Multiple cores can increase computational throughput (such as a 1x–4x increase could result from four cores), and customization can reduce execution la-

Figure 9. Three scenarios for integrating 150-million logic transistors into cores.

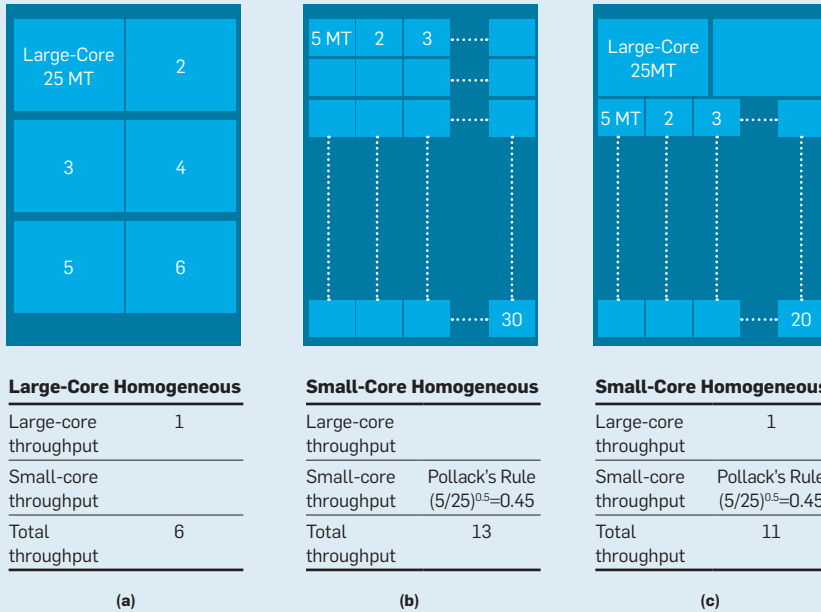


Figure 10. A system-on-a-chip from Texas Instruments.

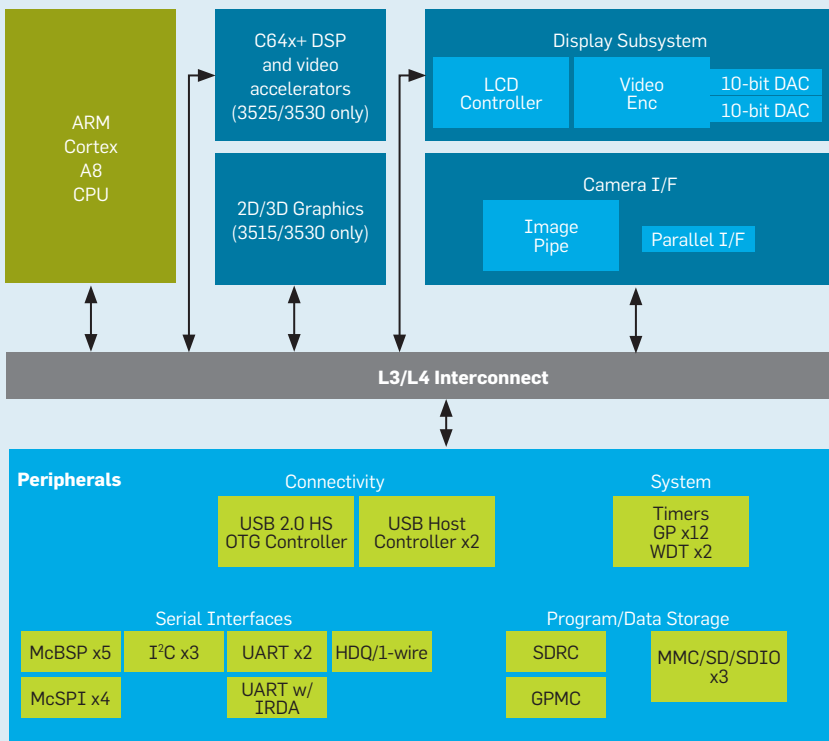


Table 3. Extrapolated transistor integration capacity in a fixed power envelope.

Year	Logic Transistors (Millions)	Cache MB
2008	50	6
2014	100	25
2018	150	80

tency. Clearly, both techniques—multiple cores and customization—can improve energy efficiency, the new fundamental limiter to capability.

Choices in multiple cores. Multiple cores increase computational throughput by exploiting Moore's Law to replicate cores. If the software has no parallelism, there is no performance benefit. However, if there is parallelism, the computation can be spread across multiple cores, increasing overall computational performance (and reducing latency). Extensive research on how to organize such systems dates to the 1970s.^{29,39}

Industry has widely adopted a multicore approach, sparking many questions about number of cores and size/power of each core and how they coordinate.^{6,36} But if we employ 25-million-transistor cores (circa 2008), the 150-million-logic-transistor budget expected in 2018 gives 6x potential throughput improvement (2x from frequency and 3x from increased logic transistors), well short of our 30x goal. To go further, chip architects must consider more radical options of smaller cores in greater numbers, along with innovative ways to coordinate them.

Looking to achieve this vision, consider three potential approaches to deploying the feasible 150 million logic transistors, as in Table 1. In Figure 9, option (a) is six large cores (good single-thread performance, total potential throughput of six); option (b) is 30 smaller cores (lower single-thread performance, total potential throughput of 13); and option (c) is a hybrid approach (good single-thread performance for low parallelism, total potential throughput of 11).

Many more variations are possible on this spectrum of core size and num-

ber of cores, and the related choices in a multicore processor with uniform instruction set but heterogeneous implementation are an important part of increasing performance within the transistor budget and energy envelope.

Choices in hardware customization. Customization includes fixed-function accelerators (such as media codecs, cryptography engines, and compositing engines), programmable accelerators, and even dynamically customizable logic (such as FPGAs and other dynamic structures). In general, customization increases computational performance by exploiting hardwired or customized computation units, customized wiring/interconnect for data movement, and reduced instruction-sequence overheads at some cost in generality. In addition, the level of parallelism in hardware can be customized to match the precise needs of the computation; computation benefits from hardware customization only when it matches the specialized hardware structures. In some cases, units hardwired to a particular data representation or computational algorithm can achieve 50x–500x greater energy efficiency than a general-purpose register organization. Two studies^{21,22} of a media encoder and TCP offload engine illustrate the large energy-efficiency improvement that is possible.

Due to battery capacity and heat-dissipation limits, for many years energy has been the fundamental limiter for computational capabil-

ity in smartphone system-on-a-chip (SoC). As outlined in Figure 10, such an SoC might include as many as 10 to 20 accelerators to achieve a superior balance of energy efficiency and performance. This example could also include graphics, media, image, and cryptography accelerators, as well as support for radio and digital signal processing. As one might imagine, one of these blocks could be a dynamically programmable element (such as an FPGA or a software-programmable processor).

Another customization approach constrains the types of parallelism that can be executed efficiently, enabling a simpler core, coordination, and memory structures; for example, many CPUs increase energy efficiency by restricting memory access structure and control flexibility in single-instruction, multiple-data or vector (SIMD) structures,^{1,2} while GPUs encourage programs to express structured sets of threads that can be aligned and executed efficiently.^{26,30} This alignment reduces parallel coordination and memory-access costs, enabling use of large numbers of cores and high peak performance when applications can be formulated with a compatible parallel structure. Several microprocessor manufacturers have announced future mainstream products that integrate CPUs and GPUs.

Customization for greater energy or computational efficiency is a longstanding technique, but broad adop-

Table 4. Logic organization challenges, trends, directions.

Challenge	Near-Term	Long-Term
Integration and memory model	I/O-based interaction, shared memory spaces, explicit coherence management	Intelligent, automatic data movement among heterogeneous cores, managed by software-hardware partnership
Software transparency	Explicit partition and mapping, virtualization, application management	Hardware-based state adaptation and software-hardware partnership for management
Lower-power cores	Heterogeneous cores, vector extensions, and GPU-like techniques to reduce instruction- and data-movement cost	Deeper, explicit storage hierarchy within the core; integrated computation in registers
Energy management	Hardware dynamic voltage scaling and intelligent adaptive management, software core selection and scheduling	Predictive core scheduling and selection to optimize energy efficiency and minimize data movement
Accelerator variety	Increasing variety, library-based encapsulation (such as DX and OpenGL) for specific domains	Converged accelerators in a few application categories and increasing open programmability for the accelerators

tion has been slowed by continued improvement in microprocessor single-thread performance. Developers of software applications had little incentive to customize for accelerators that might be available on only a fraction of the machines in the field and for which

the performance advantage might soon be overtaken by advances in the traditional microprocessor. With slowing improvement in single-thread performance, this landscape has changed significantly, and for many applications, accelerators may be the only

path toward increased performance or energy efficiency (see Table 4). But such software customization is difficult, especially for large programs (see the sidebar “Decline of 90/10 Optimization, Rise of 10x10 Optimization”).

Orchestrating data movement: Memory hierarchies and interconnects. In future microprocessors, the energy expended for data movement will have a critical effect on achievable performance. Every nano-joule of energy used to move data up and down the memory hierarchy, as well as to synchronize across and data between processors, takes away from the limited budget, reducing the energy available for the actual computation. In this context, efficient memory hierarchies are critical, as the energy to retrieve data from a local register or cache is far less than the energy to go to DRAM or to secondary storage. In addition, data must be moved between processing units efficiently, and task placement and scheduling must be optimized against an interconnection network with high locality. Here, we examine energy and power associated with data movement on the processor die.

Today’s processor performance is on the order of 100Giga-op/sec, and a 30x increase over the next 10 years would increase this performance to 3Tera-op/sec. At minimum, this boost requires 9Tera-operands or 64b x 9Tera-operands (or 576Tera-bits) to be moved each second from registers or memory to arithmetic logic, consuming energy.

Figure 11(a) outlines typical wire delay and energy consumed in moving one bit of data on the die. If the operands move on average 1mm (10% of die size), then at the rate of 0.1pJ/bit, the 576Tera-bits/sec of movement consumes almost 58 watts with hardly any energy budget left for computation. If most operands are kept local to the execution units (such as in register files) and the data movement is far less than 1mm, on, say, the order of only 0.1mm, then the power consumption is only around 6 watts, allowing ample energy budget for the computation.

Cores in a many-core system are typically connected through a network-on-a-chip to move data around the cores.⁴⁰ Here, we examine the ef-

Figure 11. On-die interconnect delay and energy (45nm).

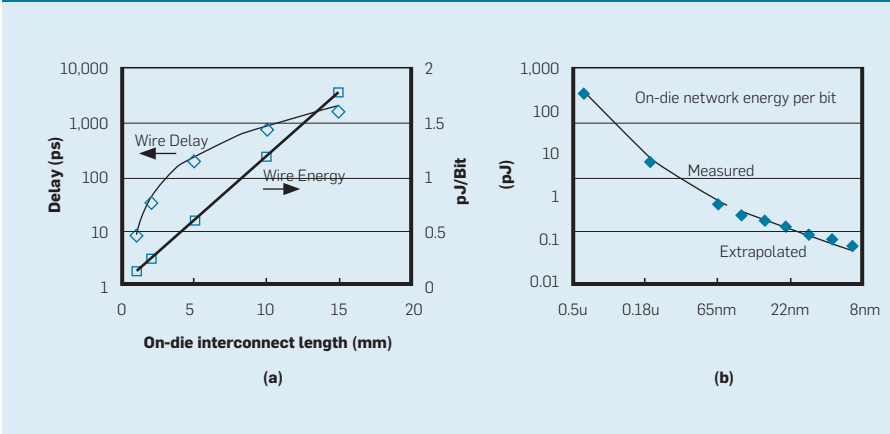


Figure 12. Hybrid switching for network-on-a-chip.

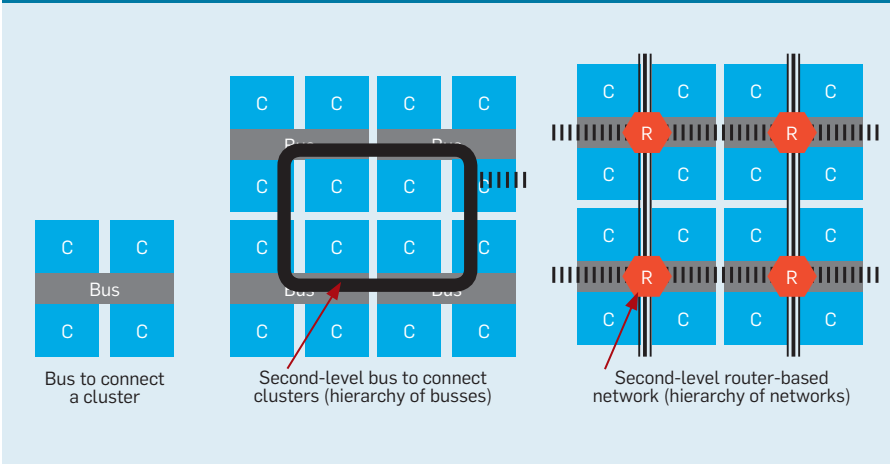


Table 5. Data movement challenges, trends, directions.

Challenge	Near-Term	Long-Term
Parallelism	Increased parallelism	Heterogeneous parallelism and customization, hardware/runtime placement, migration, adaptation for locality and load balance
Data Movement/ Locality	More complex, more exposed hierarchies; new abstractions for control over movement and “snooping”	New memory abstractions and mechanisms for efficient vertical data locality management with low programming effort and energy
Resilience	More aggressive energy reduction; compensated by recovery for resilience	Radical new memory technologies (new physics) and resilience techniques
Energy Proportional Communication	Fine-grain power management in packet fabrics	Exploitation of wide data, slow clock, and circuit-based techniques
Reduced Energy	Low-energy address translation	Efficient multi-level naming and memory-hierarchy management

fect of such a network on power consumption. Figure 11(b) shows the energy consumed in moving a bit across a hop in such a network, measured in historic networks, and extrapolated into the future from previous assumptions. If only 10% of the operands move over the network, traversing 10 hops on average, then at the rate of 0.06pJ/bit the network power would be 35 watts, more than half the power budget of the processor.

As the energy cost of computation is reduced by voltage scaling (described later), emphasizing compute throughput, the cost of data movement starts to dominate. Therefore, data movement must be restricted by keeping data locally as much as possible. This restriction also means the size of local storage (such as a register file) must increase substantially. This increase is contrary to conventional thinking of register files being small and thus fast. With voltage scaling the frequency of operation is lower anyway, so it makes sense to increase the size of the local storage at the expense of speed.

Another radical departure from conventional thinking is the role of the interconnect network on the chip. Recent parallel machine designs have been dominated by packet-switching,^{6,8,24,40} so multicore networks adopted this energy-intensive approach. In the future, data movement over these networks must be limited to conserve energy, and, more important, due to the large size of local storage data bandwidth, demand on the network will be reduced. In light of these findings on-die-network architectures need revolutionary approaches (such as hybrid packet/circuit switching⁴).

Many older parallel machines used irregular and circuit-switched networks^{31,41}; Figure 12 describes a return to hybrid switched networks for on-chip interconnects. Small cores in close proximity could be interconnected into clusters with traditional buses that are energy efficient for data movement over short distances. The clusters could be connected through wide (high-bandwidth) low-swing (low-energy) busses or through packet- or circuit-switched networks, depending on distance. Hence the network-on-a-chip could be hierarchical and heterogeneous, a radical departure from the

Figure 13. Improving energy efficiency through voltage scaling.

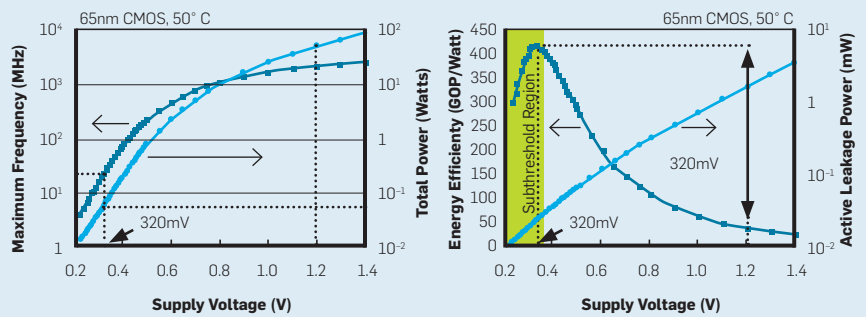


Table 6. Circuits challenges, trends, directions.

Challenge	Near-Term	Long-Term
Power, energy efficiency	Continuous dynamic voltage and frequency scaling, power gating, reactive power management	Discrete dynamic voltage and frequency scaling, near threshold operation, proactive fine-grain power and energy management
Variation	Speed binning of parts, corrections with body bias or supply voltage changes, tighter process control	Dynamic reconfiguration of many cores by speed
Gradual, temporal, intermittent, and permanent faults	Guard-bands, yield loss, core sparing, design for manufacturability	Resilience with hardware/software co-design, dynamic in-field detection, diagnosis, reconfiguration and repair, adaptability, and self-awareness

traditional parallel-machine approach (see Table 5).

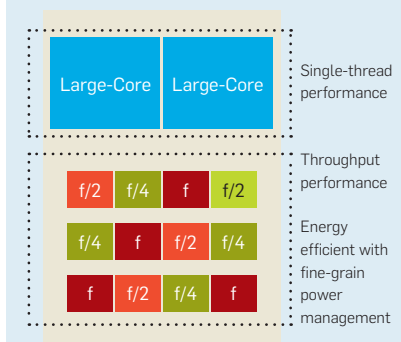
The role of microprocessor architect must expand beyond the processor core, into the whole platform on a chip, optimizing the cores as well as the network and other subsystems.

Pushing the envelope: Extreme circuits, variability, resilience. Our analysis showed that in the power-constrained scenario, only 150 million logic transistors for processor cores and 80MB of cache will be affordable due to energy by 2018. Note that 80MB of cache is not necessary for this system, and a large portion of the cache-transistor budget can be utilized to integrate even more cores if it can be done with the power-consumption density of a cache, which is 10x less than logic. This approach can be achieved through aggressive scaling of supply voltage.²⁵

Figure 13 outlines the effectiveness of supply-voltage scaling when the chip is designed for it. As the supply voltage is reduced, frequency

also reduces, but energy efficiency increases. When the supply voltage is reduced all the way to the transistor's threshold, energy efficiency increases by an order of magnitude. Employing this technique on large cores would dramatically reduce single-thread performance and is hence not recommended. However, smaller cores used

Figure 14. A heterogeneous many-core system with variation.



for throughput would certainly benefit from it. Moreover, the transistor budget from the unused cache could be used to integrate even more cores with the power density of the cache. Aggressive voltage scaling provides an avenue for utilizing the unused transistor-integration capacity for logic to deliver higher performance.

Aggressive supply-voltage scaling comes with its own challenges (such as variations). As supply voltage is reduced toward a transistor's threshold voltage, the effect of variability is even worse, because the speed of a circuit is proportional to the voltage overdrive (supply voltage minus threshold voltage). Moreover, as supply voltage approaches the threshold, any small change in threshold voltage affects the speed of the circuit. Therefore, variation in the threshold voltage manifests itself as variation in the speed of the core, the slowest circuit in the core determines the frequency of operation of the core, and a large core is more susceptible to lower frequency of operation due to variations. On the other hand, a large number of small cores has a better distribution of fast and slow small cores and can better even out the effect of variations. We next discuss an example system that is variation-tolerant, energy-efficient, energy-proportional, and fine-grain power managed.

A hypothetical heterogeneous processor (see Figure 14) consists of a small number of large cores for single-thread performance and many small cores for throughput performance. Supply voltage and the frequency of any

given core are individually controlled such that the total power consumption is within the power envelope. Many small cores operate at lower voltages and frequency for improved energy efficiency, while some small cores operate near threshold voltage at the lowest frequency but at higher energy efficiency, and some cores may be turned off completely. Clock frequencies need not be continuous; steps (in powers of two) keep the system synchronous and simple without compromising performance while also addressing variation tolerance. The scheduler dynamically monitors workload and configures the system with the proper mix of cores and schedules the workload on the right cores for energy-proportional computing. Combined heterogeneity, aggressive supply-voltage scaling, and fine-grain power (energy) management enables utilization of a larger fraction of transistor-integration capacity, moving closer to the goal of 30x increase in compute performance (see Table 6).

Software challenges renewed: Programmability versus efficiency. The end of scaling of single-thread performance already means major software challenges; for example, the shift to symmetric parallelism has created perhaps the greatest software challenge in the history of computing,^{12,15} and we expect future pressure on energy-efficiency will lead to extensive use of heterogeneous cores and accelerators, further exacerbating the software challenge. Fortunately, the past decade has seen increasing adoption of high-level “productivity” languages^{20,34,35} built on

advanced interpretive and compiler technologies, as well as increasing use of dynamic translation techniques. We expect these trends to continue, with higher-level programming, extensive customization through libraries, and sophisticated automated performance search techniques (such as autotuning) will be even more important.

Extreme studies^{27,38} suggest that aggressive high-performance and extreme-energy-efficient systems may go further, eschewing the overhead of programmability features that software engineers have come to take for granted; for example, these future systems may drop hardware support for a single flat address space (which normally wastes energy on address manipulation/computing), single-memory hierarchy (coherence and monitoring energy overhead), and steady rate of execution (adapting to the available energy budget). These systems will place more of these components under software control, depending on increasingly sophisticated software tools to manage the hardware boundaries and irregularities with greater energy efficiency. In extreme cases, high-performance computing and embedded applications may even manage these complexities explicitly. Most architectural features and techniques we've discussed here shift more responsibility for distribution of the computation and data across the compute and storage elements of microprocessors to software.^{13,18} Shifting responsibility increases potential achievable energy efficiency, but realizing it depends on significant advances in applications, compilers and runtimes, and operating systems to understand and even predict the application and workload behavior.^{7,16,19} However, these advances require radical research breakthroughs and major changes in software practice (see Table 7).

Conclusion

The past 20 years were truly the great old days for Moore's Law scaling and microprocessor performance; dramatic improvements in transistor density, speed, and energy, combined with microarchitecture and memory-hierarchy techniques delivered 1,000-fold microprocessor performance improvement. The next 20 years—the

Table 7. Software challenges, trends, directions.

Challenge	Near-Term	Long-Term
1,000-fold software parallelism	Data parallel languages and “mapping” of operators, library and tool-based approaches	New high-level languages, compositional and deterministic frameworks
Energy-efficient data movement and locality	Manual control, profiling, maturing to automated techniques (auto-tuning, optimization)	New algorithms, languages, program analysis, runtime, and hardware techniques
Energy management	Automatic fine-grain hardware management	Self-aware runtime and application-level techniques that exploit architecture features for visibility and control
Resilience	Algorithmic, application-software approaches, adaptive checking and recovery	New hardware-software partnerships that minimize checking and recomputation energy

pretty good new days, as progress continues—will be more difficult, with Moore's Law scaling producing continuing improvement in transistor density but comparatively little improvement in transistor speed and energy. As a result, the frequency of operation will increase slowly. Energy will be the key limiter of performance, forcing processor designs to use large-scale parallelism with heterogeneous cores, or a few large cores and a large number of small cores operating at low frequency and low voltage, near threshold. Aggressive use of customized accelerators will yield the highest performance and greatest energy efficiency on many applications. Efficient data orchestration will increasingly be critical, evolving to more efficient memory hierarchies and new types of interconnect tailored for locality and that depend on sophisticated software to place computation and data so as to minimize data movement. The objective is ultimately the purest form of energy-proportional computing at the lowest-possible levels of energy. Heterogeneity in compute and communication hardware will be essential to optimize for performance for energy-proportional computing and coping with variability. Finally, programming systems will have to comprehend these restrictions and provide tools and environments to harvest the performance.

While no one can reliably predict the end of Si CMOS scaling, for this future scaling regime, many electrical engineers have begun exploring new types of switches and materials (such as compound semiconductors, carbon nanotubes, and graphene) with different performance and scaling characteristics from Si CMOS, posing new types of design and manufacturing challenges. However, all such technologies are in their infancy, probably not ready in the next decade to replace silicon but will pose the same challenges with continued scaling. Quantum electronics (such as quantum dots) are even farther out and when realized will reflect major challenges of its own, with yet newer models of computation, architecture, manufacturing, variability, and resilience.

Because the future winners are far from clear today, it is way too early to

predict whether some form of scaling (perhaps energy) will continue or there will be no scaling at all. The pretty good old days of scaling that processor design faces today are helping prepare us for these new challenges. Moreover, the challenges processor design will face in the next decade will be dwarfed by the challenges posed by these alternative technologies, rendering today's challenges a warm-up exercise for what lies ahead.

Acknowledgments

This work was inspired by the Exascale study working groups chartered in 2007 and 2008 by Bill Harrod of DARPA. We thank him and the members and presenters to the working groups for valuable insightful discussions over the past few years. We also thank our colleagues at Intel who have improved our understanding of these issues through many thoughtful discussions. Thanks, too, to the anonymous reviewers whose extensive feedback greatly improved the article. 

References

- Advanced Vector Extensions. Intel; http://en.wikipedia.org/wiki/Advanced_Vector_Extensions
- Altivec, Apple, IBM, Freescale; <http://en.wikipedia.org/wiki/Altivec>
- Amdahl, G. Validity of the single-processor approach to achieving large-scale computing capability. AFIPS Joint Computer Conference (Apr. 1967), 483–485.
- Anders, M. et al. A 4.1Tb/s bisection-bandwidth 560Gb/s/W streaming circuit-switched 8x8 mesh network-on-chip in 45nm CMOS. International Solid State Circuits Conference (Feb. 2010).
- Barroso, L.A. and Högl, U. The case for energy-proportional computing. *IEEE Computer* 40, 12 (Dec. 2007).
- Bell, S. et al. TILE64 processor: A 64-core SoC with mesh interconnect. IEEE International Solid-State Circuits Conference (2008).
- Bienia, C. et al. The PARSEC benchmark suite: Characterization and architectural implications. The 17th International Symposium on Parallel Architectures and Compilation Techniques (2008).
- Blumrich, M. et al. *Design and Analysis of the Blue Gene/L Torus Interconnection Network*. IBM Research Report, 2003.
- Borkar, S. Designing reliable systems from unreliable components: The challenges of transistor variability and degradation. *IEEE Micro* 25, 6 (Nov.–Dec. 2005).
- Borkar, S. Design challenges of technology scaling. *IEEE Micro* 19, 4 (July–Aug. 1999).
- Borkar, S. et al. Parameter variations and impact on circuits and microarchitecture. The 40th Annual Design Automation Conference (2003).
- Catanzaro, B. et al. Ubiquitous parallel computing from Berkeley, Illinois, and Stanford. *IEEE Micro* 30, 2 (2010).
- Cray, Inc. *Chapel Language Specification*. Seattle, WA, 2010; <http://chapel.cray.com/spec/spec-0.795.pdf>
- Chien, A. 10x10: A general-purpose architectural approach to heterogeneity and energy efficiency. *The Third Workshop on Emerging Parallel Architectures at the International Conference on Computational Science* (June 2011).
- Chien, A. Pervasive parallel computing: An historic opportunity for innovation in programming and architecture. *ACM Principles and Practice of Parallel Programming* (2007).
- Cooper, B. et al. Benchmarking cloud serving systems with YCSB. ACM Symposium on Cloud Computing (June 2010).
- Dennard, R. et al. Design of ion-implanted MOSFETs with very small physical dimensions. *IEEE Journal of Solid State Circuits* SC-9, 5 (Oct. 1974), 256–268.
- Fatahalian, K. et al. Sequoia: Programming the memory hierarchy. ACM/IEEE Conference on Supercomputing (Nov. 2006).
- Flinn, J. et al. Managing battery lifetime with energy-aware adaptation. *ACM Transactions on Computer Systems* 22, 2 (May 2004).
- Gosling, J. et al. *The Java Language Specification, Third Edition*. Addison-Wesley, 2005.
- Hameed, R. et al. Understanding sources of inefficiency in general-purpose chips. International Symposium on Computer Architecture (2010).
- Hoskote, Y. et al. A TCP offload accelerator for 10Gb/s Ethernet in 90-nm CMOS. *IEEE Journal of Solid-State Circuits* 38, 11 (Nov. 2003).
- International Technology Roadmap for Semiconductors, 2009; <http://www.itrs.net/Links/2009ITRS/Home2009.htm>
- Karamcheti, V. et al. Comparison of architectural support for messaging in the TMC CM-5 and Cray T3D. International Symposium on Computer Architecture (1995).
- Kaul, H. et al. A 320mV 56W 411GOPS/Watt ultra-low-voltage motion-estimation accelerator in 65nm CMOS. *IEEE Journal of Solid-State Circuits* 44, 1 (Jan. 2009).
- The Khronos Group. *OpenCL, the Open Standard for Heterogeneous Parallel Programming*, Feb. 2009; <http://www.khronos.org/opencl/>
- Kogge, P. et al. *Exascale Computing Study: Technology Challenges in Achieving an Exascale System*; http://users.ece.gatech.edu/mrichard/ExascaleComputingStudyReports/exascale_final_report_100208.pdf
- Mazor, S. The history of microcomputer-invention and evolution. *Proceedings of the IEEE* 83, 12 (Dec. 1995).
- Noguchi, K., Ohnishi, I., and Morita, H. Design considerations for a heterogeneous tightly coupled multiprocessor system. AFIPS National Computer Conference (1975).
- Nvidia Corp. *CUDA Programming Guide Version 2.0*, June 2008; http://www.nvidia.com/object/cuda_home_new.html
- Pfister, G. et al. The research parallel processor prototype (RP3): Introduction and architecture. *International Conference on Parallel Processing* (Aug. 1985).
- Pollack, F. *Pollack's Rule of Thumb for Microprocessor Performance and Area*; http://en.wikipedia.org/wiki/Pollack's_Rule
- Przybylski, S.A. et al. Characteristics of performance-optimal multi-level cache hierarchies. International Symposium on Computer Architecture (June 1989).
- Richter, J. *The CLR Via C#, Second Edition*, 1997.
- Ruby Documentation Project. *Programming Ruby: The Pragmatic Programmer's Guide*; <http://www.ruby-doc.org/docs/ProgrammingRuby/>
- Seiler, L. et al. Larrabee: Many-core x86 architecture for visual computing. *ACM Transactions on Graphics* 27, 3 (Aug. 2008).
- Strecker, W. Transient behavior of cache memories. *ACM Transactions on Computer Systems* 1, 4 (Nov. 1983).
- Sarkar, V. et al. *Exascale Software Study: Software Challenges in Extreme-Scale Systems*; <http://users.ece.gatech.edu/mrichard/ExascaleComputingStudyReports/ECSS%20report%20101909.pdf>
- Tartar, J. Multiprocessor hardware: An architectural overview. ACM Annual Conference (1980).
- Weingold, E. et al. Baring it all to software: Raw machines. *IEEE Computer* 30, 9 (Sept. 1997).
- Wulf, W. and Bell, C.G. C.mmp: A multi-miniprocessor. *AFIPS Joint Computer Conferences* (Dec. 1972).

Shekhar Borkar (Shekhar.Y.Borkar@intel.com) is an Intel Fellow and director of exascale technology at Intel Corporation, Hillsboro, OR.

Andrew A. Chien (Andrew.Chien@alum.mit.edu) is former vice president of research at Intel Corporation and currently adjunct professor in the Computer Science and Engineering Department at the University of California, San Diego.

DOI:10.1145/1941487.1941508

Privacy-preserving attribution of IP packets can help balance forensics with an individual's right to privacy.

BY MIKHAIL AFANASYEV, TADAYOSHI KOHNO,
JUSTIN MA, NICK MURPHY, STEFAN SAVAGE,
ALEX C. SNOEREN, AND GEOFFREY M. VOELKER

Privacy-Preserving Network Forensics

RESEARCH IN NETWORK security has traditionally focused on defense—mechanisms designed to impede an adversary. However, paraphrasing security expert Butler Lampson, practical security requires a balance between defense and deterrence. While defense may block an adversary's current attacks, only an effective deterrent can prevent the adversary from choosing to attack in the first place. But creating such a deterrent is usually predicated on an effective means of attribution—tying an individual to an action.

In the physical world, this link is established through concrete evidence (such as DNA, fingerprints, and writing samples), but the Internet has no such robust forensic trail. Indeed, functional anonymity is

implicit in the Internet's architecture, since the lowest-level identifiers—network addresses (IP addresses)—are inherently virtual and insecure. It can be extremely challenging to attribute an online action to a physical origin, let alone to a particular individual. Reducing this expectation of anonymity even slightly can potentially disincentivize a range of criminal activity and lengthen the effective lifetime of defense mechanisms.

Compelling though this line of thinking may be, there is a natural tension between the need for attribution and user expectations of privacy. While the public generally appreciates that criminal acts should be subject to scrutiny, civil libertarians are considerably less sanguine about exposing identifying information as a matter of course. Indeed, a recently leaked document, of allegedly International Telecommunications Union provenance, lends credence to libertarian fears, motivating the need for network-level “IP traceback” capabilities via a government's desire to identify anonymous political opponents.¹² Though this is but one example, it is time to explore technical solutions that balance the enforcement interests of the state and the privacy interests of individuals.

We seek to achieve such a balance by introducing a new network-layer capability we call privacy-preserving forensic attribution. We propose a packet-level cryptographic signature

» key insights

- An anonymous Internet protects the privacy of people's Internet activity but means criminal activity could go unattributed.
- A fully attributed, non-anonymous Internet linking all Internet traffic back to its source would help monitor and track criminal activity but could also compromise the privacy of everyday users.
- All Internet packets are inherently anonymous but, with appropriate credentials, authorized parties can revoke that anonymity and attribute packets back to their source.



mechanism allowing properly authorized parties to examine any packet, even those logged months prior, unambiguously identifying the physical machine that sent it. However, absent express authorization, packet signatures do not expose identifying information. Finally, we enforce the correct use of this mechanism by allowing any network element to verify the validity of the signatures it receives.

Our goal is principally to assess the viability of privacy-preserving attribution. Over the past four years, we have built a prototype system called Clue to explore the practical engineering challenges of building a privacy-preserving forensic-attribution capability. This illuminating experience revealed the architectural requirements of our approach while forcing us to confront the challenges of the underlying cryp-

tographic overhead. Surprisingly, we found that much of the overhead can be hidden or amortized through careful protocol design alone. Thus, even our untuned user-level software prototype adds less than 30ms of latency to interactive traffic and achieves bulk TCP throughput exceeding 17Mbps. Moreover, this throughput, which is significantly greater than a typical broadband access connection, is limited by the speed of the receiver; aggregate server throughput can be considerably greater. While numerous challenges remain, our research demonstrates the feasibility of privacy-preserving forensic attribution, encouraging wider consideration of our approach.

Motivating Scenarios

Forensic attribution would create a fundamentally new network-layer ca-

pability, with numerous potential applications, including the subset we survey here. For certain types of crimes, law-enforcement officers routinely face the challenge of how to map between traffic received at some point in the network and the physical device of origin. Establishing this mapping would allow investigators to determine if the same device was used in multiple crimes, if a particular activity was perpetrated by a known device, and potentially to track even the location of a targeted device via IP geolocation.

A concrete example comes from the following email one of the authors received from a corporal in the Rhode Island State Police Department: “We are currently attempting to locate an international fugitive who is wanted for [a violent crime]. We have identified a Pocket PC device attached to the

Internet which the [fugitive] is apparently using...”

Though unable to discuss the details of the case publicly, we can consider the model: The police have a single packet trace known to be from the fugitive’s device (perhaps a threatening email message) and now seek to determine if other threatening messages were also sent from the same device, thereby identifying the fugitive’s current IP address and, hence, geographic area of operation.

Also increasingly common is for authorities to recover computer equipment when suspects are taken into custody. Tying it to other online actions, especially beyond a reasonable doubt, is challenging absent a strong forensic identifier. A strong forensic identifier would allow a recovered laptop to be directly and unambiguously bound to particular messages logged by law enforcement.

Background and Related Work

The value of forensic attribution—use of technical means to establish the presence of a person or object at a crime scene after the fact—has a long history in law enforcement, dating to the late 19th century.^a Lacking an eyewitness to a crime, forensic methods often become a critical tool in an investigation.

Forensic professionals, security researchers, and Internet industry leaders alike recognize that Internet crime poses a special challenge for forensic attribution. Unlike physical evidence (such as fingerprints and DNA), digital objects are, *prima facie*, not unique. The Internet architecture places no technical restrictions on how a host generates packets, so every bit of a packet can be trivially manipulated in subtle ways to hide its provenance.

Indeed, criminals have long spoofed the source address of their Internet traffic to conceal their activity.^{7,16} While a range of systems has been proposed to detect and/or block IP source-address spoofing, such systems are deployed inconsistently, and none are foolproof, even in their ideal embodiment. A long line of literature has focused on tracing spoofed pack-

ets back to their source,^{17,19} but their approaches are motivated by network operational needs and focus on delivering topological path information, an even more abstract property than an IP address.

More important, IP addresses are not unique identifiers, even when used as intended. An IP address represents a topological location in the network for the purpose of routing, not as a way to specify a physical endpoint. It is common for protocols (such as DHCP, Mobile IP, and NAT) to dynamically change the mapping between IP address and physical machine as part of their normal use. While some mappings are logged, this data is commonly retained for only a limited period. “The Internet,” David Aucsmith wrote, “provides criminals two of the most coveted qualities: anonymity and mobility.”³

While we are unaware of other published attempts to provide network-level forensic attribution to physical hosts, a number of related research projects make similar use of cryptographic mechanisms. The source-authentication systems, or “packet passports,” of Liu et al.¹⁴ and “Accountable Internet Protocol” of Andersen et al.¹ both use cryptographic identifiers. However, these systems focus on ensuring the consistency and topological validity of the IP source address itself to prevent address spoofing and do not address either user privacy concerns or the need for long-term physical linkage required for forensic attribution.

Design Goals

Clue reflects the following basic requirements:

Physical names. Attribution must provide a link to a physical object (such as the sending computer). A physical computer can have an associated owner and permit association via sales-and-maintenance records. Moreover, given continuous ownership, a physical computer may be reused in multiple attacks. Identifying this computer allows the attacks to be linked, even if the physical computer is never recovered. Finally, a physical computer accretes physical forensic evidence as a side effect of its use. Indeed, much of this article was written on a laptop with extensive fingerprint evidence on the

screen and, upon examination, a range of hair and skin samples beneath the keyboard. If this laptop were found, it could be unambiguously linked to one of the authors via DNA or fingerprint comparisons;

Per-packet granularity. The best deterrence is when attribution is universal, applied equally to every packet. Moreover, by providing this capability at the network layer, attribution is transparently provided to all higher-layer protocols and applications. That is, there is an inherent benefit in not tying forensic attribution to any particular higher-level network construct. Forensic attribution is most effective when provided as a fundamental building block on which arbitrary higher-level protocols, services, and applications can be built;

Unimpeachability. While we would be satisfied if a new attribution capability simply offered investigative value for those pursuing criminals, we hope that any attribution mechanism would be accepted as sufficiently accurate and trustworthy to provide evidentiary value in the courtroom as well. We therefore seek strong cryptographic mechanisms that are not easily repudiated; and

Indefinite lifetime. On the Internet, as in the physical world, many crimes are not detected until long after they are committed. Placing unnecessary restrictions on the time window for forensic discovery will undoubtedly be exploited by criminals to their advantage. Even today, many online criminals are savvy about the practical investigatory delays imposed by different bilateral mutual legal assistance treaties, locating their data accordingly. Thus, it should be possible to examine a packet and unambiguously attribute its origin long after the packet is received—even months or years later.

These requirements bring us to an architecture in which each packet is self-identifying—tagged with a unique nonforgeable signature identifying the physical machine that sent it. While such attribution does not definitively identify the person originating a packet, it is the critical building block for subsequent forensic analysis, investigation, and correlation, as it provides a beachhead onto the physical scene of the crime. We presuppose that sites

a The city of Calcutta first systematically used human fingerprints for criminal records in 1897, followed by Scotland Yard in Britain in 1901.

with sufficient risk and/or value at stake will check such signatures, associate them with higher-level transactions, and log them for enough time to cover their risk. Building such a capability is straightforward using conventional digital signatures and some form of public-key infrastructure, albeit with some performance cost—and one significant drawback: complete lack of privacy.

Privacy requirements. The approach we've just described would allow anyone receiving such a packet to attribute its physical origin. There is also a history of vigorous opposition to such capabilities. For example, in early 1999, Intel Corporation announced that new generations of its popular Pentium microprocessors would include a new feature—the Processor Serial Number (PSN)—a per-processor unique identifier intended as a building block for future security applications. Even though this feature was completely passive, public-interest groups quickly identified potential risks to privacy stemming from an available globally unique identifier. In April 2000, Intel abandoned plans to include PSN in future versions of its microprocessors.

We thus posit another critical requirement for a practical forensics tool:

Privacy. To balance the need for forensic attribution against the public's interest in privacy, packet signatures must be non-identifying, in a strong sense, to an unprivileged observer. Moreover, the signatures must not serve as an identifier (even an opaque one). As such, distinct packets sent from the same source must carry different signatures. Internet users should have at least the same expectation of anonymity they have today, except for authorized investigations.

A strawman solution to this problem is to digitally sign each packet using a per-source key that is in turn escrowed with a trusted third party. Indeed, the ill-fated Clipper chip used such an approach. If a single third party is not widely trusted (likely, given past experience), then the scheme may accommodate multiple third parties responsible for different sets of machines and/or a secret sharing approach in which multiple third parties collaborate to gener-



Unlike physical evidence (such as fingerprints and DNA), digital objects are, *prima facie*, not unique.



ate the keying material to validate the origin of a signature; for example, in the U.S., both the Department of Justice and the American Civil Liberties Union might be required to agree an investigation is warranted. However, this approach also involves a critical vulnerability. Since, by design, a normal observer cannot extract information from a packet signature, nothing prevents adversaries from incorrectly signing their packets, or random “signatures.” Any attempt at post-hoc authentication is useless. Thus, to be practical, our attribution architecture is motivated by a final requirement:

Attributability. To enforce the attribution property, any observer on the network must be empowered to verify a packet signature—to prove that the packet could be attributed if necessary, though the process of performing the proof must not reveal any information about the physical originator itself. This requirement has a natural fate-sharing property, since choosing to verify a packet is made by the recipient with a future interest in having an attribution capability.

Remaining challenges. As important as our design goals are, so, too, are our non-goals—what we do not attempt to accomplish. For one, our work is not designed to address IP-address spoofing. While there is operational value in preventing spoofing or allowing easier filtering of DDoS attacks, the virtual nature of IP addresses makes them inherently ill-suited for forensic purposes. More significant, our work is limited to attributing the physical machine that sent a particular packet and not necessarily the complete causal chain of events leading to the packet being generated. This distinction is common to most kinds of forensic investigations (such as unraveling offshore shell accounts in forensic accounting or insider communication in securities fraud investigations) but can manifest easily in the Internet context; for example, an attack might be laundered through one or more intermediate nodes, either as part of a legitimate anonymizing overlay network (such as Tor) or via proxies installed on compromised hosts, botnets, or other intermediaries.

In practice, unraveling complex dependencies is simultaneously criti-

cally important and fundamentally challenging. Previous work explored how such “stepping-stone” relationships may be inferred in the network,²⁰ and a similar approach—attributing each causal link hop-by-hop—could be employed with our architecture as well. However, unambiguously establishing such causality is not possible at the network layer alone and will ultimately require both host support and, inevitably, manual investigation. While we have a vision for how such host services should be structured, it represents future work beyond the scope of this article.

Architecture

While these requirements appear challenging, there is a well-known cryptographic tool—a group signature¹¹—that unties this particular Gordian knot. Here, we briefly review the properties of group signatures, describing their basic application to forensic packet attribution and the design issues resulting from this architecture.

Group signatures. A group signature provides the property that if a member of a group signs a message, anyone can verify that the signature was created by a group member but cannot determine which one, without the cooperation of a privileged third party, the group manager.

We describe group signatures using the formalized model of Bellare et al.,⁵ as well as a definitional extension due to Boneh et al.⁸ Specifically, a group-signature scheme assumes a group manager and a group of n unprivileged members, denoted $1, 2, \dots, n$. The group manager has a secret key msk , each group member $i \in \{1, \dots, n\}$ has its own secret signing key $\text{sk}[i]$, and there is a single public signature-verification key pk .

The group manager uses a `KeyGen` operation to create pk , msk , sk , distributing them appropriately. Subsequently, if a group member i uses its secret key $\text{sk}[i]$ to `Sign` a message m and gets back a signature σ , anyone with access to the global signature-verification key pk can `Verify` that (m, σ) is a valid message-signature pair under the secret key of a group member. The group manager can use msk to recover the identity i of the signer using the `Open` operation.

The two principal security properties for group signature schemes—full-anonymity and full-traceability—imply other properties, including unforgeability, exculpability, and framing-resistance.⁵ A group signature scheme is CCA-fully anonymous if a set of colluding members cannot learn information about the signers’ identity i , even when adversaries are allowed to `Open` the signatures of all the messages besides the target message-signature pair. A group-signature scheme is fully traceable if a set of colluding members cannot create a valid message-signature pair (m, σ) that the group manager cannot trace back to one of the colluding parties; that is, either `Verify`(pk, m, σ) fails, meaning the signature is invalid, or `Open`(msk, m, σ) returns the identity of one of the colluding members.

Basic packet attribution. We apply group signatures to our problem in the following way: Each machine is a member of some group and provided with a secret signing key. Exactly how groups are constructed, and who is authorized to open resulting signatures, is very much a policy issue, but one pragmatic approach is that each computer manufacturer defines a group across the set of machines it sells. Being manufacturer-centered is particularly appealing because it sidesteps the key distribution problem, as manufacturers now commonly include trusted platform modules that encode unique cryptographic information in each of their machines. Moreover, a tamper-resistant implementation is useful for preventing theft of a machine’s signing key. This approach would also imply the manufacturer would act as group manager in any investigation, execute `Open` under subpoena, or escrow its msk (or shares thereof) to third parties.

Given a secret signing key, each machine uses it to sign the packets it sends. This signature covers all non-variant protocol fields and payload data. The name of the group and the per-packet signature are included in a special packet header field that is part of the network layer. Any recipient can examine the header, using `Verify` to validate that a packet was correctly signed by a member of the associated group (and hence could be authenticated by the group manager). The verify step does not require protected key

material and, by virtue of fate sharing, need not be part of the trusted computing base.

Design. An implementation of group-signature-based packet attribution must address several other challenges before deployment is practical:

Replay. The basic approach we’ve outlined does not prevent an adversary from replaying messages sent (and signed) by other legitimate parties or shuffling the order in which a node receives the messages from other legitimate parties. In some cases, such replayed packets are immediately discarded by the receiving protocol stack or application (such as due to misaligned sequence numbers or routing information). On the other hand, an adversary might be able to mount an untraceable DoS attack or maliciously change application behaviors by replaying or shuffling others’ packets over time. We therefore desire some mechanism to bind these packets to a particular point in time.

A possible solution would involve having the sender include a monotonically increasing counter in each packet and the receiver discard any packets with duplicate sources and counters. However, the naive implementation of such an approach might require the receiver to maintain the list of source-counter pairs (such as through reboots). We assume loosely synchronized clocks; the signer includes the current time in each outgoing packet, and the receiver validates freshness directly. To handle jitter and network delays, as well as possible inconsistencies among different devices’ perception of time, one might employ a hybrid approach, including both a counter and the time in each packet.

Revocation. To ensure that verifiable packets are attributable back to a single physical machine, we assume the group-signature secret keys are stored in tamper-resistant hardware and not copyable to other devices. However, we anticipate that some secret signing keys will inevitably be compromised. There are two general frameworks for revoking these secret signing keys (due to Ateniese² and Camenisch and Lysyanskaya⁹). Since most parties in our system are both signers and verifiers, we adopt the Camenisch-Lysyanskaya⁹ approach in which secret keys are re-

voked by globally updating the group public key and locally updating each unrevoked party's secret signing key. In this scheme, the verifiers need not maintain a list of individual revocations, but public-key updates must be applied universally to ensure all subsequent signatures can be verified.

Middlebox modification. Middleboxes, like network address translators (NATs), create a conundrum. In our architecture, senders sign all non-volatile contents of outgoing packets, including source address. Thus, any packets traversing a NAT will no longer verify, as their contents have been changed. While some might consider this a shortcoming, it is a requirement of true attribution; signers can attest only to contents they transmitted. The only other option in our framework is to deem the source address volatile and exclude it from the packet signature. To do so would imply the source address has no significance beyond being a routing locator, though, unfortunately, this is not the case in today's Internet, where end hosts use source addresses to demultiplex incoming connections, as well as to associate flows, with the appropriate IPsec associations.

This tension has been observed many times in the past, yielding two architecturally pure alternatives: future Internet architectures can either remove end host dependence on IP source addresses or make the presence of middleboxes explicit. For the time being, deployments requiring NAT-like functionality must make a trade-off between deployability and completeness, choosing between removing source addresses from the signature—thereby limiting the scope of the attribution—and encapsulating the original, signed packets in an IP-in-IP tunnel, exposing the middlebox to the receiver.

Related questions concern virtualization technologies. In a virtualized environment, the underlying machine must sign packets. Though technically feasible, we do not expand on specific approaches here.

Clue

We developed the Clue prototype to explore the systems issues related to implementing a real-world group-signature scheme. Clue uses Stanford's

Pairing-Based Cryptography (PBC) Library¹⁵ for group-signature operations that in turn uses the GNU Multiple Precision arithmetic library (GMP). To explore group signatures in the context of real network packets, we implemented Clue as a module in the Click Modular Router.¹³ As PBC and GMP are designed as user-mode libraries, Clue employs Click as a user-mode process rather than as a Linux or BSD kernel module. While this architecture incurs some performance penalty on its own, the cryptographic operations dominate the user-mode transitions in practice, and the user-mode penalty does not interfere with fundamental system-design issues.

Figure 1 outlines the packet trailer used by the current prototype. The Clue module is mostly a straightforward packet-transformation element. When signing, the module performs the following four tasks:

- ▶ Collects nonvolatile elements of an IP packet;
- ▶ Adds an 8B local NTP-derived timestamp to implement replay detection;
- ▶ Feeds the resulting data as input to the group-signature library to generate a signature; and
- ▶ Appends the signature (and additional optimization information) to the original packet, adjusting the IP length field accordingly.

Tasks like recalculating checksums are left to other, standard Click elements in the pipeline performing these functions. Similarly, when verifying, the module performs the following five tasks:

- ▶ Validates a packet's freshness from its timestamp;
- ▶ Collects the nonvolatile elements

of an IP packet;

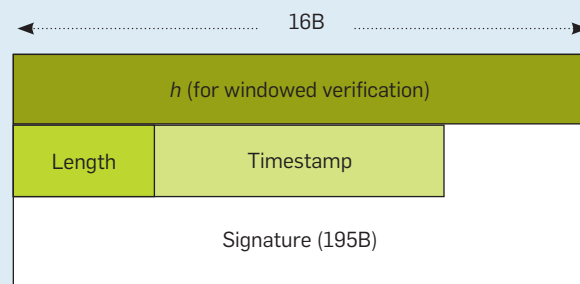
- ▶ Strips the Clue trailer from the end of the packet;
- ▶ Feeds the resulting data and signature to the group signature library; and
- ▶ Pushes the original packet to one of two output ports, depending whether verification was successful.

Clue implements Boneh et al.'s revocation scheme,⁸ polling a well-known revocation service. As might be expected, cryptographic operation overhead can be high and dominate most performance measures. While we have little doubt that more-efficient group-signature schemes will emerge with faster implementations and that hardware implementation provides more capacity, here we focus on the optimization opportunities arising from the interaction between the dynamics of network protocols themselves and the underlying cryptographic primitives. We first describe the particular group-signature construction we use and then a series of optimizations we've implemented.

BBS short-group signatures. The Clue prototype uses the Boneh et al.⁸ short-group-signature scheme, which exhibits comparatively short signatures relative to group-signature schemes based on the Strong-RSA assumption of Baric and Pfitzmann.⁴ We also refine the BBS group-signature scheme for use with Clue's optimizations. The following paragraph summarizes the basic BBS scheme at a level sufficient to understand our optimizations:

The BBS Sign algorithm (on input a group public key pk , the signer's secret key $sk[i]$, and a message m) first obtains a source of randomness ν , derives

Figure 1. Clue packet-trailer format; shaded fields are explained in the section on optimizations.



values for the variables $T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5$ from σ and pk , then computes the value c as $c \leftarrow H(m, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5)$ where $\leftarrow$ denotes assignment from right to left, and H is a hash function; for the security proofs, BBS model H is a random oracle.^{6,8} The signing algorithm then outputs

$$\sigma \leftarrow (T_1, T_2, T_3, c, s_{\alpha}, s_{\beta}, s_{\chi}, s_{\delta 1}, s_{\delta 7}), \quad (1)$$

where $s_{\alpha}, s_{\beta}, s_{\chi}, s_{\delta 1}, s_{\delta 7}$ are functions of c, ν , and $\text{sk}[i]$. The BBS Verify algorithm, on input a group public key pk , a message m , and a signature $\sigma = (T_1, T_2, T_3, c, s_{\alpha}, s_{\beta}, s_{\chi}, s_{\delta 1}, s_{\delta 7})$, derives $R'_1, R'_2, R'_3, R'_4, R'_5$ from pk and σ , computes c' as $c' \leftarrow H(m, T_1, T_2, T_3, R'_1, R'_2, R'_3, R'_4, R'_5)$, accepting the signature as valid exactly when $c = c'$. None of Clue's optimizations or extensions modify the BBS KeyGen or Open algorithms; we therefore do not survey their details here.

Optimizations. The following optimizations exploit artifacts of the BBS scheme itself, as well as properties of network protocols and clients; some of these optimizations may be of independent interest.

Precomputation (for sender). Clue is able to take advantage of client workloads to improve the overhead of Sign in the sending critical path. The Sign operation has two components, computing the T_j and R_j values, independent of packet data, using these values to sign a packet. The T_j and R_j computation step by far dominates the overhead of Sign. If Clue takes the T_j and R_j computation out of the critical sending path by precomputing them, Clue can greatly improve the throughput of using Sign. Most client workloads consist of applications with low average sending rates (such as email, Web browsing, and remote login), allowing signature precomputation to overlap I/O. Indeed, over long time scales, the CPU load of clients and servers alike is dominated by idle time—an effect further magnified by multicore processors. Thus, periods of idleness can be exploited to buffer signature precursors for subsequent periods of activity.

Windowed verification (for receiver). Clue takes advantage of the streaming nature of network protocols like TCP to amortize verification over multiple packets of data to reduce the overhead of Verify in the receive critical path.

Deriving the R'_j values from pk and σ creates a significant fixed overhead for Verify independent of the amount of signed data. When using Verify on the receiver, the attribution layer can accumulate a window of packets (such as a flight of TCP segments) and verify them all together to amortize per-packet verification overhead. We stress that the signer signs every window of k packets, even overlapping windows, and that the verifier has the option of either verifying the packets individually or verifying any window of its choosing. However, this verification optimization slightly increases the length of a signature.

To accommodate this scenario, we modify the BBS scheme as follows: Using our modified scheme, a verifier can choose to verify the signature on the j -th packet P_j in isolation (such as when no other packets are waiting to be verified or when there is packet loss) or verify in batch the signature on a window of k packets $P_{j-k+1}, \dots, P_j$. Clue achieves this goal by, on the signing side, first hashing the initial $k-1$ packets $P_{j-k+1}, \dots, P_{j-1}$ to a value h , then signing $h||P_j$ as before finally including h in the resulting signature tuple; here $||$ denotes string concatenation, and the hash function to compute h is $H' \neq H$, and P_j is implicitly prefixed with a fixed-width length field. To avoid trivial hash collisions in h , when hashing the packets $P_{j-k+1}, \dots, P_{j-1}$, Clue also prepends each packet with a 4B length field, then concatenates the resulting length fields and packets together. Including h in the signature allows the receiver to verify the signature over the j -th packet P_j in isolation (by verifying the signature over $h||P_j$). To verify the signature over the entire window $P_{j-k+1}, \dots, P_j$, the receiver first recomputes h .

In the Clue prototype the window size k is a parameter provided to the IP layer. We modified our TCP implementation to adaptively set k to match the sender's congestion window. This setting maximizes performance, as it reflects the largest number of packets that can be amortized together without expecting a packet loss (losing the benefit of amortized verification).

Asynchronous verification (for receiver). The Clue prototype can also overlap computation with network delay to reduce protocol serialization

with verification; for example, rather than wait until Verify completes on a TCP packet before sending an ACK, TCP can first optimistically send an ACK back to the sender to overlap the ACK with the Verify computation. Implementing this feature is inherently a layer violation since the Clue prototype allows TCP ACK processing to proceed independent of IP layer verification, but Clue prevents unverified data packets from being passed to the application.

Incremental verification (for receiver). Given the computational costs associated with the Verify algorithm, under some circumstances (such as DoS attacks), Clue may wish to be able to quickly reject packets that might not be attributable. While the Clue prototype cannot completely erase the cost for verification, it can decrease the amount of time to reject a nonverifiable packet by a factor of approximately three, at the expense of increased signature sizes; we make Verify incrementally verifiable. The average time to process and reject a nonattributable packet decreases, though the time to accept a legitimate packet remains essentially unchanged.

Clue's incrementally verifiable version of the BBS group signature scheme builds on our earlier observation that (1) the bulk of the computation in Verify is spent computing $R'_1, \dots, R'_5$, and (2) an implementation can derive $R'_1, \dots, R'_5$ in parallel. Technically, we change Equation 1 to

$$\sigma \leftarrow (T_1, T_2, T_3, c, s_{\alpha}, s_{\beta}, s_{\chi}, s_{\delta 1}, s_{\delta 7}, R_1, R_2, R_3, R_4, R_5).$$

We then revise the Verify algorithm to, on input a signature σ , set $c'' \leftarrow H(m, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5)$, and immediately reject if $c'' \neq c$. The modified verification algorithm would then derive the variables $R'_1, R'_2, R'_3, R'_4, R'_5$ from pk and $T_1, T_2, T_3, c, s_{\alpha}, s_{\beta}, s_{\chi}, s_{\delta 1}, s_{\delta 7}$ in random order, immediately rejecting if $R'_j \neq R_j$. Finally, the modified algorithm would accept the signature as valid, since failure to reject implies $c = H(m, T_1, T_2, T_3, R'_1, R'_2, R'_3, R'_4, R'_5)$.

Other potential optimizations. A large class of related optimizations relax security guarantees in exchange for performance; for example, the receiver could randomly verify a packet with

probability $1/n$ for some n . However, we have explicitly chosen not to explore such optimizations at this time, since our goal here is to examine an extreme point in the design space—where the attributability of each and every packet is enforced. For the same reasons, we have not yet explored protocol-specific fate-sharing optimizations (such as only signing and verifying TCP SYN packets). Such optimizations could dramatically reduce overhead, albeit in exchange for some increased risk of nonattributability (such as via TCP connection hijacking).

Evaluation

Having described and evaluated Clue's security properties, we now turn to quantifying the overhead of our implementation's basic security operations and measure its effect on TCP performance. We present these benchmarks to demonstrate the feasibility of our approach. We do not evaluate all aspects of Clue, leaving full consideration of revocation to future work.

Our user-level software prototype provides acceptable performance when using the optimizations described earlier. Clue adds about 30ms of end-to-end delay to sending a packet. For interactive applications like SSH, this extra delay is insignificant to users. Clue achieves a bulk TCP throughput of 17.5Mbps, which is greater than that enjoyed by the average wide-area Internet user. A typical Internet user browsing the Web using Clue would experience roughly the same performance as without using Clue.

Experimental setup. For our experiments, we use three hosts in a sender-delay-receiver configuration. The delay host runs Linux 2.6 with a hardware configuration of dual-2.8GHz Pentiums with 2GB of RAM and the NIST Net emulation package¹⁰ to introduce link delay. The sender is a dual-3.4GHz Pentium with 4GB of RAM, and the receiver runs dual-3.0GHz Pentiums with 16GB of RAM. Both sender and receiver run the Click-based implementation of Clue (discussed earlier) over Linux 2.6, using the default values for send and receive buffer sizes.

For all experiments, we use the d277699-175-167 parameter file prepackaged with PBC, yielding a group signature scheme with strength rough-



Without a plausible threat of accountability, the normal social processes that disincentivize criminal behavior cannot function.



ly equivalent to a standard 1,024-bit RSA signature.⁸ The BBS scheme outputs signatures 195B long using this parameter file.

Microbenchmarks. We start by measuring the overhead of the basic cryptographic operations Sign and Verify and their variants, as described earlier. The table here outlines the average time taken across 100 iterations of these operations on the receiver. The first column of results for “1 packet” are overheads when executing on a single 1,277B packet as input; we chose 1,277, since the combination of the packet, 195B for basic BBS signature, 8B for timestamp, and an extra 20B for windowed signing optimization yield a 1,500B packet. The second column, “8 packets,” are results with eight packets as input; one of Clue's optimizations amortizes overhead across windows of multiple packets. In either case, the per-packet overhead is sufficiently small (10ms–30ms total) to be unnoticeable in interactive traffic but substantial enough to have a significant effect on bulk TCP performance.

The precomputation optimization for the sender separates signature computation from signing the packet. The “precomp sign” result measures the step that remains on the critical path—signing using a set of precomputed values—and shows that almost all overhead of Sign comes from generating message-independent cryptographic values (the “precomputation” step), not from computing the message-dependent part of the signature or signing the packet itself. In our bulk-transfer experiments, we show that removing signature computation from the critical path of sending packets results in significant increase in throughput. Similarly, the row labeled “verify” represents the average time to verify a single signed packet of the same size; in our Clue implementation, verification is about 2.5x slower than signing.

The remaining two rows in the table measure the performance of Clue's incremental verification scheme designed to defend against the flood of invalid packets described earlier. The “incremental verify” row is the time required to verify a valid packet signature using this scheme, essentially identical to the original verification

operation, introducing negligible CPU overhead in the common case. In contrast, “corrupted incremental verify” measures the average time required to reject a corrupted signature. Using incremental verification Clue achieves a 70% reduction in overhead over the original scheme.

The only significant difference between the eight-packet times and the single-packet times occurs when signing a packet using precomputed values arising as a result of hashing the extra data in the additional packets. Note, however, that this cost is still roughly two orders of magnitude less than any other operation, so we do not observe any additional impact on bulk throughput. As a result, amortizing the attribution operations over multiple pack-

ets is a key mechanism for reducing receive overhead. In the experiments discussed in the following section, we show that large windows combined with precomputed signatures can dramatically improve performance over basic `Sign` and `Verify` alone.

TCP throughput. Bulk TCP throughput is an important performance metric for many Internet applications. Experimentally, our goal is to evaluate the effect of attribution on TCP throughput in our Clue prototype. We measure the TCP throughput performance of the attribution implementation relative to various baseline configurations across a range of network round-trip times (RTTs). In Clue, the implementation of attribution achieves a throughput within a factor of 1.2 of a Click for-

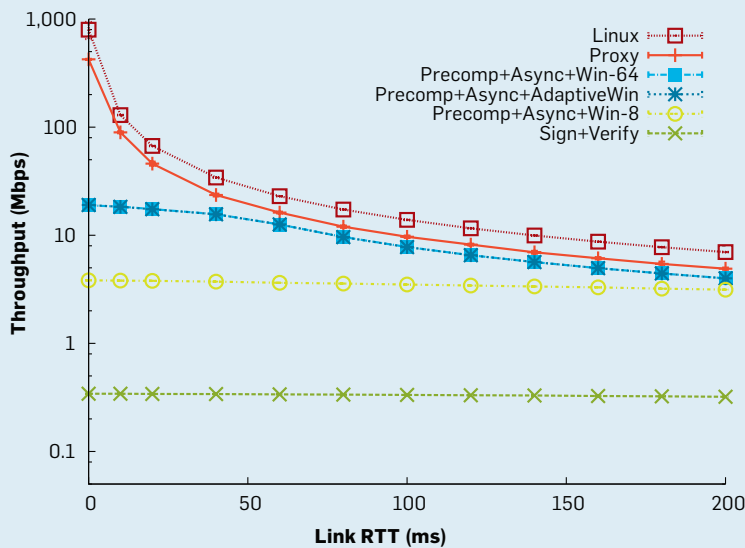
warder at typical Internet RTTs. While privacy-preserving attribution has a non-negligible effect on bulk throughput on today’s client systems, the cost is not prohibitive and will continue decreasing over time, as CPU performance increases more quickly than typical Internet bandwidth.

We conduct `ttcp` benchmarks between the sender and receiver, requiring them to forward traffic through a delay host. For every test configuration, we run each individual transfer for at least 20 seconds. We require the sender to transfer all its data before it closes the connection, timing the transfer from when the sender connects to the receiver to when the sender receives the `FIN` from the receiver. Figure 2 outlines the results of the experiments for a number of configurations. We vary the roundtrip time (RTT) between sender and receiver on the x-axis and plot the throughput achieved using the `ttcp` application benchmark on the y-axis; note the y-axis is a log scale, each point is the average of five runs, and error bars show the standard deviation.

As an upper bound, the “Linux” curve plots the forwarding rate of the default Linux networking stack on our hardware. To provide a more realistic baseline for our Clue implementation, we also show the performance of an unmodified user-level Click installation (“Proxy”); Click forwards packets received on its input to its output without processing. The difference between “Proxy” and “Linux” shows the overhead of interposing in the network stack at user level, including copying overhead when crossing the kernel boundary. However, an optimized, kernel-level packet-attribution implementation need not suffer this overhead. Though not shown, we also measured the performance of the provided Click IPsec module, finding its performance indistinguishable from the “Proxy” configuration.

The “Sign+Verify” line corresponds to the baseline performance of Clue using individual `Sign` and `Verify` on each IP datagram. Given the times required for `Sign` and `Verify`, as shown in the table, one would expect the 29ms required for the `Verify` operation to limit long-term bulk throughput to a maximum of 0.35Mbps. Not surpris-

Figure 2. TCP throughput performance for combined optimizations; y-axis is in log scale.



Overheads of cryptographic operations for both one- and eight-packet windows.

	1 packet (in ms)	8 packets (in ms)
sign	6.17	6.19
precomputed sign	0.022	0.058
precomputation	6.14	6.14
verify	15.7	15.7
incremental verify	15.6	16.3
corrupted incremental verify	4.85	4.83

ing, Clue's implementation of the default "Sign+Verify" attribution process restricts bulk TCP throughput to approximately 0.33Mbps independent of the RTT.

The poor performance of "Sign+Verify" motivates the optimizations described earlier. While precomputation dramatically decreases the overhead at the sender, it has only modest effect in isolation on TCP throughput, as performance is still receiver-limited. Similarly, asynchronous verification allows the receiver to issue ACKs immediately, but the potential for improvement is bounded by the effective decrease in flow RTT. Indeed, precomputation and asynchronous verification are most effective when combined with windowed verification and has the potential to move the performance bottleneck back to the sender.

The line in Figure 2 labeled "Precomp+Async+Win-8" is the performance of the Clue prototype when combining the three optimizations while using a fixed window size of eight packets. In theory, the larger the window size, the less overhead verification imposes. Indeed, progressively increasing the window size continues to increase throughput performance—to a point; most benefits are achieved with a window of 64 packets, as indicated by the line "Precomp+Async+Win-64" in Figure 2, exceeding 17.5Mbps at 20ms. Recall that windowed verification proceeds only in the absence of loss; if a packet is lost in a window, the remaining packets must be verified individually, negating any potential for improvement. Hence, our Clue implementation dynamically adjusts the window size to match the sender's TCP congestion window. The "Precomp+Async+AdaptiveWin" line in Figure 2 shows its performance approaches the baseline for all but the smallest RTTs; at an RTT of 80ms—typical of TCP connections on the Internet¹⁸—this combination achieves a throughput of 9.6Mbps, within a factor of 1.2 of "Proxy" itself, and exceeds the capacity of most consumer broadband links.

Conclusion

Much of the Internet's success can

be attributed to its minimalist architecture. However, the related architectural freedoms also represent ripe vulnerabilities for adversaries trying to exploit the network to their own ends. Chief among them is the lack of accountability for user actions. Without a plausible threat of accountability, the normal social processes that disincentivize criminal behavior cannot function. We suggest modifying the Internet architecture to proactively enable network forensics while preserving the privacy of network participants under normal circumstances.

Our approach ensures: authorized parties can determine the physical identity of hardware originating any given IP packets; no other party can determine the identity of the originating physical hardware; and all network participants can simultaneously verify that a packet is well-formed and attributable by the trusted authority. While still some distance from being practicable, our technique may be a viable and promising foundation for future research. A separate research strand must still consider the broader contextual issues surrounding such a solution, ranging from the role of manufacturers to international law.

Acknowledgments

We thank Hovav Shacham of the University of California, San Diego, for advice and comments. This work is funded in part by National Science Foundation grants CNS-0627157 and CNS-0722031. 

References

- Andersen, D., Balakrishnan, H., Feamster, N., Koponen, T., Moon, D., and Shenker, S. Accountable Internet Protocol. In *Proceedings of the ACM SIGCOMM Conference* (Seattle, Aug. 19–21). ACM Press, New York, 339–350.
- Ateniese, G., Tsudik, G., and Song, D. Quasi-efficient revocation of group signatures. In *Financial Cryptography*, M. Blaze, Ed. (Southampton, Bermuda, Mar. 11–14). Springer-Verlag, Berlin, 2002, 183–197.
- Aucsmith, D. The digital crime scene: A software perspective. In *Proceedings of the CyberCrime and Digital Law Enforcement Conference* (New Haven, CT, Mar. 26–28, 2004).
- Baric, N. and Pfizmann, B. Collision-free accumulators and fail-stop signature schemes without trees. In *Advances in Cryptology EUROCRYPT '97*, W. Fumy, Ed. (Konstanz, Germany, May 11–15). Springer-Verlag, Berlin, 1997, 480–494.
- Bellare, M., Micciancio, D., and Warinschi, B. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *Advances in Cryptology EUROCRYPT '03*, E. Biham, Ed. (Warsaw, May 4–8). Springer-Verlag, Berlin, 2003, 614–629.
- Bellare, M. and Rogaway, P. Random oracles are

- practical: A paradigm for designing efficient protocols. In *Proceedings of the ACM Conference on Computer and Communications Security* (Fairfax, VA, Nov. 3–5). ACM Press, New York, 1993, 62–73.
- Bellare, M. Security problems in the TCP/IP protocol suite. *ACM SIGCOMM Computer Communication Review* 19, 2 (Apr. 1989), 32–48.
 - Boneh, D., Boyen, X., and Shacham, H. Short group signatures. In *Advances in Cryptology CRYPTO 2004*, M. Franklin, Ed. (Santa Barbara, CA, Aug. 15–19). Springer-Verlag, Berlin, 2004, 41–55.
 - Camenisch, J. and Lysyanskaya, A. Dynamic accumulators and applications to efficient revocation of anonymous credentials. In *Advances in Cryptology CRYPTO 2002*, M. Yung, Ed. (Santa Barbara, CA, Aug. 18–2). Springer-Verlag, Berlin, Germany, 2002, 61–76.
 - Carson, M. and Santay, D. NIST Net: A Linux-based network-emulation tool. *ACM SIGCOMM Computer Communication Review* 33, 3 (July 2003), 111–126.
 - Chaum, D. and van Heyst, E. Group signatures. In *Advances in Cryptology EUROCRYPT '91*, D.W. Davies, Ed. (Santa Barbara, CA, Apr. 8–11). Springer-Verlag, Berlin, 1991, 257–265.
 - International Telecommunications Union. *Traceback Use Cases and Requirements*; <http://politechbot.com/docs/itu.traceback.use.cases.requirements.091108.txt>
 - Kohler, E., Morris, R., Chen, B., Jannotti, J., and Kaashoek, M.F. The Click modular router. *ACM Transactions on Computer Systems* 18, 3 (Aug. 2000), 263–297.
 - Liu, X., Yang, X., Weatherall, D., and Anderson, T. Efficient and secure source authentication with packet passports. In *Proceedings of the Second Workshop on Steps to Reducing Unwanted Traffic on the Internet* (San Jose, CA, July 7). USENIX, Berkeley, CA, 2006.
 - Lynn, B. Pairing-Based Cryptography Library. Stanford University, Palo Alto, CA, 2006; <http://crypto.stanford.edu/pbc/>
 - Moore, D., Voelker, G.M., and Savage, S. Inferring Internet denial of service activity. In *Proceedings of the USENIX Security Symposium* (Washington, D.C., Aug. 13–17). USENIX, Berkeley, CA, 2001, 9–22.
 - Savage, S., Wetherall, D., Karlin, A.R., and Anderson, T. Practical network support for IP traceback. In *Proceedings of the ACM SIGCOMM Conference* (Stockholm, Aug. 28–Sept. 1), ACM Press, New York, 2000, 295–306.
 - Shalunov, S. *TCP Over WAN Performance Tuning and Troubleshooting*, 2005; <http://shlang.com/writing/tcp-perf.html>
 - Snoeren, A.C., Partridge, C., Sanchez, L.A., Jones, C.E., Tchakountio, F., Schwartz, B., Kent, S.T., and Strayer, W.T. Single-packet IP traceback. *IEEE/ACM Transactions on Networking* 10, 6 (Dec. 2002), 721–734.
 - Zhang, Y. and Paxson, V. Detecting stepping stones. In *Proceedings of the USENIX Security Symposium* (Denver, Aug. 14–17). USENIX, Berkeley, CA, 2000, 171–184.

Mikhail Afanasyev (mafanasyev@gmail.com) is a postdoctoral fellow in the Autonomous Systems Laboratory of the Australian Commonwealth Scientific and Research Organization (CSIRO), Brisbane, Australia.

Tadayoshi Kohno (yoshi@cs.washington.edu) is an assistant professor in the Computer Science and Engineering Department of the University of Washington, Seattle, WA.

Justin Ma (jtma@eecs.berkeley.edu) is a postdoctoral scholar in the AMP Lab of the University of California, Berkeley.

Nicholas Murphy (nmurphy@eecs.harvard.edu) is a doctoral candidate in the School of Engineering and Applied Sciences of Harvard University, Cambridge, MA.

Stefan Savage (savage@cs.ucsd.edu) is a professor in the Computer Science and Engineering Department of the University of California, San Diego.

Alex C. Snoeren (snoeren@cs.ucsd.edu) is an associate professor in the Computer Science and Engineering Department of the University of California, San Diego.

Geoffrey M. Voelker (voelker@cs.ucsd.edu) is a professor in the Computer Science and Engineering Department of the University of California, San Diego.

In contrast to popular belief, proving termination is not always impossible.

BY BYRON COOK, ANDREAS PODELSKI,
AND ANDREY RYBALCHENKO

Proving Program Termination

THE PROGRAM TERMINATION problem, also known as the uniform halting problem, can be defined as follows:

Using only a finite amount of time, determine whether a given program will always finish running or could execute forever.

This problem rose to prominence before the invention of the modern computer, in the era of Hilbert's *Entscheidungsproblem*:^a the challenge to formalize all of mathematics and use algorithmic means to determine the validity of all statements. In hopes of either solving Hilbert's challenge, or showing it impossible, logicians began to search for possible instances of undecidable problems. Turing's proof³⁸ of termination's undecidability is the most famous of those findings.^b

The termination problem is structured as an infinite

set of queries: to solve the problem we would need to invent a method capable of accurately answering either "terminates" or "doesn't terminate" when given any program drawn from this set. Turing's result tells us that any tool that attempts to solve this problem will fail to return a correct answer on at least one of the inputs. No number of extra processors nor terabytes of storage nor new sophisticated algorithms will lead to the development of a true oracle for program termination.

Unfortunately, many have drawn too strong of a conclusion about the prospects of automatic program termination proving and falsely believe we are always unable to prove termination, rather than more benign consequence that we are unable to always prove termination. Phrases like "but that's like the termination problem" are often used to end discussions that might otherwise have led to viable partial solutions for real but undecidable problems. While we cannot ignore termination's undecidability, if we develop a slightly modified problem statement we can build useful tools. In our new problem statement we will still require that a termination proving tool always return answers that are correct, but we will not necessarily require an answer. If the termination prover cannot prove or disprove termination, it should return "unknown."

Using only a finite amount of time, determine whether a given program will always finish running or could execute forever, or return the answer "unknown."

» key insights

- For decades, the same method was used for proving termination. It has never been applied successfully to large programs.
- A deep theorem in mathematical logic, based on Ramsey's theorem, holds the key to a new method.
- The new method can scale to large programs because it allows for the modular construction of termination arguments.

^a In English: "decision problem."

^b There is a minor controversy as to whether or not Turing proved the undecidability in³⁸. Technically he did not, but termination's undecidability is an easy consequence of the result that is proved. A simple proof can be found in Strachey.³⁶



This problem can clearly be solved, as we could simply always return “unknown.” The challenge is to solve this problem while keeping the occurrences of the answer “unknown” to within a tolerable threshold, in the same way that we hope Web browsers will *usually* succeed to download Web pages, although we know they will sometimes fail. Note that the principled use of unknown in tools attempting to solve undecidable or intractable problems is increasingly common in computer science; for example, in program analysis, type systems, and networking.

In recent years, powerful new termination tools have emerged that return “unknown” infrequently enough that they are useful in practice.³⁵ These termination tools can automatically prove or disprove termination of many famous complex examples such as Ackermann’s function or McCarthy’s 91 function as well as moderately sized industrial examples such as Windows device drivers. Furthermore, entire families of industrially useful termination-like properties—called *liveness properties*—such as “Every call to lock is eventually followed by a call to unlock” are now automatically

provable using termination proving techniques.^{12,29} With every month, we now see more powerful applications of automatic termination proving. As an example, recent work has demonstrated the utility of automatic termination proving to the problem of showing concurrent algorithms to be non-blocking.²⁰ With further research and development, we will see more powerful and more scalable tools.

We could also witness a shift in the power of software, as techniques from termination proving could lead to tools for other problems of equal difficulty. Whereas in the past a software developer hoping to build practical tools for solving something related to termination might have been frightened off by a colleague’s retort “but that’s like the termination problem,” perhaps in the future the developer will instead adapt techniques from within modern termination provers in order to develop a partial solution to the problem of interest.

The purpose of this article is to familiarize the reader with the recent advances in program termination proving, and to catalog the underlying techniques for those interested in

adapting the techniques to other domains. We also discuss current work and possible avenues for future investigation. Concepts and strategies will be introduced informally, with citations to original papers for those interested in more detail. Several sidebars are included for readers with backgrounds in mathematical logic.

Disjunctive Termination Arguments

Thirteen years after publishing his original undecidability result, Turing proposed the now classic method of proving program termination.³⁹ His solution divides the problem into two parts:

Termination argument search: Find a potential termination argument in the form of a function that maps every program state to a value in a mathematical structure called a well-order. We will not define well-orders here, the reader can assume for now that we are using the natural numbers (a.k.a. the positive integers).

Termination argument checking: Proves the termination argument to be valid for the program under consideration by proving that result of the function decreases for every possible program transition. That is, if f is the termination argument and the program can transition from some state s to state s' , then $f(s) > f(s')$.

(Readers with a background in logic may be interested in the formal explanation contained in the sidebar here.)

A well-order can be thought of as a terminating program—in the example of the natural numbers, the program is one that counts from some initial value in the natural numbers down to 0. Thus, no matter which initial value is chosen the program will still terminate. Given this connection between well-orders and terminating programs, in essence Turing is proposing that we search for a map from the program we are interested in proving terminating into a program known to terminate such that all steps in the first program have analogous steps in the second program. This map to a well-order is usually called a *progress measure* or a *ranking function* in the literature. Until recently, all known methods of proving termination were in essence minor variations on the original technique.

Turing’s Classic Method and Disjunctive Well-Foundedness

Formally proving program termination amounts to proving the program’s transition relation R to be well-founded. If $(S, \geq)$ is a well-order then $>$ is a well-founded relation. Furthermore, any map f into S defines a well-founded relation, by lifting $>$ via f , that is, $\{(s, t) \mid f(s) > f(t)\}$. Turing’s method³⁹ of proving a program’s transition relation R well-founded amounts to finding a map f into a well-order, which defines a termination argument $T = \{(s, t) \mid f(s) > f(t)\}$. To prove the validity of T we must show $R \subseteq T$. From the well-foundedness of T and the fact that every sub-relation of a well-founded relation is well-founded follows that R is well-founded.

In this article we are using the phrase disjunctive termination argument to refer to a disjunctively well-founded transition invariant.³¹ This is a finite union $T_1 \cup \dots \cup T_n$ of well-founded relations that contains R , which is the transitive closure of the transition relation of the program, as a superset, such as, $R^+ \subseteq T_1 \cup \dots \cup T_n$.

Usually, each $T_1, \dots, T_n$ will be constructed as above via some map into a well-order. Note that the non-reflexive transitive closure (the $+$ in R^+) is crucial. It is not sufficient to show that $R \subseteq T_1 \cup \dots \cup T_n$, as the union of well-founded relations is not guaranteed to be well-founded. It is the transitive closure that makes checking the subset inclusion more difficult in practice.

The recent approaches for proving termination for general programs^{3,4,9,12,14,32} are based on the proof rule of disjunctively well-founded transition invariants. The proof rule itself is based on Ramsey’s theorem,³⁴ and it has been developed in the effort to give a logical foundation to the termination analysis based on size-change graphs.²⁴ The principle expressed by the proof rule appears implicitly already in previously developed termination algorithms for rewrite systems and logic and functional programs, see refs^{10,15,17,24}.

The problem with Turing's method is that finding a single, or monolithic, ranking function for the whole program is typically difficult, even for simple programs. In fact, we are often forced to use ranking functions into well-orders that are much more complex than the natural numbers. Luckily, once a suitable ranking function has been found, checking validity is in practice fairly easy.

The key trend that has led toward current progress in termination proving has been the move away from the search for a single ranking function and toward a search for a set of ranking functions. We think of the set as a choice of ranking functions and talk about a disjunctive termination argument. This terminology refers to the proof rule of disjunctively well-founded transition invariants.³¹ The recent approaches for proving termination for general programs^{3,4,9,12,14,26,32} are based on this proof rule. The proof rule itself is based on Ramsey's theo-

rem,³⁴ and it has been developed in the effort to give a logical foundation to the termination analysis based on size-change graphs.²⁴

The principle it expresses appears implicitly in previously developed termination algorithms for rewrite systems, logic, and functional programs, see refs ^{10,15,17,24}.

The advantage to the new style of termination argument is that it is usually easier to find, because it can be expressed in small, mutually independent pieces. Each piece can be found separately or incrementally using various known methods for the discovery of monolithic termination arguments. As a trade-off, when using a disjunctive termination argument, a more difficult validity condition must be checked. This difficulty can be mitigated thanks to recent advances in assertion checking tools (as discussed in a later section).

Example using a monolithic termination argument. Consider the example

code fragment in Figure 1. In this code the collection of user-provided input is performed via the function `input()`. We will assume the user always enters a new value when prompted. Furthermore, we will assume for now that variables range over possibly negative integers with arbitrary precision (that is, mathematical integers as opposed to 32-bit words, 64-bit words, and so on). Before reading further, please answer the question: "Does this program terminate, no matter what values the user gives via the `input()` function?" The answer is given below.^c

Using Turing's traditional method we can define a ranking function from program variables to the natural numbers. One ranking function that will work is $2x + y$, though there are many others. Here we are using the formula $2x + y$ as shorthand for a function that takes a program configuration as its input and returns the natural number computed by looking up the value of x in the memory, multiplying that by 2 and then adding in y 's value—thus $2x + y$ represents a mapping from program configurations to natural numbers. This ranking function meets the constraints required to prove termination: the valuation of $2x + y$ when executing at line 9 in the program will be strictly one less than its valuation during the same loop iteration at line 4. Furthermore, we know the function always produces natural numbers (thus it is a map into a well-order), as $2x + y$ is greater than 0 at lines 4 through 9.

Automatically proving the validity of a monolithic termination argument like $2x + y$ is usually easy using tools that check verification conditions (for example, Slam²). However, as mentioned previously, the actual search for a valid argument is famously tricky. As an example, consider the case in Figure 2, where we have replaced the command " $y := y + 1$;" in Figure 1 with " $y := input();$ ". In this case no function into the natural numbers exists that suffices to prove termination; instead we must resort to a *lexicographic* ranking function (a ranking function into ordinals, a more advanced well-order than the naturals).

Figure 1. Example program.

```

1      x := input();
2      y := input();
3      while x > 0 and y > 0 do
4          if input() = 1 then
5              x := x - 1;
6              y := y + 1;
7          else
8              y := y - 1;
9          fi
10     done

```

User-supplied inputs are gathered via calls to the function `input()`. We assume that the variables range over integers with arbitrary precision (in other words, not 64-bit or 32-bit integers). Assuming that the user always eventually enters in a value when prompted via `input()`, does the program terminate for all possible user-supplied inputs? (The answer is provided in a footnote below.)

Figure 2. Example program.

```

1      x := input();
2      y := input();
3      while x > 0 and y > 0 do
4          if input() = 1 then
5              x := x x 1;
6              y := input();
7          else
8              y := y - 1;
9          fi
10     done

```

This program is similar to Figure 1 where the command " $y := y + 1$;" replaced with " $y := input();$ ". No ranking function into the natural numbers exists that can prove the termination of this program.

^c The program does terminate.

Example using a disjunctive termination argument. Following the trend toward the use of disjunctive termination arguments, we could also prove the termination of Figure 1 by defining an argument as the unordered finite collection of measures x and y . The termination argument in this case should be read as:

x goes down by at least 1 and is larger than 0.
or
 y goes down by at least 1 and is larger than 0

We have constructed this termination argument with two ranking functions: x and y . The use of “*or*” is key: the termination argument is modular because it is easy to enlarge using additional measures via additional uses of “*or*.” As an example, we could enlarge the termination argument by adding “*or* $2w - y$ goes down by at least 1 and is greater than 1,000.” Furthermore, as we will discuss later, independently finding these pieces of the termination argument is easier in practice than finding a single monolithic ranking function.

The expert reader will notice the relationship between our disjunctive termination argument and complex lexicographic ranking functions. The advantage here is that we do not need to find an order on the pieces of the argument, thus making the pieces of the argument independent from one another.

The difficulty with disjunctive termination arguments in comparison to monolithic ones is that they are more difficult to prove valid: for the benefit of modularity we pay the price in the fact that the termination arguments must consider the transitions in all possible loop unrollings and not just single passes through a loop. That is to say: the disjunctive termination argument must hold not only between the states before and after any single iteration of the loop, but before and after *any number* of iterations of the loop (one iteration, two iterations, three iterations, and so on). This is a much more difficult condition to automatically prove. In the case of Figure 1 we can prove the more complex condition using techniques described later.

Note that this same termination argument now works for the tricky pro-

gram in Figure 2, where we replaced “ $y := y + 1$ ” with “ $y := \text{input}();$ ” On every possible unrolling of the loop we will still see that either x or y has gone down and is larger than 0.

To see why we cannot use the same validity check for disjunctive termination arguments as we do for monolithic ones, consider the slightly modified example in Figure 3. For every single iteration of the loop it is true that ei-

ther x goes down by at least one and x is greater than 0 or y goes down by at least one and y is greater than 0. Yet, the program does not guarantee termination. As an example input sequence that triggers non-termination, consider 5, 5, followed by 1, 0, 1, 0, 1, 0, ... If we consider *all possible unrollings* of the loop, however, we will see that after two iterations it is possible (in the case that the user supplied the

Figure 3. Another example program.

```

1      x := input();
2      y := input();
3      while x > 0 and y > 0 do
4          if input() = 1 then
5              x := x - 1;
6              y := y + 1;
7          else
8              x := x + 1;
9              y := y - 1;
10         fi
11     done

```

Does it terminate for all possible user-supplied inputs?

Figure 4. Example program with an assertion statement in line 3.

```

1      if y ≥ 1 then
2          while x > 0 do
3              assert (y ≥ 1);
4              x := x - y;
5              done
6          fi

```

Figure 5. Encoding of termination argument validity.

```

1      copied := 0;
2      x := input();
3      y := input();
4      while x > 0 and y > 0 do
5          if copied = 1 then
6              assert (oldx ≥ x + 1 and oldx > 0);
7          elsif input() = 1 then
8              copied := 1;
9              oldx := x;
10             oldy := y;
11         fi
12         if input() = 1 then
13             x := x - 1;
14             y := y + 1;
15         else
16             y := y - 1;
17         fi
18     done

```

Encoding of termination argument validity using the program from Figure 1 and the termination argument “ x goes down by at least one and is larger than 0.” The black code comes directly from Figure 1. The code in red implements the encoding of validity with an assertion statement.

inputs 1 and 0 during the two loop iterations) that neither x nor y went down, and thus the disjunctive termination argument is not valid for the program in Figure 3.

Argument Validity Checking

While validity checking for disjunctive termination arguments is more difficult than checking for monolithic arguments, we can adapt the problem statement such that recently developed tools for proving the validity of assertions in programs (such as Slam²).

An assertion statement can be put in a program to check if a condition is true. For example, `assert( $y \geq 1$ );` checks that $y \geq 1$ after executing the command. We can use an assertion checking tool to formally investigate at compile time whether the conditions passed to assertion statements always evaluate to true. For example, most assertion checking tools will be able to prove the `assert` statement at line 3 in Figure 4 never fails. Note that compile-time assertion checking is itself an undecidable problem, although it is technically in an easier class of difficulty than termination.^d

The reason that assertion checking is so important to termination is the validity of disjunctive termination arguments can be encoded as an assertion statement, where the statement fails only in the case that the termination argument is not valid. Once we are given an argument of the form T_1 or T_2 or ... or T_n , to check validity we simply want to prove the following statement:

Each time an execution passes through one state and then through another one, T_1 or T_2 or ... or T_n holds between these two states. That is, there does not exist a pair of states, one being reachable from the other, possibly via the unrolling of a loop, such that neither T_1 nor T_2 nor ... nor T_n holds between this pair of states.

This statement can be verified a program transformation where we introduce new variables into the program to record the state before the unrolling of the loop and then use

^d Checking validity of an assertion statement is an undecidable but co-recursively enumerable problem, whereas termination is neither r.e. nor co-r.e. problem.

Implementation Strategies

Here, we give a brief summary of implementation strategies based on disjunctive termination arguments deployed by the recent termination checkers:

Refinement:^{9,14} In Cook et al.,¹⁴ the termination argument begins with $\emptyset$. We first attempt to prove that $R_+ \subseteq \emptyset$. When this proof fails, rank function synthesis is applied to the witness, thus giving a refinement T_1 to the argument, which is then rechecked $R_+ \subseteq \emptyset \cup T_1$. This process is repeated until a valid argument is found or a real counterexample is found.

In Chawdhary et al.,⁹ the termination argument T is constructed following the structure of the transition relation $R = R_1 \cup \dots \cup R_m$ by using a ranking function synthesis procedure, which is used to compute a well-founded overapproximation $WF(X)$ of a binary relation X . The initial candidate $T = WF(R_1) \cup \dots \cup WF(R_m)$ is extended with $WF(WF(R_i) \circ R_j)$ and so on until the fixpoint is reached.

Variance analysis:^{3,32} As described in some detail in this article, the approach from Berdine et al.³ and Podolski et al.³² uses program transformations and abstract interpretation for invariants to compute an overapproximation $T_1; T_2, \dots, T_n$ such that $R_+ \subseteq T_1 \cup T_2 \dots \cup T_n$. It then uses rank function synthesis to check that each T_i is well-founded.

In contrast to the refinement-based methods, variance analysis always terminates, but may return “don’t know” in cases when a refinement-based method succeeds.

an assertion statement to check the termination argument always holds between the current state and the recorded state. If the assertion checker can prove the `assert` cannot fail, it has proved the validity of the termination argument. We can use encoding tricks to force the assertion checker to consider *all* possible unrollings.

Figure 5 offers such an example, where we have used the termination argument “ x goes down by at least one and x is greater than 0” using the en-

coding given in Cook et al.¹⁴ The new code (introduced as a part of the encoding) is given in red, whereas the original program from Figure 1 is in black. We make use of an extra call to `input()` to decide when the unrolling begins. The new variables `oldx` and `oldy` are used for recording a state. Note that the assertion checker must consider all values possibly returned by `input()` during its proof, thus the proof of termination is valid for any starting position. This has the effect of

Figure 6. Encoding of termination argument validity using previous program.

```

1      copied := 0;
2      x := input();
3      y := input();
4      while x > 0 and y > 0 do
5          if copied = 1 then
6              assert( (oldx ≥ x + 1 and oldx > 0)
7                  or
8                      (oldy ≥ y + 1 and oldy > 0)
9                  );
10         elsif input() = 1 then
11             copied := 1;
12             oldx := x;
13             oldy := y;
14         fi
15         if input() = 1 then
16             x := x - 1;
17             y := y + 1;
18         else
19             y := y - 1;
20         fi
21     done

```

Encoding of termination argument validity using the program from Figure 1 and the termination argument “ x goes down by at least one and is larger than 0 or y goes down by at least one and is larger than 0.” The black code comes directly from Figure 1. The code in red implements the encoding of validity with an assertion statement.

considering any possible unrolling of the loop. After some state has been recorded, from this point out the termination argument is checked using the recorded state and the current state. In this case the assertion can fail, meaning that the termination argument is not valid.

If we were to attempt to check this condition in a naïve way (for example, by simply executing the program) we would never find a proof for all but the most trivial of cases. Thus, assertion checkers must be cleverly designed to find proofs about all possible executions without actually executing all of

the paths. A plethora of recently developed techniques now make this possible. Many recent assertion checkers are designed to produce a path to a bug in the case that the assertion statement cannot be proved. For example, a path leading to the assertion failure is $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 16 \rightarrow 17 \rightarrow 4 \rightarrow 5 \rightarrow 6$. This path can be broken into parts, each representing different phases of the execution: the prefix-path $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ is the path from the program initial state to the recorded state in the failing pair of states. The second part of the path $4 \rightarrow 5 \rightarrow \dots \rightarrow 6$ represents how

we reached the current state from the recorded one. That is: this is the unrolling found that demonstrates that the assertion statement can fail. What we know is that the termination argument does not currently cover the case where this path is repeated forever.

See Figure 6 for a version using the same encoding, but with the valid termination argument:

x goes down by at least 1 and is larger than 0
or
 y goes down by at least 1 and is larger than 0.

This assertion cannot fail. The fact that it cannot fail can be proved by a number of assertion verification tools.

Finding Termination Arguments

We have examined how we can check a termination argument's validity via a translation to a program with an assertion statement. We now discuss known methods for finding monolithic termination arguments.

Rank function synthesis. In some cases simple ranking functions can be automatically found. We call a ranking function simple if it can be defined by a linear arithmetic expression (for example, $-3x = -2y + 100$). The most popular approach for finding this class of ranking function uses a result from Farkas¹⁶ together with tools for solving linear constraint systems. (See Colón and Sipma¹¹ or Polelski and Rybalchevko³⁰ for examples of tools using Farkas' lemma.) Many other approaches for finding ranking functions for different classes of programs have been proposed (see refs^{1,6-8,19,37}). Tools for the synthesis of ranking functions are sometimes applied directly to programs, but more frequently they are used (on small and simplified program fragments) internally within termination proving tools for suggesting the single ranking functions that appear in a disjunctive termination argument.

Termination analysis. Numerous approaches have been developed for finding disjunctive termination arguments in which—in effect—the validity condition for disjunctive termination arguments is almost guaranteed to hold by construction. In some cases—for example, Berdine et al.³—to prove termination we need only check

Figure 7. Program prepared for abstract interpretation.

```

1      copied := 0;
2      x := input();
3      y := input();
4      while x > 0 and y > 0 do
5          if copied = 1 then
6              skip;
7          elsif input() = 1 then
8              copied := 1;
9              oldx := x;
10             oldy := y;
11         fi
12         if input() = 1 then
13             x := x - 1;
14             y := y + 1;
15         else
16             y := y - 1;
17         fi
18     done

```

Figure 8. Example C loop over a linked-list data-structure with fields next and data.

```

c = head;
while (c != NULL) {
    if (c ->next != NULL && c ->next ->data == 5) {
        t = c ->next;
        c->next = c ->next ->next;
        free(t);
    }
    c = c->next;
}

```

Figure 9. Example program illustrating nontermination.

```

1      x := 10;
2      while x > 9 do
3          x := x - 232;
4      done

```

Example program demonstrating nontermination when variables range over fixed-width numbers. The program terminates if x ranges over arbitrary size integers, but repeatedly visits the state where $x = 10$ in the case that x ranges over 32-bit unsigned numbers.

that the argument indeed represents a set of measures. In other cases, such as Lee et al.²⁴ or Manolios and Vroon,²⁶ the tool makes a one-time guess as to the termination argument and then checks it using techniques drawn from abstract interpretation.

Consider the modified program in Figure 7. The termination strategy described in Berdine et al.³ and Podelski and Rybalchenko³² essentially builds a program like this and then applies a custom program analysis to find the following candidate termination argument:

```
(copied ≠ 1) or
(oldx ≥ x + 1, oldx > 0, oldy
> 0, x ≥ 0, y > 0) or
(oldx ≥ x, oldy ≥ y + 1, oldx
> 0, oldy > 0, x > 0, y ≥ 0)
```

for the program at line 4—meaning we could pass this complex expression to the assertion at line 4 in Figure 7 and know that the assertion cannot fail. We know this statement is true of any unrolling of the loop in the original Figure 1. What remains is to prove that each piece of the candidate argument represents a measure that decreases—here we can use rank function synthesis tools to prove that $\text{oldx} > x + 1$ and $\text{oldx} > 0 \dots$ represents the measure based on x . If each piece between the `ors` in fact represents a measure (with the exception of `copied ≠ 1` which comes from the encoding) then we have proved termination.

One difficulty with this style of termination proving is that, in the case that the program doesn't terminate, the tools can only report "unknown," as the techniques used inside the abstract interpretation tools have lost so much detail that it is impossible to find a non-terminating execution from the failed proof and then prove it non-terminating. The advantage when compared to other known techniques is it is much faster.

Finding arguments by refinement. Another method for discovering a termination argument is to follow the approach of Cook et al.¹⁴ or Chawdhary et al.⁹ and search for counterexamples to (possibly invalid) termination arguments and then refine them based on new ranking functions found via the counterexamples.

In recent years, powerful new termination tools have emerged that return "unknown" infrequently enough that they are useful in practice.

Recall Figure 5, which encoded the invalid termination argument for the program in Figure 1, and the path leading to the failure of the assertion: is $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 11 \rightarrow 12 \rightarrow 16 \rightarrow 17 \rightarrow 4 \rightarrow 5 \rightarrow 6$. Recall this path represents two phases of the program's execution: the path to the loop, and some unrolling of the loop such that the termination condition doesn't hold. In this case the path $4 \rightarrow 5 \rightarrow \dots 6$ represents how we reached the second failing state from the first. This is a counterexample to the validity of the termination argument, meaning that the current termination argument does not take this path and others like it into account.

If the path can be repeated forever during the program's execution then we have found a real counterexample. Known approaches (for example, Gupta et al.²¹) can be used to try and prove this path can be repeated forever. In this case, however, we know that the path cannot be repeated forever, as y is decremented on each iteration through the path and also constrained via a conditional statement to be positive. Thus this path is a spurious counterexample to termination and can be ruled out via a refinement to the termination argument. Again, using rank function synthesis tools we can automatically find a ranking function that demonstrates the spuriousness of this path. In this case a rank function synthesis tool will find y , meaning that the reason this path cannot be repeated forever is that " y always goes down by at least one and is larger than 0." We can then refine the current termination argument used in Figure 5:

```
x goes down by at least 1 and is larger than 0
with the larger termination argument:
x goes down by at least 1 and is larger than 0
or
y goes down by at least 1 and is larger than 0
```

We can then check the validity of this termination argument using a tool such as IMPACT on the program in Figure 6. IMPACT can prove this assertion never fails, thus proving the termination of the program in Figure 1.

Further Directions

With fresh advances in methods for proving the termination of sequen-

tial programs that operate over mathematical numbers, we are now in the position to begin proving termination of more complex programs, such as those with dynamically allocated data structures, or multithreading. Furthermore, these new advances open up new potential for proving properties beyond termination, and finding conditions that would guarantee termination. We now discuss these avenues of future research and development in some detail.

Dynamically allocated heap. Consider the C loop in Figure 8, which walks down a list and removes links with data elements equaling 5. Does this loop guarantee termination? What termination argument should we use?

The problem here is that there are no arithmetic variables in the program from which we can begin to construct an argument—instead we would want to express the termination argument over the lengths of paths to NULL via the next field. Furthermore, the programmer has obviously intended for this loop to be used on acyclic singly linked lists, but how do we know that the lists pointed to by head will always be acyclic? The common solution to these problems is to use *shape analysis* tools (which are designed to

automatically discover the shapes of data-structures) and then to create new auxiliary variables in the program that track the sizes of those data structures, thus allowing for arithmetic ranking functions to be more easily expressed (examples include refs^{4,5,25}). The difficulty with this approach is that we are now dependent on the accuracy and scalability of current shape analysis tools—to date the best known shape analysis tool⁴⁰ supports only lists and trees (cyclic and acyclic, singly and doubly linked) and scales only to relatively simple programs of size less than 30,000 LOC. Furthermore, the auxiliary variables introduced by methods such as Magill et al.²⁵ sometimes do not track enough information in order to prove termination (for example, imagine a case with lists of lists in which the sizes of the nested lists are important). In order to improve the state of the art for termination proving of programs using data structures, we must develop better methods of finding arguments over data structure shapes, and we must also improve the accuracy and scalability of existing shape analysis tools.

Bit vectors. In the examples used until now we have considered only variables that range over mathematical numbers. The reality is that most

programs use variables that range over fixed-width numbers, such as 32-bit integers or 64-bit floating-point numbers, with the possibility of overflow or underflow. If a program uses only fixed-width numbers and does not use dynamically allocated memory, then termination proving is decidable (though still not easy). In this case we simply need to look for a repeated state, as the program will diverge if and only if there exists some state that is repeated during execution. Furthermore, we cannot ignore the fixed-width semantics, as overflow and underflow can cause non-termination in programs that would otherwise terminate, an example is included in Figure 9. Another complication when considering this style of program is that of bit-level operations, such as left- or right-shift.

Binary executables. Until now we have discussed proving termination of programs at their source level, perhaps in C or Java. The difficulty with this strategy is the compilers that then take these source programs and convert them into executable artifacts can introduce termination bugs that do not exist in the original source program. Several potential strategies could help mitigate this problem: We might try to prove termination of the executable binaries instead of the source level programs, or we might try to equip the compiler with the ability to prove that the resulting binary program preserves termination, perhaps by first proving the termination of the source program and then finding a map from the binary to the source-level program and proving that the composition with the source-level termination argument forms a valid termination argument for the binary-level program.

Non-linear systems. Current termination provers largely ignore non-linear arithmetic. When non-linear updates to variables do occur (for example $x := y * z$), current termination provers typically treat them as if they were the instruction $x := \text{input}()$. This modification is sound—meaning when the termination prover returns the answer “terminating,” we know the proof is valid. Unfortunately, this method is not precise: the treatment of these commands can lead to the result “unknown” for programs

Figure 10. Example of multi-threaded terminating producer/consumer program.

<pre> 1 while x > 0 do 2 x := x - 1; 3 lock (lck) 4 b := x; 5 unlock (lck) 6 done </pre>	<pre> 1 while y > 0 do 2 lock (lck) 3 y := b; 5 unlock (lck) 6 done </pre>
---	--

To prove that the thread on the left terminates we must assume that the thread on the right always calls unlock when needed. To prove that the thread on the right always calls unlock when needed, we must prove that the thread on the left always calls unlock when needed, and so on.

Figure 11. Collatz program.

```

1  while x > 1 do
2      if x is divisible by 2 then
3          x := x/2;
4      else
5          x := 3x + 1;
6      fi
7  done

```

We assume that x ranges over all natural numbers with arbitrary precision (that is, neither 64-bit vectors nor 32-bit vectors). A proof of this program's termination or non-termination is not known.

that actually terminate. Termination provers are also typically unable to find or check non-linear termination arguments (x^2 , for example) when they are required. Some preliminary efforts in this direction have been made,^{1,6} but these techniques are weak. To improve the current power of termination provers, further developments in non-linear reasoning are required.

Concurrency. Concurrency adds an extra layer of difficulty when attempting to prove program termination. The problem here is that we must consider all possible interactions between concurrently executing threads. This is especially true for modern fine-grained concurrent algorithms, in which threads interact in subtle ways through dynamically allocated data structures. Rather than attempting to explicitly consider all possible interleavings of the threads (which does not scale to large programs) the usual method for proving concurrent programs correct is based on *rely-guarantee* or *assume-guarantee* style of reasoning, which considers every thread in isolation under assumptions on its environment and thus avoids reasoning about thread interactions directly. Much of the power of a rely-guarantee proof system (such as Jones²² and Misra and Chandy²⁸) comes from the cyclic proof rules, where we can assume a property of the second thread while proving property of the first thread, and then assume the recently proved property of the first thread when proving the assumed property of the second thread. This strategy can be extended to liveness properties using induction over time, for example, Gotsman et al.²⁰ and McMillan.²⁷

As an example, consider the two code fragments in Figure 10. Imagine that we are executing these two fragments concurrently. To prove the termination of the left thread we must prove that it does not get stuck waiting for the call to **lock**. To prove this we can assume the other thread will always eventually release the lock—but to prove this of the code on the right we must assume the analogous property of the thread on the left, and so on. In this case we can certainly just consider all possible interleavings of the threads, thus turning the concurrent program into a sequential model



With fresh advances in methods for proving the termination of sequential programs that operate over mathematical numbers, we are now in the position to begin proving termination of more complex programs.



representing its executions, but this approach does not scale well to larger programs. The challenge is to develop automatic methods of finding non-circular rely-guarantee termination arguments. Recent steps²⁰ have developed heuristics that work for non-blocking algorithms, but more general techniques are still required.

Advanced programming features. The industrial adoption of high-level programming features such as virtual functions, inheritance, higher-order functions, or closures make the task of proving industrial programs more of a challenge. With few exceptions (such as Giesl et al.¹⁸), this area has not been well studied.

Untyped or dynamically typed programs also contribute difficulty when proving termination, as current approaches are based on statically discovering data-structure invariants and finding arithmetic measures in order to prove termination. Data in untyped programs is often encoded in strings, using pattern matching to marshal data in and out of strings. Termination proving tools for JavaScript would be especially welcome, given the havoc that nonterminating JavaScript causes daily for Web browsers.

Finding preconditions that guarantee termination. In the case that a program does not guarantee termination from all initial configurations, we may want to automatically discover the conditions under which the program does guarantee termination. That is, when calling some function provided by a library: what are the conditions under which the code is guaranteed to return with a result? The challenge in this area is to find the right precondition: the empty precondition is correct but useless, whereas the weakest precondition for even very simple programs can often be expressed only in complex domains not supported by today's tools. Furthermore, they should be computed quickly (the weakest precondition expressible in the target logic may be too expensive to compute). Recent work has shown some preliminary progress in this direction.^{13,33}

Liveness. We have alluded to the connection between liveness properties and the program termination problem. Formally, liveness properties expressed in temporal logics can

be converted into questions of *fair termination*—termination proving were certain non-terminating executions are deemed unfair via given fairness constraints, and thus ignored. Current tools, in fact, either perform this reduction, or simply require the user to express liveness constraints directly as the set of fairness constraints.^{12,29} Neither approach is optimal: the reduction from liveness to fairness is inefficient in the size of the conversion, and fairness constraints are difficult for humans to understand when used directly. An avenue for future work would be to directly prove liveness properties, perhaps as an adaption of existing termination proving techniques.

Dynamic analysis and crash dumps for liveness bugs. In this article we have focused only on *static*, or *compile-time*, proof techniques rather than techniques for diagnosing divergence during execution. Some effort has been placed into the area of automatically detecting deadlock during execution time. With new developments in the area of program termination proving we might find that automatic methods of discovering *livelock* could also now be possible. Temporary modifications to scheduling, or other techniques, might also be employed to help programs not diverge even in cases where they do not guarantee termination or other liveness properties. Some preliminary work has begun to emerge in this area (see Julia et al.²³) but more work is needed.

Scalability, performance, and precision. Scalability to large and complex programs is currently a problem for modern termination provers—current techniques are known, at best, to scale to simple systems code of 30,000 lines of code. Another problem we face is one of precision. Some small programs currently cannot be proved terminating with existing tools. Turing's undecidability result, of course, states that this will always be true, but this does preclude us from improving precision for various classes of programs and concrete examples. The most famous example is that of the Collatz' problem, which amounts to proving the termination or non-termination of the program in Figure 11. Currently no proof of this program's termination behavior is known.

Conclusion

This article has surveyed recent advances in program termination proving techniques for sequential programs, and pointed toward ongoing work and potential areas for future development. The hope of many tool builders in this area is that the current and future termination proving techniques will become generally available for developers wishing to directly prove termination or liveness. We also hope that termination-related applications—such as detecting livelock at runtime or Wang's tiling problem—will also benefit from these advances.

Acknowledgments

The authors would like to thank Lucas Bourdeaux, Abigail See, Tim Harris, Ralf Herbrich, Peter O'Hearn, and Hongseok Yang for their reading of early drafts of this article and suggestions for improvement. 

References

- Babic, D., Hu, A.J., Rakamaric, Z., and Cook, B. Proving termination by divergence. In *SEFM*, 2007.
- Ball, T., Bounimova, E., Cook, B., Levin, V., Lichtenberg, J., McGarvey, C., Ondrusek, B., Rajamani, S.K. and Ustuner, A. Thorough static analysis of device drivers. In *Proceedings of EuroSys*, 2006.
- Berdine, J., Chawdhary, A., Cook, B., Distefano, D. and O'Hearn, P. Variance analyses from invariance analyses. In *Proceedings of POPL*, 2007.
- Berdine, J., Cook, B., Distefano, D. and O'Hearn, P. Automatic termination proofs for programs with shape-shifting heaps. In *Proceedings of CAV*, 2006.
- Bouajjani, A., Bozga, M., Habermehl, P., Iosif, R., Moro, Y. and Vojnar, T. Programs with lists are counter automata. In *Proceedings of CAV*, 2006.
- Bradley, A., Manna, Z. and Sipma, H. Termination of polynomial programs. In *Proceedings of VMCAI*, 2005.
- Bradley, A., Manna, Z. and Sipma, H.B. Linear ranking with reachability. In *Proceedings of CAV*, 2005.
- Bradley, A., Manna, Z. and Sipma, H.B. The polyranking principle. In *Proceedings of ICALP*, 2005.
- Chawdhary, C., Cook, B., Gulwani, S., Sagiv, M. and Yang, H. Ranking abstractions. In *Proceedings of ESOP*, 2008.
- Codish, M., Genaim, S., Bruynooghe, M., Gallagher, J. and Vanhoof, W. One loop at a time. In *Proceedings of WST*, 2003.
- Colón, M. and Sipma, H. Synthesis of linear ranking functions. In *Proceedings of TACAS*, 2001.
- Cook, B., Gotsman, A., Podelski, A., Rybalchenko, A. and Vardi, M. Proving that programs eventually do something good. In *Proceedings of POPL*, 2007.
- Cook, B., Gulwani, S., Lev-Ami, T., Rybalchenko, A. and Sagiv, M. Proving conditional termination. In *Proceedings of CAV*, 2008.
- Cook, B., Podelski, A. and Rybalchenko, A. Termination proofs for systems code. In *Proceedings of PLDI*, 2006.
- Dershowitz, N., Lindenstrauss, N., Sagiv, Y. and Serebrenik, A. A general framework for automatic termination analysis of logic programs. *Appl. Algebra Eng. Commun. Comput.*, 2001.
- Farkas, J. Über die Theorie der einfachen Ungleichungen. *Journal für die reine und angewandte Mathematik*, 1902.
- Geser, A. Relative termination. PhD dissertation, 1990.
- Giesl, J., Swiderski, S., Schneider-Kamp, P. and Thiemann, R. Automated termination analysis for Haskell: From term rewriting to programming languages. In *Proceedings of RTA*, 2006.

- Giesl, J., Thiemann, R., Schneider-Kamp, P. and Falke, S. Automated termination proofs with AProVE. In *Proceedings of RTA*, 2004.
- Gotsman, A., Cook, B., Parkinson, M. and Vafeiadis, V. Proving that non-blocking algorithms don't block. In *Proceedings of POPL*, 2009.
- Gupta, A., Henzinger, T., Majumdar, R., Rybalchenko, A., and Xu, R. Proving non-termination. In *Proceedings of POPL*, 2008.
- Jones, C.B. Tentative steps toward a development method for interfering programs. *ACM Trans. Program. Lang. Syst.*, 1983.
- Julia, H., Tralamazza, D., Zamfir, C. and Candea, G. Deadlock immunity: Enabling systems to defend against deadlocks. In *Proceedings of OSDI*, 2008.
- Lee, C.S., Jones, N.D. and Ben-Amram, A.M. The size-change principle for program termination. In *Proceedings of POPL*, 2001.
- Magill, S., Berdine, J., Clarke, E. and Cook, B. Arithmetic strengthening for shape analysis. In *Proceedings of SAS*, 2007.
- Manolios, P. and Vroon, D. Termination analysis with calling context graphs. In *Proceedings of CAV*, 2006.
- McMillan, K.L. Circular compositional reasoning about liveness. In *Proceedings of CHARME*, 1999.
- Misra, J. and Chandy, K.M. Proofs of networks of processes. *IEEE Trans. Software Eng.*, 1981.
- PNueli, A., Podelski, A., and Rybalchenko, A. Separating fairness and well-foundedness for the analysis of fair discrete systems. In *Proceedings of TACAS*, 2005.
- Podelski, A. and Rybalchenko, A. A complete method for the synthesis of linear ranking functions. In *Proceedings of VMCAI*, 2004.
- Podelski, A. and Rybalchenko, A. Transition invariants. In *Proceedings of LICS*, 2004.
- Podelski, A. and Rybalchenko, A. Transition predicate abstraction and fair termination. In *Proceedings of POPL*, 2005.
- Podelski, A., Rybalchenko, A., and Wies, T. Heap assumptions on demand. In *Proceedings of CAV*, 2008.
- Ramsey, F. On a problem of formal logic. *London Math. Soc.*, 1930.
- Stix, G. Send in the Terminator. *Scientific American* (Nov. 2006).
- Strachey, C. An impossible program. *Computer Journal*, 1965.
- Tiwari, A. Termination of linear programs. In *Proceedings of CAV*, 2004.
- Turing, A. On computable numbers, with an application to the Entscheidungsproblem. *London Mathematical Society*, 1936.
- Turing, A. Checking a large routine. In Report of a Conference on High Speed Automatic Calculating Machines, 1949.
- Yang, H., Lee, O., Berdine, J., Calcagno, C., Cook, B., Distefano, D. and O'Hearn, P. Scalable shape analysis for systems code. In *Proceedings of CAV*, 2008.

Byron Cook is a Principal Researcher at Microsoft's research laboratory at Cambridge University, and a professor of computer science at Queen Mary, University of London, England.

Andreas Podelski is a professor of computer science at the University of Freiburg, Germany.

Andrey Rybalchenko is a professor of computer science at the Technische Universität München, Germany.

research highlights

P. 100

**Technical
Perspective
Complex Financial
Products:
*Caveat Emptor***

By David C. Parkes

P. 101

**Computational Complexity
and Information Asymmetry
in Financial Products**

By Sanjeev Arora, Boaz Barak, Markus Brunnermeier, and Rong Ge

P. 108

**Technical
Perspective
Images Everywhere
Looking for Models**

By Guillermo Sapiro

P. 109

**Self-Similarity-based
Image Denoising**

By Antoni Buades, Bartomeu Coll, and Jean-Michel Morel

Technical Perspective

Complex Financial Products: *Caveat Emptor*

By David C. Parkes

THE FLOW OF capital in the financial industry relies on the packaging of assets into products that can be reliably valued and then sold to global investors. For example, many home mortgages were packaged into products known as *Collateralized Debt Obligations* (CDOs) in the run-up to the sub-prime mortgage crisis of 2007. An investor in a CDO buys the rights to a share of the principal and interest payments collected from homeowners. By pooling assets and promising to pass along payments before making payments to other investors, new financial products offering lower risk than the underlying assets can be constructed. CDOs are examples of *financial derivatives*, with a value that depends on the underlying assets—mortgages in this case—with which they are linked.

These kinds of complex financial products are the cause célèbre of the financial crisis, and many have called for their regulation or even elimination. In the following paper, Arora, Barak, Brunnermeier, and Ge provide new insight into the problem: a complexity-theoretic explanation for how sellers can hide bad assets in these derivatives. Even when buyers are fully informed, with correct beliefs about the probability with which underlying mortgages are likely to default, sellers can package a disproportionate number of bad assets into some products, and do so without detection. The reason is the intractability of checking whether or not this manipulation has occurred. By focusing on this missing angle of computational complexity, this paper starts to bridge the gap between the common view that derivatives can be rigged and a viewpoint from economics that this is impossible when buyers are fully informed. Computationally bounded buyers may end up significantly overpaying, and a trustworthy seller cannot even prove that financial products have not been rigged.

To understand the reason to sell derivatives in the first place, we can

consider Akerlof's famous "lemons problem." Suppose that 80% of second-hand cars are good, and worth \$1,000 to buyers, while the rest are lemons and worth \$0. Without the ability for a seller to credibly signal the quality of a car, buyers will only pay \$800 and trades of good cars by sellers with values in the range [\$800, \$1,000] are forfeited. If all sellers of good cars want close to \$1,000 then the effect of information asymmetry between buyers and sellers is much worse—only lemons remain in the market and there is complete market collapse! Still, a seller with 100 cars, each correctly known by a buyer to be a lemon with probability 0.2, can make a new deal: the right to use up to 80 of the cars. Because it is highly likely that at least this many cars will be good, this deal can be priced at about \$80,000, around the price at which it would trade without information asymmetry. The same thing happens in a simple model of CDOs, in which a seller packages assets into a *single* derivative that can be accurately priced and sold.

Now consider a seller with 1,000,000 cars, with the cars partitioned into classes, and the association with a class known to buyers. Each class is a "lemons class" with some probability, in which case it contains only lemons, and otherwise is a "good class" and contains a mixture of good cars and lemons. The probability of a lemons class, and the fraction of lemons in a good class, is known to buyers. The seller again constructs deals, each deal this time consisting of 100 cars drawn from one or more classes. But whereas a buyer knows only the distributional properties of the classes, the seller knows which are lemons and which are good classes. The new problem is this information asymmetry allows a seller to assign a disproportionate number of cars from lemons classes to some deals, and to do so without detection by a computationally bounded buyer! The same story applies for CDOs, where a big bank can

choose how to package together assets into derivatives.

The authors establish the intractability of detecting rigged financial products for the kinds of CDOs that arise in the financial industry. The penalty for the realism of their model is that the hardness assumption that they require is not as standard as **P** vs **NP**; rather the results assume the intractability of finding *planted dense subgraphs* in random graphs. The seller is doing the "planting" in this case, by placing a disproportionate number of assets from one class into some subset of products. Under this assumption, CDOs cannot alleviate the lemons problem: either buyers are fooled and sellers make excess profits, or buyers know not to trust sellers. Many believe the planted dense subgraph problem is hard, and this has been considered a plausible conjecture before this paper was published. Still, it is possible this hardness assumption is false, and this should be studied by computer scientists.

This provocative paper should be required reading for commentators and financial regulators alike. Among the questions it raises: Are sellers using this information asymmetry to their advantage in packaging "booby-trapped" CDOs and other financial derivatives? Given that buyers and ratings agencies may not be aware of their own computational limitations, is there a role for regulation in protecting buyers by banning complex financial products that are provably untrustworthy? Do there exist derivatives that cannot be manipulated by strategic sellers, thus avoiding this new lemons cost due to computational complexity?

Buyers might like to reflect on the implications of their bounded rationality. *Caveat emptor!* 

David C. Parkes (parkes@eecs.harvard.edu) is Gordon McKay Professor of Computer Science in the School of Engineering and Applied Sciences at Harvard University, where he founded the EconCS research group.

© 2011 ACM 0001-0782/11/05 \$10.00

Computational Complexity and Information Asymmetry in Financial Products

By Sanjeev Arora, Boaz Barak, Markus Brunnermeier, and Rong Ge

1. INTRODUCTION

A *financial derivative* is a contract entered between two parties, in which they agree to exchange payments based on events or on the performance of one or more underlying assets—independently of whether they own or control the underlying assets (e.g., the DOW Jones index falling below 10,000). Securitization of cash flows using derivatives transformed the financial industry over the last three decades. However, mispricing of these derivatives is believed to have contributed to the financial crash of 2008 (see e.g. Brunnermeier⁸ and Coval et al.¹⁰).

There are also suggestions that derivatives were deliberately misused. In Spring 2010 there was a famous allegation about investment bank Goldman Sachs's Abacus derivative. The security and exchange commission alleged that Goldman Sachs collaborated with short-seller Paulson to select particularly bad mortgages as the underlying assets for this derivative. Tranches of Abacus were sold to ABN Amro and IKB Deutsche Industriebank, who were unaware of Goldman's selection methods and lost almost \$1 billion within a few months of buying these assets.

Hence, it is not surprising that derivatives have attracted criticism—Warren Buffett famously called them “financial weapons of mass destruction”—accompanied with calls for extensive regulation and even an outright ban. Others point out—with ample justification from economic theory—that derivatives are beneficial because they allow better risk sharing by “completing” markets, and also protect buyers from the effects of asymmetric information, ameliorating the so-called “lemon” problem (which arises whenever one party in the transaction has more information about the asset than the other; cf. Section 2). According to this viewpoint, problems with derivatives would disappear with use of more accurate financial models, more vigilance by buyers and better governmental oversight.

In our paper,⁵ which the current write-up is trying to describe at a simplified level, we injected a new aspect into this debate. We show that *even when the underlying financial model used by buyers and sellers is correct* there is an inherent obstacle to accurate pricing due to *computational complexity*. Formally, even in industry-standard models, the pricing problem can be as difficult as solving the *planted dense subgraph* problem, which has been proposed as a basis for cryptosystems. The practical implication is that though derivatives such as collateralized debt obligations (CDOs) can theoretically ameliorate the effects of asymmetric

information in the market, in practice these effects will persist—or even get worse—because market participants are not computationally sophisticated enough to solve cryptographic problems. This suggests that better regulation and more informed buyers are not sufficient for derivatives market to work correctly, or at least that regulators and buyers should take computational complexity into account. Such issues are further discussed at the end of the paper in Section 5.

The bite of computational complexity: Computational complexity studies *intractable* problems, such as NP-complete problems, which are conjectured to require more computational resources than can be provided by the fastest computers. (For an introduction see the text.⁴) The key reason it comes naturally into the study of financial derivatives is that it implies an asymmetry between the *ease* of creating problems and *solving* them. A simple example is the problem of *factoring* integers. It is easy to take two random prime numbers—say 7019 and 5683—and multiply them—in this case, to obtain 39888977. However, given 39888977, it is not that easy to factor it to get the two numbers 7019 and 5683. Algorithms that search over potential factors take very long time. This difficulty becomes more pronounced as the numbers have more and more digits. Computer scientists believe that factoring an n -digit number requires roughly $\exp(n^{1/3})$ time to solve,^a a quantity that becomes astronomical even for a moderate n like 10,000. The intractability of this problem leads to a concrete realization of *information asymmetry*. Anybody who knows how to multiply can randomly generate (using a few coin flips and a pen and paper) a large integer by multiplying two smaller factors. This integer could have say 1000 digits, and hence can fit in a paragraph of text. The person who generated this integer knows its (prime) factors, but no computational device in the universe can find a nontrivial factor in any plausible amount of time.^b This informational asymmetry underlies modern cryptosystems, which allow (for

^a The precise function is more complicated, but in particular the security of most electronic commerce depends on the infeasibility of factoring integers with roughly 800 digits.

^b Experts in computational complexity should note that we use factoring merely as a simple illustrative example. For this reason we ignore the issue of *quantum computers*, whose possible existence is relevant to the factoring problem, but does not seem to have any bearing on the computational problems used in this paper.

A previous version of this paper appeared in *The First Symposium on Innovations in Computer Science (ICS 2010)*. Tsinghua University Press, Beijing, China; 49–65.

example) two parties to exchange information over an open channel in a way that an eavesdropper can extract *no* information from it—not even distinguish it from a randomly generated sequence of symbols. More generally, in computational complexity we consider a computational task *infeasible* if the resources needed to solve it grow *exponentially* in the length of the input, and consider it *feasible* if these resources only grow polynomially in the input length.

Computational complexity immediately implies the existence of hard-to-price derivatives, albeit unnatural ones. Consider for example a derivative whose contract contains a 10,000 digit integer n and has a nonzero payoff iff the unemployment rate next January, when rounded to the nearest integer, is the last digit of a factor of n . A relatively unsophisticated seller can generate such a derivative together with a fairly accurate estimate of its yield (to the extent that unemployment rate is predictable), yet even a sophisticated investor like Goldman Sachs would have no idea what to pay for it. This example shows both the difficulty of pricing arbitrary derivatives and the possible increase in asymmetry of information via derivatives.

While this “factoring derivative” is obviously far removed from anything used in current markets, in this work we show that similar effects can be obtained in simpler and more popular classes of derivatives that are essentially the ones used in real life in securitization of mortgages and other forms of debt. The person selling the derivative can structure (“rig”) the derivative in a way such that it has low yield, but distinguishing it from a normal (“unrigged”) higher yield derivative is computationally intractable. Thus any efficient pricing mechanism would either overvalue the rigged derivative or undervalue the unrigged one, hence creating an inefficiency in the market.

Densest subgraph problem: Our result relies on the conjecture that there does not exist a tractable algorithm to detect large *dense subgraphs* in random graphs. This is a more specialized assumption than the familiar $P \neq NP$ conjecture. We needed this assumption because we needed to exhibit the intractability of “real-life” derivatives, and the setting there naturally leads to random graphs, as will be clear in the description in Section 4.

Computational complexity and “bounded rationality”: Computational complexity can be related to the *bounded rationality* concept in economics. Simon¹⁴ proposed the notion of bounded rationality to recognize that in decision making, real-life agents are limited by their cognitive ability to process information and the finite amount of time they have. Simon postulates that agents use heuristics instead of time-consuming and complex optimizing behavior. Experimental evidence on behavioral biases supports this notion (e.g. Kahneman,¹³ etc.). On the other hand, economic experiments also suggest that as the stakes rise and people face similar situations repeatedly, they behave more deliberately in a way that approaches rationality. In particular this is the case in the setting of finance, where stakes are high and traders have access to cutting edge technology. However, even the most sophisticated traders cannot escape the limitations of computational complexity, since no physically realizable computational device can solve intractable problems.

And we show in this note that pricing certain financial derivatives may require solving problems that are believed to be intractable, hence placing it beyond the reach of any real-life agent.

2. THE “LEMONS PROBLEM” IN ECONOMICS

To understand the theoretical benefits of financial derivatives it is useful to recall the *lemons problem*, introduced in Akerlof’s classic 1970 paper.¹ The simplest setting is as follows. You are in the market for a used car. A used car in working condition is worth \$1000. However, 20% of the used cars are *lemons* (i.e., are useless, even though they look fine on the outside) and their true worth is \$0. Thus if you could pick a used car at random then its expected worth would be only \$800 and not \$1000. Now consider the seller’s perspective. Suppose sellers know whether or not they own a lemon. A seller who knows he has a non-lemon would be unwilling to sell for \$800, and would therefore withdraw from the market. The market would be left only with lemons, and knowing this, buyers would refuse to buy any car. Thus the market grounds to a halt. Akerlof’s paper goes on to analyze reasons why used cars do sell in real life. We will be interested in one of the reasons, namely, that there could be a *difference* between what a car is worth to a buyer versus a seller. In the above example, the seller’s value for a working car might be \$200 less than the buyer’s—perhaps because the seller is moving across the country and needs the cash—thus allowing trade to occur. In this case we say that the “lemon cost” of this market is \$200. Some authors refer to the lemon cost as a *wedge*. Generally, the higher this cost, the less efficient the market.

The lemons problem can potentially arise in almost every area of the economy, and a large body of work in *information economics* describes how it can be ameliorated. Akerlof’s original paper already described signaling mechanisms by which a seller can reliably communicate his private information—namely, prove that his car is not a lemon—to the buyer. For example, a used car dealer can show the buyer repair records, or provide a warranty for 6 months, or point out to his stellar reputation and rating from the Better Business Bureau.

3. FINANCIAL DERIVATIVES AND CDOs

The lemons problem also arises in the financial industry and the usual mechanisms for dealing with lemons problems (as identified by Akerlof) have also flowered: borrower FICA ratings (a.k.a. credit scores), ratings of individual securities by agencies such as Moody’s, etc. Financial derivatives provide another mechanism for dealing with the lemons problem. Below we will illustrate this with a common derivative called the *collateralized debt obligation* or *CDO*.

It is not commonly known, but the humble home mortgage is actually a very complex instrument that presents great risks for lenders. The payoff structure is complicated; risk of default is fairly high (compared say to a U.S. treasury bond); and there is high risk of prepayment at times of low interest rates, which is precisely when the mortgage’s locked-in higher rate is most valuable to the lender. CDOs are financial devices that allow many mortgages to be aggregated into a security that has a supposedly much more

predictable yield, and hence more attractive to risk-averse lenders such as retirement funds.

Consider the following simplistic example: suppose a bank holds a portfolio of 100 mortgages, and that each mortgage yields \$0 if the borrower defaults and \$1 million otherwise. For now assume the probability of default is 10%, which implies that the expected yield (and hence fair price) for the entire portfolio is \$90 million. The bank would like to get all or most of these mortgages off its books because this is favorable for regulatory reasons. Since each individual mortgage may be unacceptably risky for a risk-averse buyer, the bank holding the mortgages can do the following. Create two new assets by combining the above 100 mortgages. Set a *threshold*, say \$80 million. The first asset, called the *senior tranche*, has claim to the first 80 million of the yield; and the second, called the *junior tranche*, has claim to the rest. These assets are known as collateralized debt obligations (CDOs). The bank offers the senior tranche to the risk-averse buyer and holds on to the junior tranche (Figure 1).

Now a risk-averse buyer may reason as follows: *if* the mortgage defaults are independent events (which is a big if, though in practice justified by pooling mortgages made to a geographically diverse group of homeowners, whose defaults are presumably independent) then the senior tranche is extremely unlikely to ever yield less than its maximum total value of \$80 million. In fact for this to happen, more than 20 mortgages need to default, which happens only with probability $\sum_{k=20}^{100} \binom{100}{k} (0.1)^k (1-0.1)^{100-k} \approx 0.08\%$. Thus the senior tranche is a very safe asset with a highly predictable yield, and such derivatives were often rated by credit-rating agencies to be as safe as a U.S. treasury bond. In real-life CDOs the mortgage yields, payment streams, and tranching are all much more complex, but the basic idea is the same.

The lemons problem enters: There is an obvious lemons problem in the above scenario because of asymmetric information: The bank that issued the mortgages has the most accurate

estimate of the rate of default, whereas the buyer may have a less precise idea—say that the rate lies between 10% and 15%. Economic theory says that in addition to transforming the risk profile of the asset, the CDO also protects the investor from this lemons problem. The crucial observation is that even if the probability of default was 15%, the probability of the senior tranche yielding its maximum of \$80 million would still be roughly 99.9%, and hence the tranching *insulates* the buyer from the information-sensitive part of the mortgage pool.

In fact, as was shown by DeMarzo,¹¹ the choice of the threshold can be used as a *signaling* mechanism that allows the bank to transmit in a trustworthy way the true default value. We now illustrate the idea behind this result. Consider the problem from the bank's viewpoint. It is interested in getting rid of as many mortgages—specifically, the largest possible portion of the entire portfolio—from its book as possible, so it wants to set the threshold as high as possible. Suppose it knows the default rate is 15%. This is the highest possible default rate, so it can simply sell the whole bundle of mortgages (i.e. set the threshold to 100%). The buyers will use their most pessimistic evaluation (that the default rate is 15%) and pay the price of \$85 million. Both the seller and the buyer are satisfied because the price is just equal to the estimated yield. Suppose on the other hand that the bank knows the default rate is actually only 10%, the lowest possible rate. Now setting the threshold to 100% is no longer its best strategy, since the buyers will just pay \$85 million while the bundle is now worth \$90 million. To signal its confidence in the quality of mortgages, the bank will tranch the pool, set the threshold to 80%,^c and offer to hold the riskier part—the junior tranche. Knowing that the bank will not offer this lower threshold when the default rate is high (indeed, the best threshold for default rate 15% is 100%), the rational buyers should correctly interpret this as a signal to the true default rate and pay close to \$80 million for the senior tranche. Again, both the seller and the buyer are satisfied because the price is almost equal to the estimated yield of the senior tranche.

Making the above intuitive argument precise in the usual rational expectations framework of economic theory takes some work, and was done in DeMarzo,¹¹ where it is shown that the CDO is the *optimum* solution to the lemons problem in a setting somewhat more general than the above simplistic one. Specifically, the CDO allows the lemon cost—i.e., the difference in valuation of the security by buyer and bank required for the sale to occur—to approach 0.^d That is, the bank's secret information does not lead to large market inefficiencies. Henceforth we will refer to this as *DeMarzo's Theorem*.

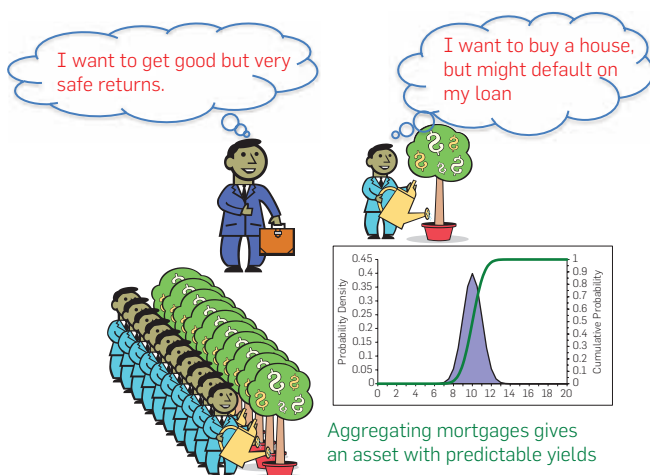
4. WHY COMPLEXITY MATTERS

We presented above the traditional justification for CDOs from economic theory. Now we explain at an intuitive level

^c The exact threshold here depends on a number of factors, including default rate and *discount factor*. The discount factor shows how much the seller prefers cash to assets. The threshold can be computed exactly using methods in DeMarzo.¹¹

^d A nonzero difference in valuation or *wedge* between the bank and buyer arises because the buyer holds cash and the bank holds the mortgages, and the bank prefers cash to mortgages because of regulatory or other reasons.

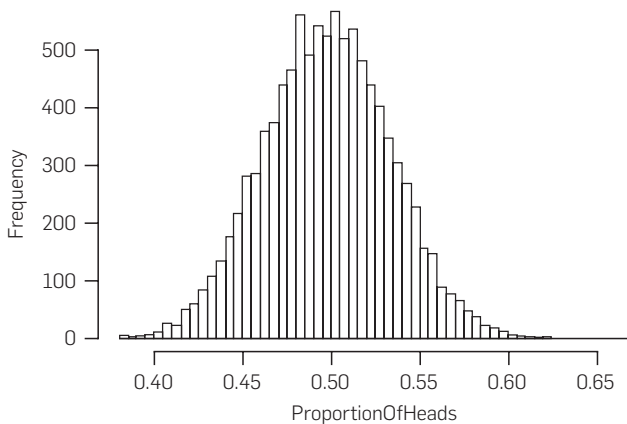
Figure 1. Aggregating many mortgages into a single asset makes the yield more predictable due to the law of large numbers (central limit theorem). This assumes that the yields of the different mortgages are independent random variables.



GAUSSIANS AND THE CENTRAL LIMIT THEOREM

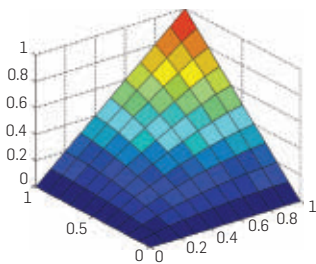
The ability of CDOs to ameliorate effects of asymmetric information relies on the *law of large numbers*, which informally speaking states that the total payoff of a bundle of mortgages is close to the mean (or expected) value. More precisely, we have the central limit theorem: The sum of sufficiently many bounded and independent random variables will be approximately distributed like a Gaussian with same mean and variance. The Gaussian is a good approximation even when the number of variables is only in the hundreds. Therefore, if there are 100 mortgages, each paying 1 with probability 1/2 and paying 0 otherwise, the distribution of the total payoff is like a Gaussian with mean 50 and standard deviation 5. The probability that the payoff is outside [35, 65] (three standard deviations) is less than 0.2%.

Histogram of ProportionOfHeads

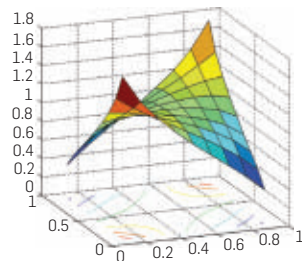


In addition to their value as limiting distributions for the sum of independent random variables, Gaussians arise in one other way in finance: often the payoffs of assets themselves are assumed to be Gaussians. The joint distribution of these Gaussian valued assets is the well-known *Gaussian copula*:

Gaussian copula



Density of Gaussian copula



Although in the illustrative example we assumed binary payoffs for a single asset, similar results hold for asset yields that form a Gaussian copula with the same mean, variance, and covariance.

why introducing computational complexity into the picture greatly complicates the picture, and even takes away some of the theoretical benefits of CDOs. The full analysis appears in our longer paper.

The important twist we introduce in the above scenario is that a large bank is selling many CDOs and not just a single one. DeMarzo's theorem does not generalize to this case, and indeed we will show that whether or not the CDOs ameliorate the lemons problem depends upon the computational ability of the buyer. The lemons problem gets ameliorated only if the buyer is capable of solving the densest subgraph problem, which is currently believed to be computationally difficult.

We will use the assumption—consistent with practice—that mortgages are grouped into *classes* depending upon factors such as the borrower's credit score, geographic location, etc., and that default rates within a class are the same. Consider for simplicity a bank with N asset classes, each of which contains C assets. Some asset classes are “lemons”: assets in these classes will always *default* and have payoff 0. All other asset classes are good: assets in these classes pay $1/C$ with probability 1/2 and default (i.e., have payoff 0) with probability 1/2. The yields of assets from different asset classes are independent.

The buyer's prior is that the number of lemon classes is uniformly distributed in $[0, 2n]$ for some $n < N/2$, and the set of lemon classes is uniformly picked among all classes. However, the bank has additional information: it knows precisely which classes are lemons (this implies that it knows the number of lemons as well). This is the *asymmetric information*.

Since the expected number of lemon classes is n , each with payoff 0 and the remaining $N - n$ good classes have payoff $1/2$, a buyer purchasing the entire portfolio would be willing to pay the expected yield, which is $(N - n)/2$. Thus a *wedge* à la Akerlof arises for banks who discover that the number of lemons is lower than the expectation, and they would either exit the market, or would need to prefer cash by an amount that overcomes the wedge.

Of course, DeMarzo's theorem allows this lemons problem to be ameliorated, via securitization of the entire portfolio into a single CDO. As already mentioned, we are interested in the case where the number of assets held by the bank is *large*, and so, rather than using a single CDO, the bank partitions them into multiple CDOs. Now how does the bank's extra information affect the sale? Clearly, it has new *cherry-picking possibilities*, involving which asset to package into which CDO. We will assume that all transactions are public and visible to all buyers, which means that seller must do any such cherry picking in full public view.^c

Now let us show that in principle derivatives should still allow buyers to rule out any significant cherry picking, thus ameliorating the lemon wedge. Consider the following: the seller creates M new financial products, each of them

^c This assumption of transparency only makes our negative results stronger. It also may be a reasonable approximation if buyers are well-informed, and recent financial regulation has mandated more transparency into the market.

a CDO depending on a pool of D of the underlying assets. We assume $MD = NC$ and that every asset occurs in exactly one pool. Clearly, assets in the same class should be distributed to different products in order to ensure the diversity of the products. Each CDO will have assets from distinct classes (and hence have stochastically independent yields) so that sum of their yields follows the central limit theorem. Suppose each one of the M products has the following design: it pays off $N/(3M)$ units as long as the number of assets in its pool that defaulted is at most $D/2 + t\sqrt{D}$ for some parameter t (set to be about $\sqrt{\log D}$), and otherwise it pays 0. Henceforth we call such a product a “binary CDO”; it can be viewed as the senior tranche of a simple CDO.^f Assets contained in the same product come from different classes, the yields of good assets are uniformly iid, so the expected number of defaults among D good assets is $D/2$ and the standard deviation is $\sqrt{D}/2$. Now the central limit theorem applies and the total number of defaults may be assumed to be distributed like a Gaussian. Thus so long as the fraction of lemon classes is much smaller than the safety margin of t standard deviations, the probability of default for an individual CDO is tiny. Thus, if V denotes the combined expected yield of these M products, then $V \approx M \times N/3M = N/3$. (The exact value of V is unimportant below.)

If the bank were to pick the pools truly randomly—i.e., the entire portfolio of assets is randomly partitioned into the M pools—then the portfolio’s expected yield is only mildly affected by the presence of lemons. Specifically, if V is the expected yield when there are no lemon classes, then it can be shown that the yield is still $V - o(n)$ (i.e. larger than $V - \epsilon n$ for any $\epsilon > 0$) when the number of lemon classes is $2n$, the maximum possible. In this sense derivatives can help significantly reduce the lemon wedge from n to $o(n)$, thus performing their task of allowing a party to sell off the least information-sensitive portion of the risk.

However, the above description assumed that the seller creates the pools *disinterestedly* using pure randomness. But this may be against his self-interest given his secret information! Given that the seller’s interest is to give out the minimum yield possible, as long as this is undetected by the buyer, it turns out that his optimum strategy is to pick some subset of m of the financial products, and ensure that the lemon assets are *overrepresented* in the pools of these m products—to an extent about $t\sqrt{D}$, which is just enough to significantly skew the probability of default. We call this subset of n CDOs the “boobytrap.” Thus the CDOs in the boobytrap have a much higher probability of default than buyers expect, causing the expected yield of the entire portfolio of CDOs to be smaller by an amount proportional to m (roughly $mN/(3M)$).^g

With some settings of m the tampered derivative can have much smaller yield than the yield of $V - o(n)$ obtained

by random pooling. The question is whether buyers can be fooled by this cherry picking. We have to consider two cases, based on the buyer’s computational powers.

Fully rational (computationally unbounded) buyer: He will not be fooled. Even though he does not know the set of lemon classes, he knows thanks to *random graph theory* (see the excellent references of Alon and Spencer² and Bollobás⁷) that in a randomly chosen portfolio of CDOs the possibility of accidentally setting up such a boobytrap is vanishingly remote. Therefore it suffices for him to rule out the existence of any boobytrap in the presented portfolio: he enumerates over *all* possible $2n$ -sized subsets of the N classes and verifies that *none* of them are over-represented in *any* subset of m products. The same calculations as above guarantee him that in this case the yield of the derivative is at least $V - o(n)$, even though he does not know the identity of the lemon classes. Thus a seller has no incentive to plant a boobytrap for a fully rational buyer, and the lemon wedge is indeed ameliorated greatly if buyers are fully rational.

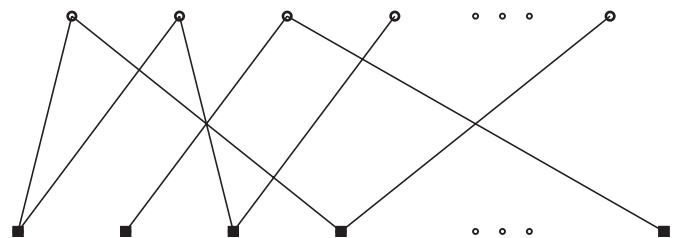
Real-life buyer, who is feasibly rational (computationally bounded): For him the above computation for detecting boobytraps is infeasible even for moderate parameter values.

To get an appreciation of the infeasible problem lurking here, it helps to take a graph-theoretic view of the problem. Recall that a *bipartite* graph consists of two disjoint sets of vertices A, B such that each edge has an endpoint in both A and B . We can use a bipartite graph to represent the portfolio of CDOs: A is the set of asset classes and B is the set of CDOs, and an edge (a, b) indicates that the CDO numbered b contains an asset from the asset class numbered a (see Figure 2).

Of course the buyer will also try other possible algorithms to detect the boobytrap. If the bank randomly throws assets into CDOs, then this graph that represents the portfolio is some kind of random graph. If the bank creates a boobytrap as described above, then the boobytrap corresponds to a *dense subgraph* in this bipartite graph: it is a subset of asset classes (the lemons) and a subset of CDOs (the boobytrapped ones) where the number of edges lying between them is substantially higher than it would be in a random graph.

The problem of detecting a boobytrap is equivalent to the so-called *hidden dense subgraph* problem, which is widely believed to be intractable. In fact the conjecture is that there is no efficient way to distinguish the truly random bipartite

Figure 2. Using a bipartite graph to represent asset classes and derivatives. There are M vertices on top corresponding to the derivatives and N vertices at the bottom corresponding to asset classes. Each derivative references D assets in different classes.



^f This is a so-called *synthetic binary option*. The more popular CDO derivative described above behaves in a similar way, except that if there are defaults above the threshold (in this case $D/2 + t\sqrt{D}$) then the payoff is not 0 but the defaults are just deducted from the total payoff. We call this a “tranching CDO” to distinguish it from the binary CDO.

^g The non-booby trapped CDOs will have a slightly smaller probability of default than in the untampered (i.e., random) case, but a simple calculation shows that this will only contribute a negligible amount to the yield.

graph from one in which the bank has “planted” the dense subgraph (a.k.a. boobytrap). Formally, the two kinds of graphs are believed to be *computationally indistinguishable* for polynomial-time algorithms and this was the basis of a recent cryptosystem proposed by Applebaum et al.³

The conjecture is that even quite large boobytraps may be undetectable: the expected yield of the entire portfolio could be much less than say $V - n^{1.1}$ and yet the buyer may not be able to distinguish it from a truly random (i.e., honestly constructed) portfolio, whose yield is $V - o(n)$.

We conclude that if buyers are computationally bounded, then introducing derivatives into the picture not only *fails to reduce* the lemon wedge, but paradoxically, *amplifies* it even beyond the total value $2n$ of all lemon assets. Though the above example is highly simplified, it can be embedded in settings that are closer to real life and similar results are obtained.

4.1. Can the cost of complexity be mitigated?

In Akerlof’s classic analysis, the no-trade outcome dictated by lemon costs can be mitigated by appropriate signaling mechanism—e.g., car dealers offering warranties to increase confidence that the car being sold is not a lemon. In the above setting, however, there seems to be no direct way for seller to prove that the financial product is *untampered* i.e., free of boobytraps. (It is believed that there is no simple way to prove the absence of a dense subgraph; this is related to the $NP \neq coNP$ conjecture.) Furthermore, we can show that for suitable parameter choices the tampering is *undetectable* by the buyer even *ex post*. The buyer realizes at the end that the financial products had a much lower yield than expected, but would be unable to prove that this was due to the seller’s tampering. Nevertheless, we do show in our paper⁵ that one could use ideas from computer science in designing derivatives that are tamperproof in our simple setting.

4.2. Complexity ranking

Recently, Brunnermeier and Oehmke⁹ suggested that traders have an intuitive notion of complexity for derivatives. Real-life markets tend to view derivatives such as CDO^2 (a CDO whose underlying assets are CDOs like the one described earlier) as complex and derivatives like CDO^3 (a CDO whose underlying assets are CDO^2) as even more so. One might think that the number of layers of removal from a simple underlying real asset could be a natural measure of complexity. However, as Brunnermeier and Oehmke⁹ point out, such a definition might not be appropriate, since it would rank e.g. highly liquid stocks of investment banks, which hold CDO^2 s and other complex assets, as one of the most complex securities. Our paper⁵ proposes an alternative complexity ranking which is based on the above discussed notion of *lemon cost due to complexity*. This ranking also confirms the standard intuition that CDO^2 s are more complex than CDO s. Roughly speaking, the cherry-picking possibilities for sellers of CDOs described in this paper become even more serious for derivatives such as CDO^2 and CDO^3 .

5. DISCUSSION

The notion that derivatives need careful handling has been extensively discussed before. Coval et al.¹⁰ show that pricing (or rating) a structured finance product like a CDO is extremely fragile to modest imprecision in evaluating underlying risks, including systematic risks. The high level idea is that these everyday derivatives are based upon the *threshold* function, which is *highly sensitive* to small perturbations of the input distribution. Indeed, empirical studies suggest that valuations for a given financial product by different sophisticated investment banks can be easily 17% apart⁶ and that even a single bank’s evaluations of different “tranches” of the same derivative may be mutually inconsistent.¹² Thus one imagines that banks are using different models and assumptions in evaluating derivatives.

The question studied in our work is: *Is there a problem with derivatives even if one assumes away the above possibilities, in other words the yield of the underlying asset exactly fits the stochastic model assumed by the buyer?* Economic theory suggests the answer is “No”: informed and rational buyers need not fear derivatives. (Recall our discussion of DeMarzo’s theorem.)

The main contribution of our work has been to formalize settings in which this prediction of economic theory may fall short (or even be falsified), and manipulation is possible and undetectable by all real-life (i.e., computationally bounded) buyers. We have worked within existing conceptual frameworks for asymmetric information. It turns out that the seller can benefit from his secret information (viz., which assets are lemons) by using the well-known fact that a random election involving n voters can be swung with significant probability by making $\sqrt{n}$ voters vote the same way; this was the basis of the boobytrap described earlier. The surprising fact is that a computationally limited buyer may not have any way to distinguish such a *tampered* CDO from untampered CDOs. Formally, the indistinguishability relies upon the conjectured intractability of the *planted dense subgraph* problem.^h

The model in our more detailed paper has several notable features:

1. The largeness of the market—specifically, the fact that sellers are constructing thousands of financial products rather than a single product as was the case in the model of DeMarzo¹¹—allows sellers to cherry pick in such a way that cannot be detected by feasible rational (computationally bounded) buyers—i.e., all real-world buyers—while it can be detected by fully rational (computationally unbounded) buyers.
2. The possibility of cherry picking by sellers creates an Akerlof-like *wedge* between buyer’s and seller’s valuations of the financial product. We call this the *lemon cost due to computational complexity*. In our detailed paper we can quantify this wedge for several classes of derivatives popular in securitization. This allows a par-

^h Note that debt-rating agencies such as Moody’s or S&P currently use simple simulation-based approaches to evaluate derivatives, which certainly do not attempt to solve something as complicated as densest subgraph.

tial ranking of these classes, which can be seen as a quantification of more familiar heuristic notions of “complexity.” This answers the open question of Brunnermeier and Oehmke.⁹

3. It can be difficult for regulatory bodies to control the above-mentioned cherry picking because the cherry picking can be difficult to detect *ex ante*. In some models the cherry picking seems undetectable even *ex post*. Both these remain true even in a fully transparent market where all transactions occur on a public exchange. It also implies that verifying the existence of the lemon cost due to computational complexity in historical data (in other words, an empirical test of our paper) may prove difficult, especially given that the market has not been fully transparent.

In sum, our approach of combining insights from computer science with economic questions allows one to formally study phenomena, such as complexity and bounded rationality, that are of first-order importance but were difficult to capture in formal economic models. These new insights should help shape future regulation and the post-2008 financial architecture. 

References

1. Akerlof, G.A. The market for “lemons”: Quality uncertainty and the market mechanism. *Q. J. Econ.* 84(3) (1970), 488–500.
2. Alon, N., Spencer, J.H. *The Probabilistic Method*, 3rd edn. Wiley, Hoboken, NJ, 2008.
3. Applebaum, B., Barak, B., Wigderson, A. Public-key cryptography from different assumptions. In *Proceedings of STOC*, 2010, 171–180.

4. Arora, S. and Barak, B. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.
5. Arora, S., Barak, B., Brunnermeier, M., Ge, R. Computational complexity and information asymmetry in financial products. In *The First Symposium on Innovations in Computer Science, ICS 2010*, Tsinghua University Press, Beijing, 2010, 49–65.
6. Bernardo, A.E., Cornell, B. The valuation of complex derivatives by major investment firms: Empirical evidence. *J. Finance* 52 (1997), 785–798.
7. Bollobás, B. *Random Graphs*, 2nd edn. Cambridge University Press, 2001.
8. Brunnermeier, M.K. Deciphering the liquidity and credit crunch 2007–08. *J. Econ. Perspect.* 23(1) (2009), 77–100.
9. Brunnermeier, M.K., Oehmke, M. Complexity in financial markets. *Working Paper*. Princeton University.
10. Coval, J., Jurek, J., Stafford, E. The economics of structured finance. *J. Econ. Perspect.* 23(1) (2009), 3–25.
11. DeMarzo, P.M. The pooling and tranching of securities: A model of informed intermediation. *Rev. Financ. Stud.* 18(1) (2005), 1–35.
12. Duffie, D. Innovations in credit risk transfer: Implications for financial stability. *BIS Working Papers*.
13. Kahneman, D. Maps of bounded rationality: Psychology for behavioral economics. *Am. Econ. Rev.* 93(5) (2003), 1449–1475.
14. Simon, H.A. Bounded rationality and organizational learning. *Organ. Sci.* 2(1) (1991), 125–134.

Sanjeev Arora (arora@cs.princeton.edu), Department of Computer Science, Center for Computational Intractability, Princeton University, Princeton, NJ.

Markus Brunnermeier (markus@princeton.edu), Department of Economics, Bendheim Center for Finance, Princeton University, Princeton, NJ.

Boaz Barak (boaz@microsoft.com), Microsoft Research New England, Princeton University, Princeton, NJ.

Rong Ge (rongge@cs.princeton.edu), Department of Computer Science, Center for Computational Intractability, Princeton University, Princeton, NJ.

© 2011 ACM 0001-0782/11/05 \$10.00



Association for
Computing Machinery

Advancing Computing as a Science & Profession



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



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

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



Make a difference to a student in your field.
Sign up today at: www.mentornet.net
Find out more at: www.acm.org/mentornet

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

Technical Perspective

Images Everywhere Looking for Models

By Guillermo Sapiro

ABOUT 5,000 IMAGES per minute are uploaded to the photo-sharing site <http://www.flickr.com/>; over 7,000,000 a day. Similar numbers are uploaded to other social sites. Often these images are acquired by amateur photographers under non-ideal conditions and with low-end digital cameras such as those available in mobile phones. Such images often look noisy, blurry, and with the wrong colors or contrast. Even images acquired by high-end devices, such as MRI or microscopy, suffer from these effects due to the intrinsic physics of the device and the structure of the material being photographed. A key challenge in image science then is how to go from the “low-” quality image to a high-quality one that is sharp, has good contrast, and is clean of artifacts. This is an intrinsically ill-posed inverse problem, according to Hadamard’s definition. So, what do we do?

We have to include additional assumptions, a process often called *regularization*. These assumptions come with different names depending on one’s particular area of research or interest, and are often called *priors* or *models*. Deriving appropriate regularization terms, priors or models, has occupied the research community since the early days of digital image processing, and we have witnessed fantastic and very inspiring models such as linear and nonlinear diffusion, wavelets, and total variation. Different image models can be appropriate for different types of images; for example, MRI and natural images should have different models. Indeed, some models might be useful for some inverse problems and not for others.

In their landmark paper, Buades, Coll, and Morel discuss a number of image models under a unified framework. Let us concentrate on the *self-similarity* model, which leads to the important *non-local means* algorithm proposed by the authors for image denoising and its extensions to other

image inverse problems. The basic underlying concept is that local image information repeats itself across the non-local image. Noise, on the other hand, is expected in numerous scenarios to be random. Therefore, collecting those similar local regions all across the image, the noise can be eliminated by simple estimators based on having multiple observations of the same underlying signal under different noise conditions. This simple and powerful idea of self-similarity, which brings a unique perspective of simultaneous local and non-local processing, dates at least to Shannon’s model for English writings in 1950 (“Prediction and Entropy of Printed English,” *Bell Sys. Tech. J.*, 50–64), and was used in image processing for synthesis tasks. But it was not until the 2005 elegant paper by Buades et al. that the community had its Eureka moment and clearly realized it could be exploited for reconstructions challenges as well.

This idea of self-similarity opened a large number of questions. At the practical level, we could ask how to define the scale of the local regions, how to efficiently find similar regions in an image, how to define the distance between local image regions to determine that they are “similar,” and what type of image processing tasks can be addressed with this model. At the theoretical level, standard questions like consistency of the estimator and

In their landmark paper, Buades, Coll, and Morel discuss a number of image models under a unified framework.

its optimality are naturally raised. The image processing community is busy addressing these questions.

There is another critical aspect clearly illustrated by the following seminal work, this being the idea of addressing image inverse problems with overlapping local image regions, or *overlapping image patches*. In many scenarios this became the working unit, replacing the standard single point or pixel (sometimes these are now called *super-pixels*). While some researchers have adopted models that are different than the self-similarity one, it is safe to say that today, six years after their original paper was published, the state-of-the-art techniques for image reconstruction, as well as for image classification, are all based on working with these super-pixels or patches. This has become a fundamental building block of virtually all image models.

The authors’ work also starts hinting at the idea that we can learn the model from the data itself, or at least adapt it to the image, instead of relying on predefined mathematical structures. This relates to *dictionary learning*, where the image is modeled as being represented via a learned dictionary. The self-similarity model assumes the dictionary is the image itself, or actually its local patches. All these models indicate that images, and in particular image patches, do not actually live in the ambient high-dimensional space, but in some much lower dimensional stratification embedded on it.

For over 40 years, the image processing community has been on the lookout for image models. The most fundamental of them have left important footprints in the community. Many of the questions are still open today, from the eternal battle between generative and discriminative models to the need of deriving computationally feasible and fundamentally useful models. All this work goes to the root of our desire to know “*What is an image?*”



Guillermo Sapiro (guille@umn.edu) is a professor in the Department of Electrical and Computer Engineering at the University of Minnesota.

© 2011 ACM 0001-0782/11/05 \$10.00

Self-Similarity-based Image Denoising

By Antoni Buades, Bartomeu Coll, and Jean-Michel Morel

Abstract

The search for efficient image denoising methods is still a valid challenge at the crossing of functional analysis and statistics. In spite of the sophistication of the recently proposed methods, most algorithms have not yet attained a desirable level of applicability. All show an outstanding performance when the image model corresponds to the algorithm assumptions but fail in general and create artifacts or remove image fine structures. The main focus of this paper is, first, to define a general mathematical and experimental methodology to compare and classify classical image denoising algorithms and, second, to describe the nonlocal means (NL-means) algorithm⁶ introduced in 2005 and its more recent extensions. The mathematical analysis is based on the analysis of the “method noise,” defined as the difference between a digital image and its denoised version. NL-means, which uses image self-similarities, is proven to be asymptotically optimal under a generic statistical image model. The denoising performance of all considered methods are compared in four ways: mathematical, asymptotic order of magnitude of the method noise under regularity assumptions; perceptual-mathematical, the algorithms artifacts and their explanation as a violation of the image model; perceptual-mathematical, analysis of algorithms when applied to noise samples; quantitative experimental, by tables of L^2 distances of the denoised version to the original image.

1. INTRODUCTION

The goal of image denoising methods is to recover the original image from a noisy measurement:

$$v(i) = u(i) + n(i), \quad (1)$$

where $v(i)$ is the observed value, $u(i)$ is the “true” value, and $n(i)$ is the noise perturbation at a pixel i . The best simple way to model the effect of noise on a digital image is to add a Gaussian white noise. In that case, $n(i)$ are i.i.d. Gaussian values with zero mean and variance σ^2 .

Several methods have been proposed to remove the noise and recover the true image u . Even though they may be very different in tools it must be emphasized that a wide class share the same basic remark: denoising is achieved by averaging. This averaging may be performed locally: the Gaussian smoothing model,²⁷ the anisotropic filtering,^{2, 35} and the neighborhood filtering;^{40, 42, 46} by the calculus of variations: the total variation minimization;³⁹ or in the frequency domain: the empirical Wiener filters⁴⁶ and wavelet thresholding methods.^{12, 17}

Formally we define a denoising method D_h as a decomposition

$$v = D_h v + n(D_h, v),$$

where v is the noisy image and h is a filtering parameter, which usually depends on the standard deviation of the noise. Ideally, $D_h v$ is smoother than v and $n(D_h, v)$ looks like the realization of a white noise.

The denoising methods should not alter the original image u . Now, most denoising methods degrade or remove the fine details and texture of u . In order to better understand this removal, we shall introduce and analyze the *method noise*. The method noise is defined as the difference between the original (always slightly noisy) image u and its denoised version.

The denoising methods should not introduce visual artifacts. The *noise-to-noise* principle requires that a denoising algorithm transforms a white noise into white noise. This paradoxical requirement seems to be the best way to characterize artifact-free algorithms.

We also propose and analyze the *NL-means* algorithm, which is defined by the simple formula

$$NL[u](x) = \frac{1}{C(x)} \int_{\Omega} e^{-\frac{(G_a * |u(x+) - u(y+)|^2)(0)}{h^2}} u(y) dy,$$

where $x \in \Omega$, $C(x) = \int_{\Omega} e^{-\frac{(G_a * |u(x+) - u(z+)|^2)(0)}{h^2}} dz$ is a normalizing constant, G_a is a Gaussian kernel, and h acts as a filtering parameter. This formula amounts to say that the denoised value at x is a mean of the values of all points whose Gaussian neighborhood looks like the neighborhood of x . The main difference of the NL-means algorithm with respect to local filters or frequency domain filters is the systematic use of all possible self-predictions the image can provide, in the spirit of Efros and Leung.¹⁹ Section 2 gives formulas for the method noise of several classic algorithms. Section 3 describes NL-means. Section 4 gives its consistency results. A substantial experimental Section 5 compares several classic algorithms by the new introduced criteria and Section 6 reviews many recent extensions, applications, and variants.

2. METHOD NOISE

DEFINITION 1 (METHOD NOISE). Let u be an image and D_h a denoising operator depending on a filtering parameter h . Then, we define the method noise as the image difference $u - D_h u$.

The original version of this paper is entitled “A Review of Image Denoising Algorithms, with a New One.” It was published in *Multiscale Modeling and Simulation*, 2005.

The application of a denoising algorithm should not alter the non-noisy images. So the method noise should be very small when some kind of regularity for the image is assumed. If a denoising method performs well, the method noise must look like a noise even with non-noisy images and should contain as little structure as possible. Since even good quality images have some noise, it makes sense to evaluate any denoising method in that way, without the traditional “add noise and then remove it” trick. We shall list formulas permitting to compute and analyze the method noise for several classical local smoothing filters: the Gaussian filtering,²⁷ the anisotropic filtering,^{2, 35} the total variation minimization,³⁹ and the neighborhood filtering.⁴⁶ The formal analysis of the method noise for the frequency domain filters falls out of the scope of this paper. These method noises can also be computed but their interpretation depends on the particular choice of the wavelet or Fourier basis.

The Gaussian Filtering: The image isotropic linear filtering boils down to the convolution of the image by a linear symmetric kernel. The paradigm of such kernels is of course the Gaussian kernel $\mathbf{x} \rightarrow G_h(\mathbf{x}) = \frac{1}{(4\pi h^2)} e^{-\frac{|\mathbf{x}|^2}{4h^2}}$. In that case, G_h has standard deviation h and it is easily seen that

THEOREM 1 (GABOR 1960). *The image method noise of the Gaussian convolution satisfies $u - G_h * u = -h^2 \Delta u + o(h^2)$.*

The Gaussian method noise is zero in harmonic parts of the image and very large near edges or texture, where the Laplacian cannot be small. As a consequence, the Gaussian convolution is optimal in flat parts of the image but edges and texture are blurred.

The Anisotropic Filtering: The anisotropic filter (AF) attempts to avoid the blurring effect of the Gaussian by convolving the image u at $\mathbf{x}$ only in the direction orthogonal to $Du(\mathbf{x})$. The idea of this filter goes back to Perona and Malik³⁵ and is interpreted in Alvarez et al.² It is defined by

$$AF_h u(\mathbf{x}) = \int G_h(t) u\left(\mathbf{x} + t \frac{Du(\mathbf{x})^\perp}{|Du(\mathbf{x})|}\right) dt,$$

for $\mathbf{x}$ such that $Du(\mathbf{x}) \neq 0$ and where $(x, y)^\perp = (-y, x)$ and G_h is the one-dimensional Gauss function with variance h^2 . If one assumes that the original image u is twice continuously differentiable (C^2) at $\mathbf{x}$, it is easily shown by a second-order Taylor expansion that

THEOREM 2. *The image method noise of AF_h satisfies (for $Du(\mathbf{x}) \neq 0$)*

$$u(\mathbf{x}) - AF_h u(\mathbf{x}) = -\frac{1}{2} h^2 |Du| \text{curv}(u)(\mathbf{x}) + o(h^2).$$

By $\text{curv}(u)(\mathbf{x})$, we denote the curvature, that is, the signed inverse of the radius of curvature of the level line passing by $\mathbf{x}$. This method noise is zero wherever u behaves locally like a straight line and large in curved edges or texture (where the curvature and gradient operators take high values). As a consequence, the straight edges are well restored

while flat and textured regions are degraded.

Total Variation Minimization: The total variation minimization was introduced by Rudin et al.³⁹ Given a noisy image $v(\mathbf{x})$, these authors proposed to recover the original image $u(\mathbf{x})$ as the solution of the minimization problem:

$$TVF_\lambda(v) = \arg \min_u TV(u) + \lambda \int |v(\mathbf{x}) - u(\mathbf{x})|^2 dx,$$

where $TV(u)$ denotes the total variation of u and λ is a given Lagrange multiplier. The minimum of the above minimization problem exists and is unique. The parameter λ is related to the noise statistics and controls the degree of filtering of the obtained solution.

THEOREM 3. *The method noise of the total variation minimization is*

$$u(\mathbf{x}) - TVF_\lambda(u)(\mathbf{x}) = -\frac{1}{2\lambda} \text{curv}(TVF_\lambda(u))(\mathbf{x}).$$

As in the anisotropic case, straight edges are maintained because of their small curvature. However, details and texture can be over smoothed if λ is too small.

Neighborhood Filtering: The previous filters are based on a notion of spatial neighborhood or proximity. *Neighborhood filters* instead take into account gray-level values to define neighboring pixels. In the simplest and more extreme case, the denoised value at pixel i is an average of values at pixels that have a gray-level value close to $u(i)$. The gray-level neighborhood is therefore

$$B(i, h) = \{j \in I \mid u(i) - h < u(j) < u(i) + h\}.$$

This is a fully nonlocal algorithm, since pixels belonging to the whole image are used for the estimation at pixel i . This algorithm can be written in a more continuous form:

$$NF_h u(\mathbf{x}) = \frac{1}{C(\mathbf{x})} \int_\Omega u(\mathbf{y}) e^{-\frac{|u(\mathbf{y}) - u(\mathbf{x})|^2}{h^2}} d\mathbf{y},$$

where $\Omega \subset \mathbb{R}^2$ is an open and bounded set, and $C(\mathbf{x}) = \int_\Omega e^{-\frac{|u(\mathbf{y}) - u(\mathbf{x})|^2}{h^2}} d\mathbf{y}$ is the normalization factor.

The Yaroslavsky neighborhood filters⁴⁶ consider mixed neighborhoods $B(i, h) \cap B_\rho(i)$, where $B_\rho(i)$ is a ball of center i and radius ρ . So the method takes an average of the values of pixels that are both close in gray-level and spatial distance. This filter can be easily written in a continuous form as

$$YNF_{h,\rho} u(\mathbf{x}) = \frac{1}{C(\mathbf{x})} \int_{B_\rho(\mathbf{x})} u(\mathbf{y}) e^{-\frac{|u(\mathbf{y}) - u(\mathbf{x})|^2}{h^2}} d\mathbf{y},$$

where $C(\mathbf{x}) = \int_{B_\rho(\mathbf{x})} e^{-\frac{|u(\mathbf{y}) - u(\mathbf{x})|^2}{h^2}} d\mathbf{y}$ is the normalization factor. More recent versions, namely the *SUSAN filter*⁴⁰ and the *Bilateral filter*⁴² weigh the distance to the reference pixel $\mathbf{x}$ instead of considering a fixed spatial neighborhood.

In the next theorem, we compute the asymptotic expansion of the Yaroslavsky neighborhood filter when $\rho, h \rightarrow 0$ have the same order of magnitude. The proof and generalizations of this result can be found in Buades et al.⁷

THEOREM 4. Suppose $u \in C^2(\Omega)$, and let $\rho, h, \alpha > 0$ such that $\rho, h \rightarrow 0$ and $h = O(\rho)$. Let us consider the continuous function $\tilde{g}$ defined by $\tilde{g}(t) = \frac{1}{3} \frac{te^{-t^2}}{E(t)}$, for $t \neq 0$, $\tilde{g}(0) = \frac{1}{6}$, where $E(t) = 2 \int_0^t e^{-s^2} ds$. Let $\tilde{f}$ be the continuous function defined by $\tilde{f}(t) = 3\tilde{g}(t) + \frac{3\tilde{g}(t)}{t^2} - \frac{1}{2t^2}$, $\tilde{f}(0) = \frac{1}{6}$. Then, for $\mathbf{x} \in \Omega$,

$$\text{YNF}_{h,\rho} u(\mathbf{x}) - u(\mathbf{x}) \simeq \left[\tilde{g} \left(\frac{\rho}{h} |Du(\mathbf{x})| \right) u_{\xi\xi}(\mathbf{x}) + \tilde{f} \left(\frac{\rho}{h} |Du(\mathbf{x})| \right) u_{\eta\eta}(\mathbf{x}) \right] \rho^2$$

According to Theorem 4 the Yaroslavsky neighborhood filter acts as an evolution PDE with two terms. The first term is proportional to the second derivative of u in the direction ξ , which is tangent to the level line passing through $\mathbf{x}$. The second term is proportional to the second derivative of u in the direction η , which is orthogonal to the level line passing through $\mathbf{x}$.

The weighting coefficient of the tangent diffusion, $u_{\xi\xi}$, is given by $\tilde{g}(\frac{\rho}{h}|Du|)$. The function $\tilde{g}$ is positive and decreasing. Thus, there is always diffusion in that direction. The weight of the normal diffusion, $u_{\eta\eta}$, is given by $\tilde{f}(\frac{\rho}{h}|Du|)$. As the function $\tilde{f}$ takes positive and negative values (see Figure 1), the filter behaves as a filtering/enhancing algorithm in the normal direction depending on $|Du|$. The intensity of the filtering in the tangent diffusion and the enhancing in the normal diffusion tend to zero when the gradient tends to infinity. Thus, points with a very large gradient are not altered.

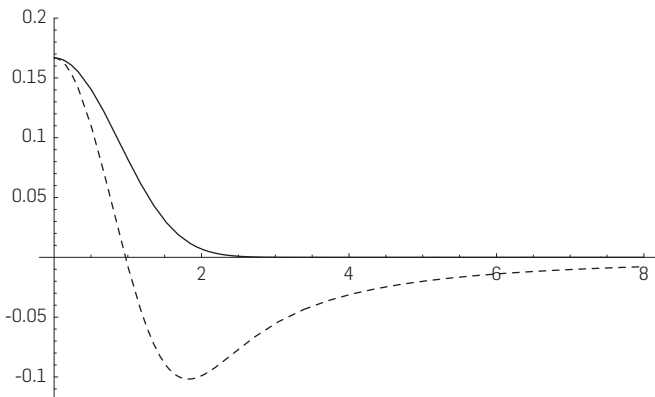
The neighborhood filter asymptotically behaves as the Perona–Malik equation,³⁵ also creating shocks inside smooth regions (see Buades et al.⁷ for more details on this comparison).

3. NL-MEANS ALGORITHM

Given a discrete noisy image $v = \{v(i) \mid i \in I\}$, the estimated value $NL[v](i)$, for a pixel i , is computed as weighted average of all the pixels in the image:

$$NL[v](i) = \sum_{j \in I} w(i, j) v(j),$$

Figure 1. Magnitude of the tangent diffusion (continuous line) and normal diffusion (dashed line –) of Theorem 4.



where the family of weights $\{w(i, j)\}_j$ depend on the similarity between the pixels i and j and satisfy the usual conditions $0 \leq w(i, j) \leq 1$ and $\sum_j w(i, j) = 1$.

The similarity between two pixels i and j depends on the similarity of the intensity gray level vectors $v(\mathcal{N}_i)$ and $v(\mathcal{N}_j)$, where $\mathcal{N}_k$ denotes a square neighborhood of fixed size centered at a pixel k . This similarity is measured as a decreasing function of the weighted Euclidean distance, $\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,a}^2$, where $a > 0$ is the standard deviation of the Gaussian kernel. The expectation of the Euclidean distance of the noisy neighborhoods is

$$E \|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,a}^2 = \|u(\mathcal{N}_i) - u(\mathcal{N}_j)\|_{2,a}^2 + 2\sigma^2.$$

This equality shows the robustness of the algorithm since in expectation the Euclidean distance preserves the order of similarity between pixels.

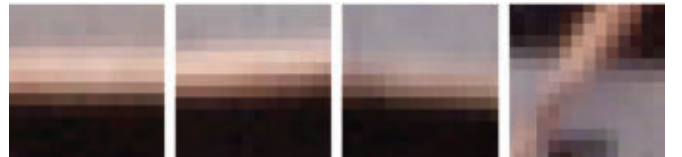
The pixels with a color neighborhood similar to $v(\mathcal{N}_i)$ have larger weights in the average, see Figure 2. These weights are defined by

$$w(i, j) = \frac{1}{Z(i)} e^{-\frac{\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,a}^2}{h^2}},$$

$$Z(i) = \sum_j e^{-\frac{\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,a}^2}{h^2}},$$

where $Z(i)$ is the normalizing constant and the parameter h controls the decay of the weights as a function of the Euclidean distances. NL-means not only compares the gray level at a single point but the geometrical configuration in a whole neighborhood. This fact allows a more robust comparison than neighborhood filters (see Figure 2). This figure illustrates how NL-means implicitly chooses a best averaging neighborhood depending on the image structure in a wide neighborhood.

Figure 2. q1 and q2 have a large weight in NL-means because their similarity windows are similar to that of p, while the weight $w(p, q3)$ is much smaller.



In Section 2, we have computed explicitly the method noise of the local smoothing filters. These formulas are corroborated by the visual experiments of Figure 3. This figure displays the method noise for the standard image Boat, that is, the difference $u - D_h(u)$, where the parameter h is been fixed in order to remove a noise with standard deviation 2.5. The method noise helps us in understanding the performance and limitations of the denoising algorithms, since removed details or texture correspond to a large method noise. We see in Figure 3 that the NL-means method noise does not present noticeable geometrical structures. Figure 4 explains this property since it shows how the NL-means algorithm chooses a weighting configuration adapted to the local and nonlocal geometry of the image.

4. NL-MEANS CONSISTENCY

Under stationarity assumptions, for a pixel i , the NL-means algorithm converges to the conditional expectation of i once observed a neighborhood of it. In this case, the stationarity conditions amount to say that as the size of the image grows, we can find many similar patches for all the details of the image.

Let V be a random field and suppose that the noisy image v is a realization of V . Let Z denote the sequence of random variables $Z_i = \{Y_i, X_i\}$ where $Y_i = V(i)$ is real valued and $X_i = V(\mathcal{N}_i \setminus \{i\})$ is $\mathbb{R}^p$ valued. The NL-means is an estimator of the conditional expectation $r(i) = E[Y_i | X_i = v(\mathcal{N}_i \setminus \{i\})]$.

THEOREM 5 (CONDITIONAL EXPECTATION THEOREM). *Let $Z = \{V(i), V(\mathcal{N}_i \setminus \{i\})\}$ for $i = 1, 2, \dots$ be a strictly stationary and mixing process. Let NL_n denote the NL-means algorithm applied to the sequence $Z_n = \{V(i), V(\mathcal{N}_i \setminus \{i\})\}_{i=1}^n$. Then for $j \in \{1, \dots, n\}$,*

$$|NL_n(j) - r(j)| \rightarrow 0 \quad a.s.$$

The full statement of the hypothesis of the theorem and its proof can be found in a more general framework in Roussas.³⁸ This theorem tells us that the NL-means algorithm corrects the noisy image rather than trying to separate the noise (oscillatory) from the true image (smooth).

In the case that an additive white noise model is assumed, the next result shows that the conditional expectation is the function of $V(\mathcal{N}_i \setminus \{i\})$ that minimizes the mean square error with the true image u .

THEOREM 6. *Let V, U, N be random fields on I such that $V = U + N$, where N is a signal-independent white noise. Then, the following statements hold good.*

- (i) $E[V(i) | X_i = x] = E[U(i) | X_i = x]$ for all $i \in I$ and $x \in \mathbb{R}^p$.
- (ii) *The expected random variable $E[U(i) | V(\mathcal{N}_i \setminus \{i\})]$ is the function of $V(\mathcal{N}_i \setminus \{i\})$ that minimizes the mean square error*

$$\min_g E[U(i) - g(V(\mathcal{N}_i \setminus \{i\}))]^2$$

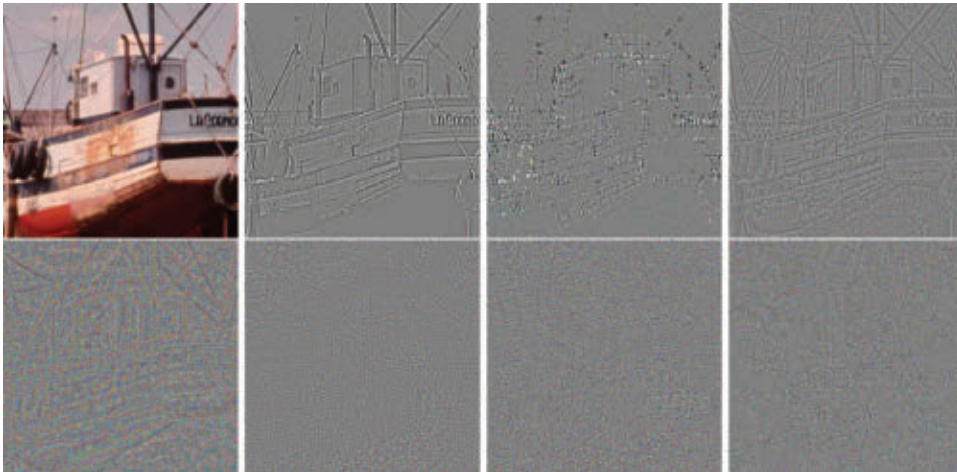
Similar optimality theoretical results have been obtained in Ordentlich et al.³⁴ and presented for the denoising of binary images.

5. DISCUSSION AND EXPERIMENTATION

In this section, we compare the local smoothing filters, the wavelet thresholding algorithms,¹⁷ sliding DCT Wiener filter,⁴⁶ and the NL-means algorithm. The wavelet thresholding and the sliding DCT algorithms yield state-of-the-art results among frequency domain filters.

For computational purposes of the NL-means algorithm, the search of similar windows was restricted to a larger “search window” with size $S \times S$ pixels. In all experiments, the search window has 21×21 pixels and the similarity square neighborhood 3×3 pixels for color images and 5×5 pixels for gray images. When denoising a color

Figure 3. Image method noise. From left to right and from top to bottom: original image, Gaussian convolution, anisotropic filtering, total variation minimization, neighborhood filter, translation invariant wavelet thresholding, DCT sliding window Wiener filter, and the NL-means algorithm. The parameters have been set for each method to remove a method noise with variance $\sigma^2 = 2.5^2$.



image, the whole color patch containing the red, green, and blue pixels is compared. Thus, a single weight is obtained for any pair of pixels and used for the denoising of the three channels at this pixel.

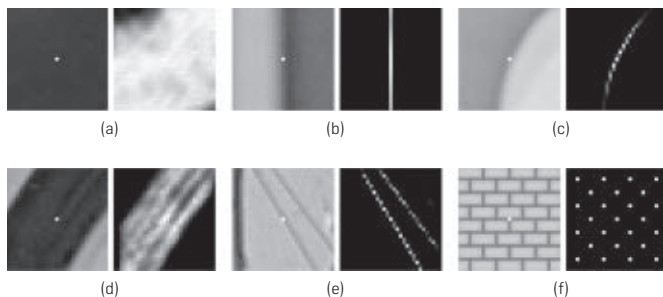
The filtering parameter h has been fixed to 0.8σ when a noise of standard deviation σ is added. Due to the fast decay of the exponential kernel, large Euclidean distances lead to nearly zero weights acting as an automatic threshold (Figure 4).

Visual Comparison: Visual criteria remains essential to decide if the quality of the image has been improved by the denoising method. We display some denoising experiences comparing the NL-means algorithm with local smoothing and frequency domain filters. All experiments have been simulated by adding a Gaussian white noise of standard deviation σ to the true image. The objective is to compare the visual quality of the restored images, the nonpresence of artifacts, and the correct reconstruction of edges, texture, and details.

Due to the nature of the algorithm, the most favorable case for NL-means is the textured or periodic case. In this situation, for every pixel i , we can find a large set of samples with a very similar configuration. See Figure 4f for an example of the weight distribution of the NL-means algorithm for a periodic image. Figure 5 illustrates the performance of the NL-means for a natural texture and Figure 6 for an artificial periodic pattern.

Natural images also have enough redundancy to be restored by NL-means, see Figure 7. Flat zones present a huge number of similar configurations lying inside the same object, see Figure 4a. Straight or curved edges have a complete line of pixels with similar configurations, see Figure 4b, c. In addition, natural images allow us to find many similar configurations in far away pixels, as Figure 4f

Figure 4. On the right-hand side of each pair, we display the weight distribution used to estimate the central pixel of the left image by the NL-means algorithm. (a) In flat zones, the weights are distributed as a convolution filter (as a Gaussian convolution). (b) On straight edges, the weights are distributed in the direction of the edge (like for AF). (c) On curved edges, the weights favor pixels belonging to the same contour or level line, which is a strong improvement with respect to AF. (d) In a flat neighborhood, the weights are distributed in a gray-level neighborhood (like for a neighborhood filter). In the cases of (e) and (f), the weights are distributed across the more similar configurations, even though they are far away from the observed pixel. This shows a behavior similar to a nonlocal neighborhood filter or to an ideal Wiener filter.



shows. Figure 7 shows an experiment on a natural image. This experience must be compared with Figure 3, where we display the method noise of the original image. The blurred or degraded structures of the restored images coincide with the noticeable structures of its method noise. Figure 8 shows that the frequency domain filters are well adapted to the recovery of oscillatory patterns. Although some artifacts are noticeable in both solutions, the stripes are well reconstructed. The DCT transform seems to be more adapted to this type of texture, and stripes are little better reconstructed. NL-means also performs well on this type of texture, due to its high degree of redundancy.

Noise-to-Noise Criterion: The noise-to-noise principle requires that a denoising algorithm transforms a white noise into white noise. This paradoxical requirement seems

Figure 5. NL-means denoising experiment with a Brodatz texture image. Left: Noisy image with standard deviation 30. Right: NL-means restored image. The Fourier transform of the noisy and restored images show how main features are preserved even at high frequencies.

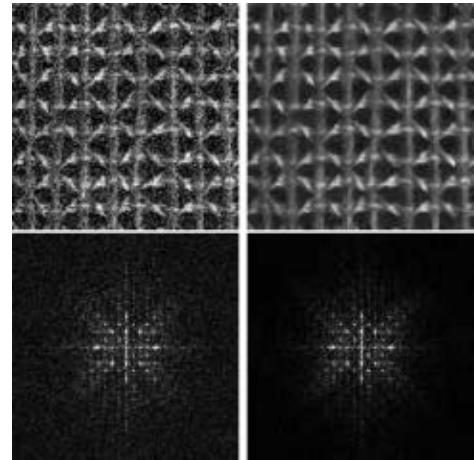


Figure 6. Denoising experience on a periodic image. From left to right and from top to bottom: noisy image (standard deviation 35), total variation minimization, neighborhood filter, translation invariant wavelet thresholding, DCT sliding window Wiener filtering, and NL-means.

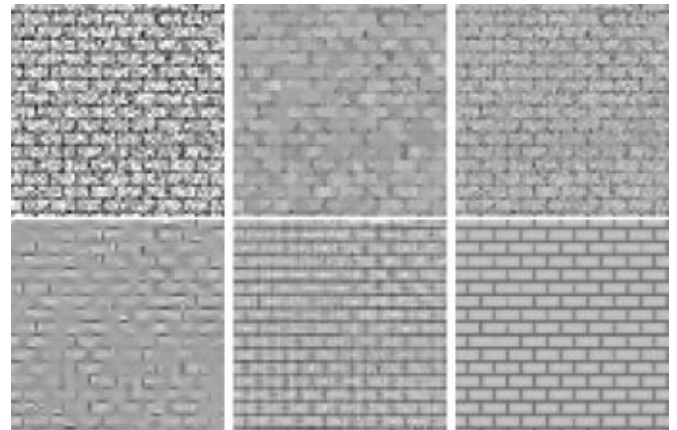


Figure 7. Denoising experience on a natural image. From left to right and from top to bottom: noisy image (standard deviation $\sigma = 20$), Gaussian convolution ($h = 1.8$), anisotropic filter ($h = 2.4$), total variation ($\lambda = 0.04$), the Yaroslavsky neighborhood filter ($\rho = 7$, $h = 28$), and the NL-means algorithm. Parameters have been set for each algorithm so that the removed quadratic energy is equal to the energy of the added noise.



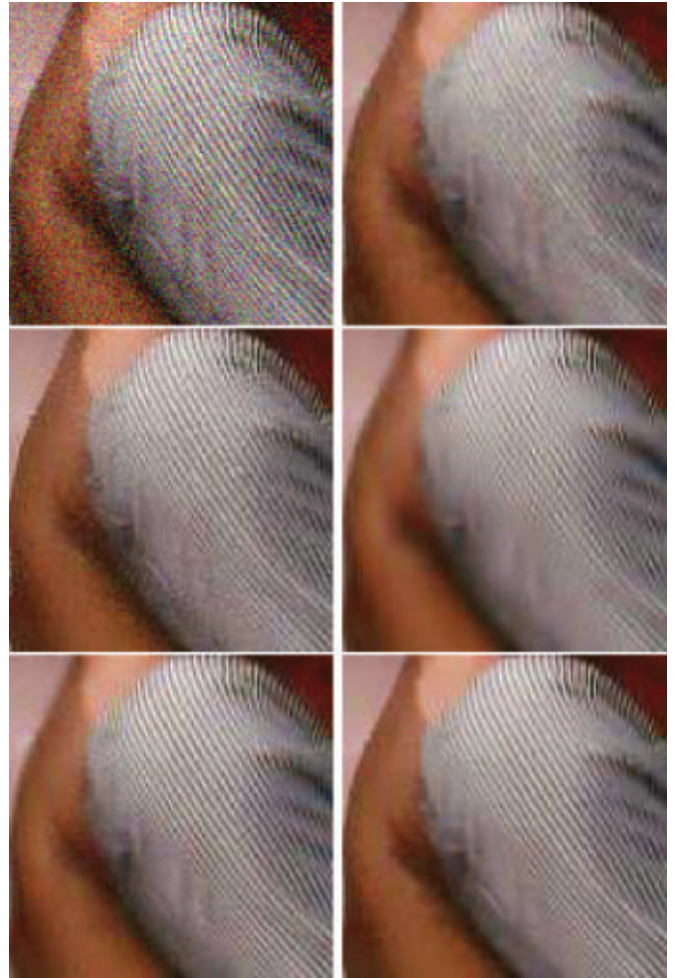
to be the best way to characterize artifact-free algorithms. The transformation of a white noise into any correlated signal creates structure and artifacts. Only white noise is perceptually devoid of visual structure, a principle first stated by Attneave.⁴ Figure 9 shows how denoising methods transform a white noise.

The convolution with a Gauss kernel is equivalent to the product in the Fourier domain with a Gauss kernel of inverse standard deviation. Therefore, convolving the noise with a kernel reinforces the low frequencies and cancels the high ones. Thus, the filtered noise actually shows big grains due to its prominent low frequencies.

Noise filtered by a wavelet thresholding is no more a white noise. The few coefficients with a magnitude larger than the threshold are spread all over the image. The pixels that do not belong to the support of one of these coefficients are set to zero. The visual result is a constant image with superposed wavelets as displayed in Figure 9. It is easy to prove that the denoised noise is spatially highly correlated.

Given a noise realization, the filtered value by the neighborhood filter at a pixel i only depends on its value $n(i)$ and the parameters h and ρ . The neighborhood filter averages noise values at a distance from $n(i)$ less or equal than h . Thus, when the size ρ of the neighborhood increases, by the law of large numbers the filtered value tends to the expectation of the Gauss distribution restricted to the interval $(n(i) - h, n(i) + h)$. This filtered value is therefore a deterministic function of $n(i)$ and h . Independent random variables are mapped by a deterministic function on independent variables. Thus the *noise-to-noise* requirement is asymptotically satisfied by the neighborhood filter. NL-means satisfies the *noise-to-noise* principle in a similar manner. However, a mathematical statement and proof of this property are more intricate and we shall skip them.

Figure 8. Denoising experience on a natural image. From left to right and from top to bottom: noisy image (standard deviation 35), total variation minimization, neighborhood filter, translation invariant wavelet thresholding, DCT sliding window Wiener filter, and NL-means.



Numerical Comparison: Table 1 displays the mean square error for the denoising experiments given in the paper. This numerical measurement is the most objective one, since it does not rely on any visual interpretation. However, this error is not computable in a real problem and a small mean square error does not assure a high visual quality. So all above-discussed criteria seem necessary to compare the performance of denoising algorithms.

6. EXTENSIONS SINCE 2005

Our conclusions on the better denoising performance of nonlocal methods with respect to former state-of-the-art algorithms such as the total variation or the wavelet thresholding have been widely accepted. The “method noise” methodology to compare the denoising performance has been adopted ever since.

The NL-means algorithm is easily extended to the denoising of image sequences and video. The denoising algorithm involves indiscriminately pixels not belonging only to same frame but also to all frames in the image.

Figure 9. The noise-to-noise criterion. From left to right and from top to bottom: original noise image of standard deviation 20, Gaussian convolution, anisotropic filtering, total variation, neighborhood filter, translation invariant wavelet thresholding, DCT sliding window Wiener filter, and NL-means. Parameters have been fixed for each method so that the noise standard deviation is reduced by a factor 4.

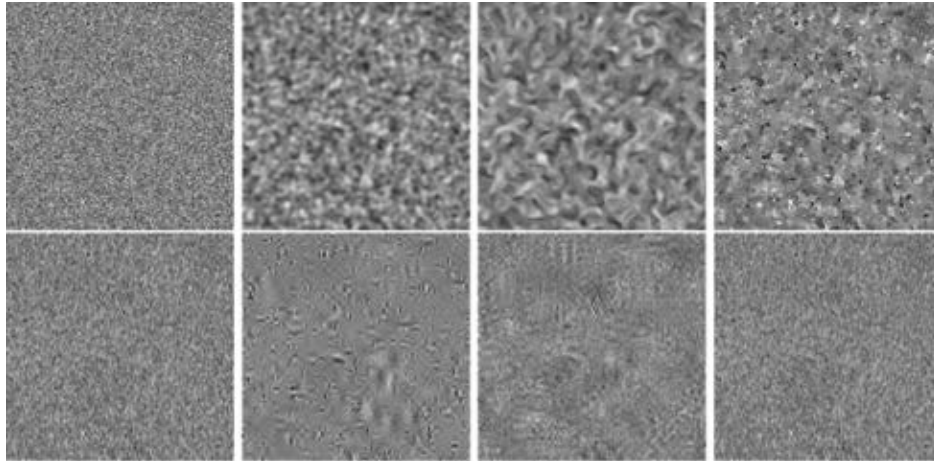


Table 1. Mean square error table.

Image	σ	GF	AF	TV	YNF	DCT	Wav. Thresh	NL-Means
Boat	8	53	38	39	39	33	28	23
Lena	20	120	114	110	129	105	81	68
Barbara	25	220	216	186	176	111	135	72
Baboon	35	507	418	365	381	396	365	292
Wall	35	580	660	721	598	325	712	59

A smaller mean square error indicates that the estimate is closer to the original image. The numbers have to be compared on each row. The square of the number on the left-hand column gives the real variance of the noise. By comparing this square to the values on the same row, it is quickly checked that all studied algorithms indeed perform some denoising. This is a sanity check! In general, the comparison performance corroborates the previously mentioned quality criteria.

The algorithm favors pixels with a similar local configuration, as the similar configurations move, so do the weights. Thus, the algorithm is able to follow the similar configurations when they move without any explicit motion computation (see Figure 10). This is not the case of classical movie denoising algorithms, which are motion compensated (see Buades et al.⁹ for more details on this discussion). The very same idea on movie denoising can be applied for *super-resolution*, an image zooming method by which several frames from a video, or several low resolution photographs, can be fused into a larger image.^{20, 37}

Improvements or adaptations of the NL-means algorithm have been proposed for the denoising of several types of data: in fluorescence microscopy,⁵ cryon microscopy,¹⁵ magnetic resonance imaging (MRI),³¹ and 3D data set points.⁴⁷

Most successful improvement of NL-means combine the nonlocal principle with former classic algorithms and have indeed shown an improved denoising performance. Probably the best performing methods so far are the hybrid method BM3D proposed in Dabov et al.¹⁴ and the NL-PCA proposed in Zhang et al.⁴⁸ Both algorithms combine not less than block-matching, a linear transform thresholding,

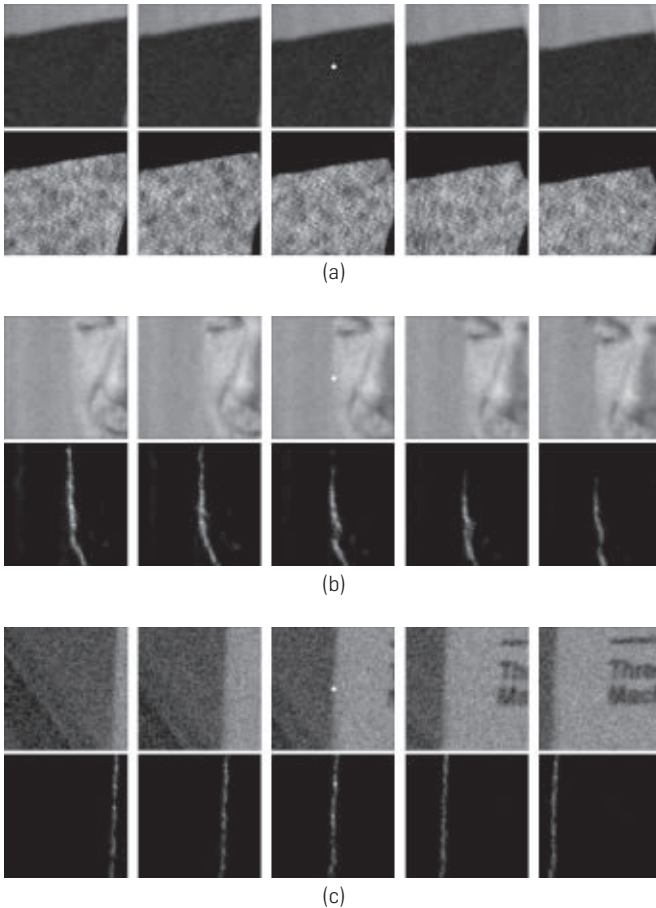
and Wiener filtering.

The NL-means algorithm has also expanded to most image processing tasks: *Demosaicking*, which is the operation that transforms the “R or G or B” raw image in each camera into an “R and G and B” image;^{10, 30} *movie colorization*^{21, 26}; *image inpainting* by proposing a nonlocal image inpainting variational framework with a unified treatment of geometry and texture³ (see also Wong and Orchard⁴⁴); *Zooming* by a fractal like technique where examples are taken from the image itself at different scales¹⁸; *movie flicker stabilization*¹⁶ that compensates spurious oscillations in the colors of successive frames.

NL-means is a computationally demanding algorithm. Several papers have proposed fast and extremely fast (linear) implementations, by block preselection,²⁹ by Gaussian KD-trees to classify image blocks,¹ by SVD,³³ by using the FFT to compute correlation between blocks⁴³, and by statistical arguments.¹³ The statistical validity of the NL-means algorithm is wide open. See Ebrahimi and Vrscay,¹⁸ Kervrann et al.,²⁴ and Thacker et al.⁴¹ (where a Bayesian interpretation is proposed) or Xu et al.⁴⁵ (where a bias of NL-means is corrected).

The relationship of neighborhood filters to classic

Figure 10. Weight distribution of NL-means applied to a movie. In (a), (b), and (c) the first row shows a five frames image sequence. In the second row, the weight distribution used to estimate the central pixel (in white) of the middle frame is shown. The weights are equally distributed over the successive frames, including the current one. They actually involve all the candidates for the motion estimation instead of picking just one per frame. The aperture problem can be taken advantage of for a better denoising performance by involving more pixels in the average.



local PDE's has been discussed in Buades et al.,^{7, 8} leading to an adaptation of NL-means that avoids the stair casing effect. Yet, the main interest has shifted to defining *nonlocal PDEs*. The extension of the NL-means method to define nonlocal image-adapted differential operators and nonlocal variational methods starts with Kindermann et al.,²⁵ who propose to perform denoising and deblurring by nonlocal functionals. Several articles on deblurring have followed this variational line^{22, 23, 32} (for image segmentation) and Lou et al.²⁸ for deconvolution and tomographic reconstruction.

A particular notion of nonlocal PDE has emerged, whose coefficients are actually image dependent. For instance, in Elmoataz et al.,²¹ the image colorization is viewed as the minimization of a discrete partial differential functional on the weighted block graph. Thus, it can be seen either as a nonlocal heat equation on the image or as a local heat equation on the space of image patches.

The exploration of image redundancy and its application to image restoration has led to new attempts at sparse image representations by block dictionaries.²⁴ Algorithms and application have been developed by Chatterjee and Milanfar,¹¹ Mairal et al.,³⁰ and Protter and Elad.³⁶

Acknowledgment

This work was partially financed by MCYIT grant number TIN2008-04752. G

References

- Adams, A., Gelfand, N., Dolson, J., Levoy, M. Gaussian kd-trees for fast high-dimensional filtering. *ACM Trans. Graphics* 28, 3 (2009), 1–12.
- Alvarez, L., Lions, P., Morel, J. Image selective smoothing and edge detection by nonlinear diffusion. II. *SIAM J. Numer. Anal.* 29, 3 (1992), 845–866.
- Arias, P., Caselles, V., Sapiro, G. A variational framework for non-local image inpainting. In *Proceedings of the 7th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMVCPR'09)* (Bonn, Germany, August 24–27, 2009), Springer, Heidelberg.
- Attneave, F. Some informational aspects of visual perception. *Psychol. Rev.* 61, 3 (1954), 183–193.
- Boulanger, J., Sibarita, J., Kervrann, C., Boutheimy, P. Non-parametric regression for patch-based fluorescence microscopy image sequence denoising. In *5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2008* (Paris, France, May 14–17, 2008), ISBI 2008, 748–751.
- Buades, A., Coll, B., Morel, J. A review of image denoising algorithms, with a new one. *Multiscale Model. Simul.* 4, 2 (2005), 490–530.
- Buades, A., Coll, B., Morel, J. Neighborhood filters and PDE's. *Numer. Math.* 105, 1 (2006), 1–34.
- Buades, A., Coll, B., Morel, J. The staircasing effect in neighborhood filters and its solution. *IEEE Trans. I.P.* 15, 6 (2006), 1499–1505.
- Buades, A., Coll, B., Morel, J. Nonlocal image and movie denoising. *Int. J. Comput. Vision* 76, 2 (2008), 123–139.
- Buades, A., Coll, B., Morel, J., Sbert, C. Self-similarity driven color demosaicking. *IEEE Trans. I.P.* 18, 6 (2009), 1192–1202.
- Chatterjee, P., Milanfar, P. Image denoising using locally learned dictionaries. *Comput. Imag. VII, SPIE* 7246 (2009), 72460V–72460V.
- Coifman, R., Donoho, D. Translation-invariant de-noising. In *Wavelets and Statistics. Lecture Notes in Statistics* (Springer Verlag, New York, 1995), 125–150.
- Coupé, P., Yger, P., Barillot, C. Fast non local means denoising for 3D MR images. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2006. Lecture Notes in Computer Science* (Springer, New York, 2006), 33–40.
- Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Trans. I.P.* 16, 8 (2007), 2080–2095.
- Darbon, J., Cunha, A., Chan, T., Osher, S., Jensen, G. Fast nonlocal filtering applied to electron cryomicroscopy. In *5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro* (Paris, France, May 14–17, 2008), 1331–1334.
- Delon, J., Desolneux, A. Stabilization of flicker-like effects in image sequences through local contrast correction. *SIAM J. Imag. Sci.* 3, 4 (2010), 703–734.
- Donoho, D. De-noising by soft-thresholding. *IEEE Trans. Inf. Theory* 41 (1995), 613–627.
- Ebrahimi, M., Vrscaj, E. Solving the inverse problem of image zooming using “Self-Examples”. In *Volume 4633 of Lecture Notes in Computer Science* (2007), 117.
- Efros, A., Leung, T. Texture synthesis by non parametric sampling. In *Volume 2 of Proceedings of the 7th IEEE International Conference on Computer Vision* (Corfu, Greece, 1999), 1033–1038.
- Elad, M., Datsenko, D. Example-based regularization deployed to super-resolution reconstruction of a single image. *Comput. J.* 50, 4 (Apr. 2007), 1–16.
- Elmoataz, A., Lezoray, O., Bougleux, S., Ta, V. Unifying local and nonlocal processing with partial difference operators on weighted graphs. In *International Workshop on Local and Non-Local Approximation in Image Processing* (Lausanne, Switzerland, August 23–24, 2008).
- Gilboa, G., Osher, S. Nonlocal linear image regularization and supervised segmentation. *Multiscale Model. Simul.* 6, 2 (2008), 595–630.
- Jung, M., Vese, L. Nonlocal variational image deblurring models in the presence of Gaussian or impulse noise. *Scale Space and Variational Methods in Computer Vision* (2009), 401–412.
- Kervrann, C., Boulanger, J., Coupe, P. Bayesian non-local means filter, image redundancy and adaptive dictionaries for noise removal. In *Volume 4485 of Lecture Notes in Computer Science* (Ischia, Italy, May 30–June 2, 2007), 520.

25. Kindermann, S., Osher, S., Jones, P. Deblurring and denoising of images by nonlocal functionals. *SIAM MMS* 4, 4 (2006), 1091–1115.
26. Lezoray, O., Ta, V., Elmoataz, A. Nonlocal graph regularization for image colorization. In *International Conference on Pattern Recognition* (Tampa, FL, December 8–11, 2008).
27. Lindenbaum, M., Fischer, M., Bruckstein, A. On Gabor's contribution to image enhancement. *Pattern Recognit.* 27, 1 (1994), 1–8.
28. Lou, Y., Zhang, X., Osher, S., Bertozzi, A. Image recovery via nonlocal operators. *J. Sci. Comput.* 42, 2 (2010), 185–197.
29. Mahmoudi, M., Sapiro, G. Fast image and video denoising via nonlocal means of similar neighborhoods. *IEEE Signal Process. Lett.* 12, 12 (2005), 839–842.
30. Mairal, J., Elad, M., Sapiro, G. Sparse representation for color image restoration. *IEEE Trans. I.P.* 17, 1 (2007), 53–69.
31. Manjón, J., Carbonell-Caballero, J., Lull, J., García-Martí, G., Martí-Bonmatí, L., Robles, M. MRI denoising using Non-Local Means. *Med. Image Anal.* 12, 4 (2008), 514–523.
32. Mignotte, M. A non-local regularization strategy for image deconvolution. *Pattern Recognit. Lett.* 29, 16 (2008), 2206–2212.
33. Orchard, J., Ebrahimi, M., Wong, A. Efficient non-local-means denoising using the SVD. In *Proceedings of IEEE International Conference on Image Processing* (San Diego, CA, October 12–15, 2008).
34. Orntlich, E., Seroussi, G., Verdu, M.W.S., Weissman, T. A discrete universal denoiser and its application to binary images. In Volume 1 of *Proceedings of IEEE International Conference on Image Processing* (Barcelona, Spain, 2003), 117–120.
35. Perona, P., Malik, J. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. PAMI* 12, 7 (1990), 629–639.
36. Protter, M., Elad, M. Image sequence denoising via sparse and redundant representations. *IEEE Trans. Image Process.* 18, 1 (2009), 27–35.
37. Protter, M., Elad, M., Takeda, H., Milanfar, P. Generalizing the non-local-means to super-resolution reconstruction. *IEEE Trans. Image Process.* 18, 1 (January 2009), 36–51.
38. Roussas, G. Nonparametric regression estimation under mixing conditions. *Stochastic Process. Appl.* 36 (1990), 107–116.
39. Rudin, L., Osher, S., Fatemi, E. Nonlinear total variation based noise removal algorithms. *Physica D* 60 (1992), 259–268.
40. Smith, S., Brady, J. SUSAN: A new approach to low level image processing. *Int. J. Comput. Vision* 23, 1 (1997), 45–78.
41. Thacker, N., Manjon, J., Bromiley, P. A statistical interpretation of non-local means. In *5th International Conference on Visual Information Engineering* (Xian, China, 2008), 250–255.
42. Tomasi, C., Manduchi, R. Bilateral filtering for gray and color images. In *International Conference on Computer Vision* (Bombay, India, 1998), 839–846.
43. Wang, J., Guo, Y., Ying, Y., Liu, Y., Peng, Q. Fast non-local algorithm for image denoising. In *IEEE International Conference on Image Processing* (Atlanta, GA, October 8–11, 2006), 1429–1432.
44. Wong, A., Orchard, J. A nonlocal-means approach to exemplar-based inpainting. In *IEEE International Conference on Image Processing*, 2600–2603, 2008.
45. Xu, H., Xu, J., Wu, F. On the biased estimation of Nonlocal Means filter. In *IEEE International Conference on Multimedia and Expo* (San Diego, CA, October 12–15, 2008), 1149–1152.
46. Yaroslavsky, L.P. *Digital Picture Processing*. Springer, New York, 1985.
47. Yoshizawa, S., Belyaev, A., Seidel, H. Smoothing by example: Mesh denoising by averaging with similarity-based weights. In *IEEE International Conference on Shape Modeling and Applications* (Matsushima, Japan, June 14–16, 2006), 9–9.
48. Zhang, L., Dong, W., Zhang, D., Shi, G. Two-stage image denoising by principal component analysis with local pixel grouping. *Pattern Recognit.* 43, 4 (2010), 1531–1549.

Antoni Buades (toni.buades@uib.es),
Université Paris Descartes, 45, rue Saints
Pères, Paris, France.

Bartomeu Coll (tomeu.coll@uib.es),
Dpt Matemàtiques Informàtica,
Universitat Illes Balears, Ctra
Valldemossa km 7.5, Palma
de Mallorca, Spain.

Jean-Michel Morel (morel@cmla.
enscachan.fr), CMLA, ENS Cachan,
61 av. Président Wilson, Cachan 94235,
France.

© 2011 ACM 0001-0782/11/05 \$10.00

Take Advantage of ACM's Lifetime Membership Plan!

- ◆ **ACM Professional Members** can enjoy the convenience of making a single payment for their entire tenure as an ACM Member, and also be protected from future price increases by taking advantage of **ACM's Lifetime Membership** option.
- ◆ **ACM Lifetime Membership** dues may be tax deductible under certain circumstances, so becoming a Lifetime Member can have additional advantages if you act before the end of 2011. (Please consult with your tax advisor.)
- ◆ Lifetime Members receive a certificate of recognition suitable for framing, and enjoy all of the benefits of **ACM Professional Membership**.

Learn more and apply at:
<http://www.acm.org/life>



Association for
Computing Machinery

Advancing Computing as a Science & Profession

CAREERS

Center for Genomics and Bioinformatics (CGB) Indiana University Genome Computing Unit Leader

The **Center for Genomics and Bioinformatics (CGB)** (<http://cgb.indiana.edu>) is a research center at **Indiana University**, Bloomington that carries out research in all aspects of genomics and bioinformatics. The CGB Bioinformatics Group – Genome Computing Unit conducts research in the area of next generation sequencing and gene microarray data analysis, genome assembly and annotation, functional genomics and systems biology. We have an immediate vacancy for the Genome Computing Unit Leader. Responsibilities include guiding research projects; managing all aspects of the analysis of next generation sequencing and microarray projects, along with hiring, training, and supervising staff. Ideal candidate will hold a Ph.D., have a strong background in computer science and/or bioinformatics, and have extensive experience in genome data analysis.

This is a non-tenure track research faculty appointment with salary and benefits commensurate with prior experience. The position is available immediately. Those received by May 15, 2011 will be assured full consideration. To apply please submit a CV and a description of your background and interests, and arrange that 3 letters of recommendation be sent to: Search: CGB-019: Genome Computing Unit Leader, Center for Genomics and Bioinformatics, Indiana University, 1001 E. 3rd St., Bloomington IN 47405-7005.

Indiana University is an affirmative action equal opportunity employer.

Georgia Southern University Assistant Professor

Tenure-track position in Computer Sciences at the Asst Prof level to begin August 1, 2011. For the full position description and information about the department, see <http://cit.georgiasouthern.edu/cs/>. Georgia is an Open Records state. Georgia Southern University is an AA/EO institution.

Jacksonville University Visiting Assistant Professor of Computing Science

Position to begin August 2011. Expertise in programming, web app. development & related internet technologies. Master's degree required, Ph.D. preferred. To apply: Send CV, transcripts, 3 current letters of support. Email: cssearch@ju.edu, Fax: 904-256-7573. Application screening begins immediately. JU actively seeks application from individuals with diverse backgrounds & experiences. We believe that diversity enriches the workplace & the academic experience.

North Carolina State University Computer Science Department Teaching Assistant Professor

The Computer Science Department at North Carolina State University (NCSU) invites applications from outstanding committed teachers for one or more Teaching Assistant Professor positions starting August 2011. Candidates must have a Ph.D. in Computer Science or a Ph.D. in a related area along with appropriate experience, by August 15, 2011. Candidates should provide substantial evidence of interest and excellence in teaching at the university level, and leadership in curricular innovation.

Primary responsibilities will include teaching undergraduate Computer Science courses, with initial focus on lower-division courses and management of the introductory course sequences. This includes supervision of Teaching Assistants and development of (1) course and curricular materials (including interactive web-based materials), (2) classroom demonstrations, and (3) laboratory exercises. Successful candidates will interact with the regional and national community, and must be student-centered with excellent communication skills. They will also be expected to contribute to the department efforts in scholarship and service. Participation in high quality research activities centered on teaching, learning and Computer Science-related pedagogy is expected.

The Department, part of the university's College of Engineering, is one of the oldest and largest in the country, and consistently ranks among the highest nationwide as a source of new university hires by major international companies. NCSU is located in Raleigh, capital of North Carolina, which forms one vertex of the world-famous Research Triangle. The Research Triangle area is routinely recognized in

nationwide surveys as one of the best places to live in the U.S. We enjoy outstanding public schools, affordable housing, and great weather, all in the proximity of both mountains and the seashore.

Applications will be reviewed as they are received. The positions will remain open until suitable candidates are identified. Applicants are encouraged to apply by April 1, 2011. Salary will be commensurate with qualifications. Applicants should submit the following online at <http://jobs.ncsu.edu> (reference position number 102077): cover letter, curriculum vitae, teaching statement, and names and complete contact information of four references, including email addresses and phone numbers. For more information about the position and department see <http://www.csc.ncsu.edu/employment/>. Inquiries may be sent via email to: teaching_faculty_search@csc.ncsu.edu.

North Carolina State University is an equal opportunity and affirmative action employer. In addition, NC State University welcomes all persons without regard to sexual orientation. Individuals with disabilities desiring accommodations in the application process should contact the Department of Computer Science at (919) 515-2858.

Purdue University Post-doctoral Research Associate

Wanted a post-doctoral research associate to participate in a leadership role on a team to develop software for the modeling and analysis of agent-based missile defense architectures, including validation of the software. Team includes 8 other participants. US citizenship or permanent residency is required.

Knowledge of:

- ▶ Computer and communication modeling
- ▶ Cybersecurity modeling
- ▶ Optimization
- ▶ Skills in:
- ▶ Software development (Matlab, C, scripting)
- ▶ Software validation

Apply:

Contact Person: Saurabh Bagchi, Email Address: sbagchi@purdue.edu; Phone: 765-494-3362

University of New Hampshire Lecturer in Computer Science

The Department of Computer Science of the University of New Hampshire invites applications for a full-time non-tenure track Lecturer position to begin August 22, 2011. The candidate should be able to teach a variety of undergraduate computer science and information technology courses. For more information see: <http://www.cs.unh.edu/search.htm>

UNH is an AA/EEO Employer. UNH is committed to excellence through the diversity of its faculty and staff and encourages women and minorities to apply.



The Microsoft Research – University of Trento COSBI
Principal Investigator in computer science

We are seeking an outstanding, highly motivated and experienced Principal Investigator in modeling and simulation.

PhD in Computer Science and at least eight years experience of research activity in this field are required.

Skills and experience:

- international reputation, distinguished record of scientific results, large number of invited talks, high bibliometric indexes, participation in steering committees, program committees and advisory bodies;
- scientific and administrative coordination of large international projects;
- demonstrated ability to establish strong industrial collaborations, start and develop funded research programs;
- consulting experience for industries;
- strong interpersonal and research leadership, vision of the future development of biological and ecological systems modeling;
- demonstrated ability to manage stressing situations and delivering results in time according to pre-defined schedules;
- experience as CS professional is preferred.

We offer a competitive salary. This job is in Northern Italy. Other vacancies at www.cosbi.eu/index.php/open-positions

Send your application letter with statement of research interests, CV and contact information for three references to:

COSBI HR Department • Trento – Italy • hr@cosbi.eu;
Ph: +39 0461 282811 • Ref: Principal Investigator in Computer Science

Expansion of the Research School "Service-Oriented Systems Engineering" at Hasso-Plattner-Institute

Hasso-Plattner-Institute (HPI) is a privately financed institute affiliated with the University of Potsdam, Germany. The Institute's founder and benefactor Professor Hasso Plattner, who is also co-founder and chairman of the supervisory board of SAP AG, has created an opportunity for students to experience a unique education in IT systems engineering in a professional research environment with a strong practice orientation.

In 2005, HPI initiated the research school on „Service-Oriented Systems Engineering“ under the scientific supervision of Professors Jürgen Döllner, Holger Giese, Robert Hirschfeld, Christoph Meinel, Felix Naumann, Hasso Plattner, Andreas Polze, Mathias Weske and Patrick Baudisch.

We are expanding our research school and are currently seeking

8 Ph.D. students (monthly stipends 1450-1650 Euro) 2 Postdocs (monthly stipend 1850 Euro)

Positions will be available starting October 1, 2011.

The stipends are not subject to income tax.

The main research areas in the research school at HPI are:

- Self-Adaptive Service-Oriented Systems
- Operating System Support for Service-Oriented Systems
- Architecture and Modeling of Service-Oriented Systems
- Adaptive Process Management
- Services Composition and Workflow Planning
- Security Engineering of Service-Based IT Systems
- Quantitative Analysis und Optimization of Service-Oriented Systems
- Service-Oriented Systems in 3D Computer Graphics
- Service-Oriented Geoinformatics
- Human Computer Interaction for Service-Oriented Systems

Prospective candidates should have demonstrated expertise in one of the above-mentioned research areas and are invited to apply with:

- Curriculum vitae and copies of degree certificates/transcripts
- A short research proposal
- Writing samples/copies of relevant scientific papers (e. g. thesis etc.)
- Letters of recommendation

Please submit your applications by August 1, 2011 to the coordinator of the research school:

Prof. Dr. Andreas Polze
Hasso-Plattner-Institute, Universität Potsdam
Postfach 90 04 60, 14440 Potsdam, Germany

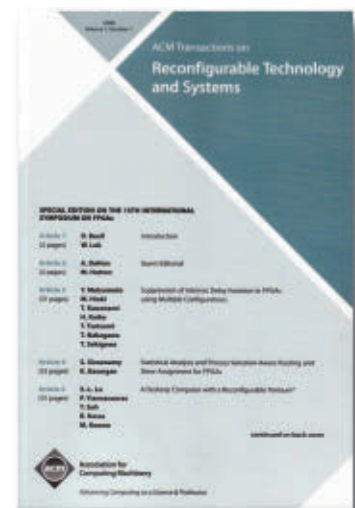
Successful candidates will be notified by September 15, 2011 and are expected to enroll into the program on October 1, 2011.

For additional information see: <http://kolleg.hpi.uni-potsdam.de>

or contact the office:
Telephone +49-331-5509-220, Telefax +49-331-5509-229
Email: office-polze@hpi.uni-potsdam.de



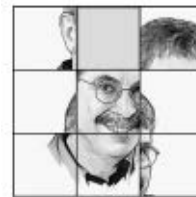
ACM Transactions on Reconfigurable Technology and Systems



This quarterly publication is a peer-reviewed and archival journal that covers reconfigurable technology, systems, and applications on reconfigurable computers. Topics include all levels of reconfigurable system abstractions and all aspects of reconfigurable technology including platforms, programming environments and application successes.

www.acm.org/trets
www.acm.org/subscribe





DOI:10.1145/1941487.1941514

Peter Winkler

Puzzled Games, Roles, Turns

Welcome to three new puzzles. Solutions to the first two will be published next month; the third is (as yet) unsolved. In each, the issue is how your intuition matches up with the mathematics.

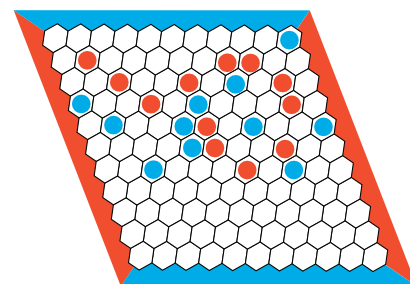
The theme is games. The twist is a little bit of role-switching, taking the randomness out of poker and putting it into chess.

1. Tired of the vagaries of the random deal, Alice and Bob embark on a game of deterministic draw poker. The 52 cards are spread out face up. Alice chooses five; then Bob chooses five. Alice can now discard any number of cards (which go out of play) and draw a like number, so she finishes with five cards. It's then Bob's turn to draw; he has the same options. All actions are deterministic and seen by both players. Finally, Alice and Bob compare hands. Since Alice had the advantage of going first, Bob is deemed the winner if the hands are equally good. Who wins with best play?

2. Clarissa, a computer science major and president of her university's chess club, decides to program her laptop to play random chess. At each position, all

possible legal moves are computed, with one chosen uniformly at random. The program is designed to quit if and only if a checkmate or stalemate is achieved or if only the two kings remain on the board. The following day, Clarissa finds the program is still running—caught in a loop. How could this happen?

3. The game of hex (see, for example, <http://mathworld.wolfram.com/GameofHex.html>) is played on an 11x11 diamond cut from a hexagonal grid; the figure here is of a game, which, with best play, will be won by the red player. Invented independently in the 1940s by two beautiful minds on different sides of the world—Piet Hein and John Nash—the game is played by alternately placing red and blue stones on the hexagonal cells. Playing red, Alice makes the first move, intending to create an uninterrupted path of red stones from left to right. Playing blue, Bob tries



to create a path of blue stones from top to bottom. It is not difficult to see that only one player can succeed, and that by the time the board is filled one player must succeed. Also not difficult to see is that the first player (Alice) has a winning strategy. Game theory tells us that either Alice or Bob has such a strategy, but it can't be Bob because an extra red stone on the board can never hurt Alice. The problem is no one has been able to devise such a strategy (either for the general $n \times n$ game or the standard 11x11 game). Your problem, however, is potentially simpler than devising a winning strategy: Prove the eminently plausible statement that one of Alice's winning first moves is the center cell.

All readers are encouraged to submit prospective puzzles for future columns to puzzled@cacm.acm.org.

Peter Winkler (puzzled@cacm.acm.org) is Professor of Mathematics and of Computer Science and Albert Bradley Third Century Professor in the Sciences at Dartmouth College, Hanover, NH.

October 22–27, 2011

A SPLASH Conference

Hilton Portland & Executive Tower

Portland, Oregon USA

ONWARD! 2011

**ACM Symposium on New Ideas in
Programming and Reflections on Software**

Chair

Robert Hirschfeld

Hasso-Plattner-Institut Potsdam, Germany

chair@onward-conference.org

Papers

Eelco Visser

Delft University of Technology, The Netherlands

papers@onward-conference.org

Workshops

Pascal Costanza

Vrije Universiteit Brussel, Belgium

workshops@onward-conference.org

Essays

David West

New Mexico Highlands University, USA

essays@onward-conference.org

Films

Bernd Bruegge

Technische Universität München, Germany

films@onward-conference.org



Association for
Computing Machinery

<http://onward-conference.org/>

The 2012 ACM Conference on Computer Supported Cooperative Work

February 11-15, 2012 | Seattle, Washington



Call for Submissions

CSCW is an international and interdisciplinary conference on technical and social aspects of communication, collaboration, and coordination. The conference addresses the design and use of technologies that affect groups, organizations, and communities. CSCW topics continue to expand as we increasingly use technologies to live, work and play with others.

This year we have adopted a new review process for papers and notes intended to increase their diversity and quality. The submission deadline is early to avoid conflicting with the CHI 2012 deadline. This enables us to include a revision cycle: Qualifying authors can revise their submissions after the initial reviews.

For more details about CSCW 2012, please review the Call for Participation on our conference homepage, www.cscw2012.org. To learn about future details as they become available, please follow us on Twitter (@acm_2012) and/or "like" us on Facebook (www.facebook.com/CSCW2012).

Conference Chairs: Steve Poltrock, Carla Simone

Papers and Notes Chairs: Gloria Mark, John Riedl, Jonathan Grudin

Also consider attending WSDM (wsdm2012.org) immediately before CSCW 2012.

Sponsored by



Association for
Computing Machinery



SIGCHI
special interest group computer human interaction

Submission Deadlines

Papers & Notes

3 June 2011

Workshops

28 July 2011

Panels

Interactive Posters

Demonstrations

CSCW Horizon

Videos

Student Volunteers

9 September 2011

Doctoral Colloquium

16 October 2011



<http://www.cscw2012.org>