

COMMUNICATIONS

CACM.ACM.ORG

OF THE

ACM

08/2011 VOL.54 NO.8



Cognitive Computing

Computing Without Processors

Business Intelligence Technology

Skinput

Storage Strife

Association for
Computing Machinery



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/>

ACM's Career & Job Center!

Are you looking for your next IT job?

Do you need Career Advice?

Visit ACM's Career & Job Center at:

<http://www.acm.org/careercenter>



The ACM Career & Job Center offers ACM members a host of career-enhancing benefits:

- A highly targeted focus on job opportunities in the computing industry
- Access to hundreds of corporate job postings
- Resume posting keeping you connected to the employment market while letting you maintain full control over your confidential information
- An advanced Job Alert system that notifies you of new opportunities matching your criteria
- Career coaching and guidance from trained experts dedicated to your success
- A content library of the best career articles compiled from hundreds of sources, and much more!



Association for
Computing Machinery

Advancing Computing as a Science & Profession

The ACM Career & Job Center is the perfect place to begin searching for your next employment opportunity!

Visit today at

<http://www.acm.org/careercenter>



Departments

- 5 **Presidents' Letter**
**From the Presidents of the IEEE
Computer Society and ACM**
By Sorel Reisman and Alain Chesnais
-
- 7 **Letters To The Editor**
**A Policy that Deters Violation
of Security Policy**
-
- 8 **In the Virtual Extension**
-
- 10 **BLOG@CACM**
**Stonebraker on
NoSQL and Enterprises**
Michael Stonebraker discusses
several reasons why NoSQL has not
caught on with enterprise users.
-
- 12 **CACM Online**
Why You Should Be Happy to Sign In!
By Scott E. Delman
-
- 31 **Calendar**
-
- 119 **Careers**

Last Byte

- 120 **Puzzled**
Uncommon Divisors
By Peter Winkler

News



- 13 **A New Benchmark
for Artificial Intelligence**
Computers are unable to defeat
the world's best Go players, but that
may change with the application
of a new strategy that promises to
revolutionize artificial intelligence.
By Kirk L. Kroeker
-
- 16 **Supercomputing's Exaflop Target**
The twin challenges of
parallelism and energy
consumption are enlivening
supercomputers' progress.
By Tom Geller
-
- 19 **Life, Translated**
The Holy Grail of language
translation is to develop a
machine-based system that can
handle the task transparently
and accurately.
By Samuel Greengard

Viewpoints



- 24 **Economic and Business Dimensions**
**Geography Is Alive and Well
in Virtual Teams**
Just because a team is virtual,
it doesn't mean geography is dead.
By Jonathon N. Cummings
-
- 27 **Education**
From Interests to Values
Computer science is not that difficult
but wanting to learn it is.
By Betsy DiSalvo and Amy Bruckman
-
- 30 **Privacy and Security**
**As Simple As Possible—
But Not More So**
An assessment of the U.S.
government's EINSTEIN project.
*By S.M. Bellovin, S.O. Bradner,
W. Diffie, S. Landau, and J. Rexford*
-
- 34 **Kode Vicious**
Storage Strife
Beware keeping data
in binary format.
By George V. Neville-Neil
-
- 36 **Viewpoint**
**Journals for Certification,
Conferences for
Rapid Dissemination**
Rethinking the role of journals
in computer science.
*By Joseph Y. Halpern
and David C. Parkes*

Practice

40 **The Robustness Principle Reconsidered**

Seeking a middle ground.
By Eric Allman

46 **Computing Without Processors**

Heterogeneous systems allow us to target our programming to the appropriate environment.

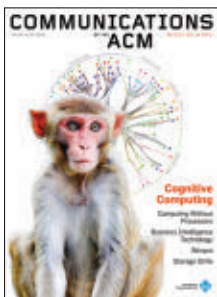
By Satnam Singh

55 **The Pain of Implementing LINQ Providers**

It's no easy task for NoSQL.
By Oren Eini



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

**About the Cover:**

Cognitive computing, say the authors of our cover story, seeks a profound, reliable understanding of the mechanism of the mind toward developing intelligent machines. Exploring the brain of the macaque monkey, their research ponders the previous work, ultimately advocating a middle

path more faithful to neuroscience than to a purely abstractionist connectionist model.

Contributed Articles

62 **Cognitive Computing**

Unite neuroscience, supercomputing, and nanotechnology to discover, demonstrate, and deliver the brain's core algorithms.

By Dharmendra S. Modha, Rajagopal Ananthanarayanan, Steven K. Esser, Anthony Ndirango, Anthony J. Sherbondy, and Raghavendra Singh

72 **Gender and Computing Conference Papers**

Women increasingly publish in ACM conference proceedings and are notably productive authors.

By J. McGrath Cohoon, Sergey Nigai, and Joseph "Jofish" Kaye

81 **Reputation Systems for Open Collaboration**

Algorithmic-based user incentives ensure the trustworthiness of evaluations of Wikipedia entries and Google Maps business information.

By Luca de Alfaro, Ashutosh Kulshreshtha, Ian Pye, and B. Thomas Adler

**Should Privacy Impact Assessments Be Mandatory?**

Privacy impact assessments should be integrated into the overall approach to risk management with other strategic planning instruments.

By David Wright

**Cyberwarfare: Connecting the Dots in Cyber Intelligence**

Perpetrators and victims usually act as if it never happened, while the evidence is designed to erase itself.

By Sanjay Goel

Review Articles

88 **An Overview of Business Intelligence Technology**

BI technologies are essential to running today's businesses and this technology is going through sea changes.

By Surajit Chaudhuri, Umeshwar Dayal, and Vivek Narasayya

Research Highlights

100 **Technical Perspective**
Sketches Get Sketchier

By Peter J. Haas

101 **Theory and Applications of b-Bit Minwise Hashing**

By Ping Li and Arnd Christian König

110 **Technical Perspective**
SkinIntroducing the Future

By Scott Klemmer

111 **Skinput: Appropriating the Skin as an Interactive Canvas**

By Chris Harrison, Desney Tan, and Dan Morris



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 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

G. Scott Owens; Andrew Sears;

Douglas Terry

BOARD CHAIRS

Education Board

Andrew McGettrick

Practitioners Board

Stephen Bourne

REGIONAL COUNCIL CHAIRS

ACM Europe Council

Fabrizio Gagliardi

ACM India Council

Anand S. Deshpande, PJ Narayanan

ACM China Council

Jianguang Sun

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

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.acm.org

Letters to the Editor

letters@cacm.acm.org

WEB SITE

<http://cacm.acm.org>

AUTHOR GUIDELINES

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

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

Association for Computing Machinery (ACM)

2 Penn Plaza, Suite 701

New York, NY 10121-0701 USA

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

EDITORIAL BOARD

EDITOR-IN-CHIEF

Moshe Y. Vardi

vic@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

Robert Austin; Yannis Bakos; Elisa Bertino;

Gilles Brassard; Kim Bruce; Alan Bundy;

Peter Buneman; Andrew Chien;

Peter Druschel; Blake Ives; James Larus;

Igor Markov; Gail C. Murphy; Shree Nayar;

Bernhard Nebel; Lionel M. Ni;

Sriram Rajamani; Marie-Christine Rousset;

Avi Rubin; Krishan Sabnani;

Fred B. Schneider; Abigail Sellen;

Ron Shamir; Marc Snir; Larry Snyder;

Veda Storey; Manuela Veloso; Michael Vitale;

Wolfgang Wahlster; Hannes Werthner;

Andy Chi-Chih Yao

RESEARCH HIGHLIGHTS

Co-chairs

Stuart J. Russell and Gregory Morrisett

Board Members

Martin Abadi; Stuart K. Card; Jon Crowcroft;

Shafi Goldwasser; Monika Henzinger;

Maurice Herlihy; Dan Huttenlocher;

Norm Jouppi; Andrew B. Kahng;

Daphne Koller; Michael Reiter;

Mendel Rosenblum; Ronitt Rubinfeld;

David Salesin; Lawrence K. 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) 626-0686.

Single Copies

Single copies of *Communications of the*

ACM are available for purchase. Please

contact acmhlp@acm.org.

COMMUNICATIONS OF THE ACM

(ISSN 0001-0782) is published monthly

by ACM Media, 2 Penn Plaza, Suite 701,

New York, NY 10121-0701. Periodicals

postage paid at New York, NY 10001,

and other mailing offices.

POSTMASTER

Please send address changes to

DOI:10.1145/1978542.1978543

Sorel Reisman and Alain Chesnais

From the Presidents of the IEEE Computer Society and ACM

As most members of our organizations probably know, there is a great deal of overlap between the products and services offered by the IEEE CS and ACM. In fact, about 27% of ACM members belong to IEEE CS.

This became very clear to us when we first met at the SC2010 Conference in New Orleans, and had a brief conversation before stepping onstage to present the conference awards. Clearly, this wasn't a serendipitous meeting, as the SC Conference is sponsored annually by both organizations. What did surprise us was to learn that one of us (Alain), while born in Paris resides in Toronto, and the other (Sorel), resides in California but was born in Toronto. Moreover, we both have French names!

Two months later, over lunch in Toronto, we began to explore ways we might capitalize on the healthy competition our organizations have shared that has served the profession well over the decades. In a subsequent breakfast meeting three months later, we came up with the idea to craft a joint message to our members to appear concurrently in the August issue of *Communications* and *Computer*. Our goal is twofold: to describe areas in which we have cooperative and synergistic activities, and to explore new areas of mutual interest—all for the betterment of our profession.

One of the interesting challenges in writing this letter together has been how to present it without one or the other appearing to be “in control.” Rather than flipping a coin to determine whose name should come first, we decided we would alternate leads—with Alain as the lead author in the *Computer*-based version and Sorel taking the lead here in *Communications*.

As mentioned earlier, there are many areas in which the two organizations have overlapping interests, but the one for which both associations are internationally recognized is *computing education*. While our strategic strengths in this area may appear to have different foci—with ACM's being academic education geared toward schools and universities and the IEEE CS's efforts centered on professional education—in fact both organizations support life-long computing education programs. For example, both ACM and IEEE CS:

- ▶ Have representatives on the Computing Research Association (CRA), a premier resource for information about computer science education and technology-related public policy in the U.S.;
- ▶ Have representatives on the International Federation of Information Processing (IFIP);
- ▶ Share metadata in respective digital libraries;
- ▶ Promote Computer Science Education Week (Dec. 5–11, 2011);
- ▶ Developed and maintain the widely adopted curricula for Computing, Computer Engineering, Computer Science, Software Engineering, and Information Science;
- ▶ Sponsor the Graduate Software Engineering curriculum;
- ▶ Are working together to develop the Partnership for Advancing Computing Education (PACE), to be recognized as a cross-organization unified voice in support of computing education; and,

▶ Sponsor the aforementioned annual SC Conference (Seattle, WA, Nov. 12–18), the International Conference on Software Engineering (June 2–9, 2012, Zurich, Switzerland), along with many other conferences.

This letter to our joint membership is the first of what we hope will be many more cooperative activities aimed at capitalizing on the strengths of our organizations. We intend to continue to discuss new ideas about other joint efforts. More importantly, we would really like to hear from you, our members, with ideas and suggestions you might have that will help us along this path. To facilitate exchanges of ideas regarding this message, we set up electronic forums at <http://cooperation.computer.org/> and <http://cooperation.acm.org/>. Please visit those sites and participate in the discussions there.

In today's environment, we recognize that resources are constrained, and are likely to be so for a very long time. We also recognize the technologies we have invented and continue to invent are changing the world at a pace unlike any point in history. If ever there was a time for us to cooperate, it is now. □

Sorel Reisman (sreisman@computer.org) is president of IEEE CS and a professor at California State University, Fullerton.

Alain Chesnais (Chesnais@acm.org) is president of ACM and founder of Visual Transitions, Toronto, Canada.

© 2011 ACM 0001-0782/11/08 \$10.00

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



DOI:10.1145/1978542.1978544

A Policy that Deters Violation of Security Policy

IN THEIR ARTICLE “Does Deterrence Work in Reducing Information Security Policy Abuse by Employees?,” Qing Hu et al. (June 2011) analyzed deterrence of employee violation of information-security policy based on various criminological theories. Along the same lines, some years ago, when I interviewed more than 200 information security abusers,³ I found one of Donald R. Cressey’s criminological theories especially useful.¹ Cressey deduced from interviews of several hundred convicted embezzlers that mostly they were motivated by wanting to solve intense, non-shareable problems, exceeding the limits of their moral beliefs of right and wrong and self-control.

The survey Hu et al. described in their article, asking what a random sample of employees would do given several scenarios, is not particularly meaningful in the absence of the intense stress and highly variable conditions and circumstances I found to be present in cases of actual violation. In addition, perpetrators often find it easier to act against emotionless and faceless computers and prosperous organizations than directly against their fellow humans. Computers don’t cry or hit back, and, as perpetrators rationalize, organizations can easily help solve their problems and write off any loss.

Unfortunately, Hu et al.’s model did not include avoidance, separating or eliminating potential threats and assets, along with deterrence, leading only to the obvious advice of proactively hiring people with strong self-control and high moral standards. Organizations don’t knowingly hire people with such deficiencies; rather, employees become deficient under conditions and circumstances that emerge only during their employment. I concluded that providing employees in positions of trust free, easily accessible, confidential, problem-solving services is an important information-security safeguard,² sub-

sequently recommending it to many of my clients.

Donn B. Parker, Los Altos, CA

References

1. Cressey, D.R. *Other People’s Money*. Wadsworth Publishing Company, Inc., Belmont, CA, 1953.
2. Parker, D.B. *Fighting Computer Crime, A New Framework for Protecting Information*. Wiley, New York, 1998.
3. Parker, D.B. The dark side of computing: Computer crime. *IEEE Annals of the History of Computing* 29, 1 (Jan.–Mar. 2007), 3–15.

Authors’ Response:

We appreciate Parker’s critique of our approach to studying corporate computer abuses. Including known offenders in such a study would certainly be desirable. However, including the general population in any study of criminal behavior is a proven approach in criminology, as was our approach of using randomly selected office workers who may or may not have committed some kind of abuse. Both approaches are needed to better understand the complex social, economic, and psychological causes of employee abuse against their employers’ systems.

Qing Hu, Ames, IA,
Zhengchuan Xu, Shanghai,
Tamara Dinev, Boca Raton, FL,
Hong Ling, Shanghai

Agility Sometimes Another Extreme

I commend Phillip G. Armour’s Viewpoint “Practical Application of Theoretical Estimation” (June 2011), as I’m always on the lookout for ideas concerning software estimation, even as I ponder my own eternal mantra: “Estimates are always wrong.”

I agree with Armour but think he missed an opportunity in his section labeled “Practicing the Theory” to emphasize how agile methods avoid the extremes of compression and relaxation. Relaxation is avoided by breaking traditionally slow-to-deliver projects into small agile pieces, each easily delivered within the related market window. Working with these pieces also serves to avoid compression, since the same number of people can deliver the smaller agile pieces more quickly.

Armour also did say this is all theoretical and that even under the guise of agility companies regularly try to ramp up too many agile pieces too quickly.

Geoffrey A. Lowney, Issaquah WA

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

© 2011 ACM 0001-0782/11/08 \$10.00

Coming Next Month in COMMUNICATIONS

**Protecting Users
of the Cyber Commons**

**The Future of Wireless
Data Communications**

**Realizing the Value
of Social Media**

Code Bubbles

**Qualitative Analysis
of Real-time Systems Using
Priced-Timed Automata**

**Satisfiability
Modulo Theories**

**And the latest news about data breaches,
mobile app ecosystems, and solving systems of linear equations.**

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: 10.1145/1978542.1978568

Should Privacy Impact Assessments Be Mandatory?

David Wright

This article considers the issue of whether privacy impact assessments (PIAs) should be mandatory. The author examines the benefits and disadvantages of PIAs, the case for and against mandatory PIAs, and ultimately concludes they should be mandatory.

In 2000, an eight-year-old U.K. girl died from the abuse she had suffered at the hands of her great-aunt and the latter's boyfriend. There was a huge public outcry as details of the case became known. The

U.K. government eventually launched a public inquiry, the report of which found that the girl's murder could have been prevented had there been better communication between social services and other professionals.

The report led to the creation of a database, called ContactPoint, which the U.K. government said would improve child protection by improving the way information about children is shared among different social services.

While it might have been designed to solve one set of problems, the ContactPoint database created another set of problems: It has attracted significant criticism over the risks to privacy and personal data protection. The fact that some 330,000 people will have access to the database suggests that fears about the risks are

not entirely misplaced. There is a wide range of such risks—from identity theft to unauthorized, secondary use of personal data for research, for “sharing” with law enforcement agencies or other government agencies, for sale to insurance companies or companies engaged in personalized advertising.

ContactPoint is just one of many massive databases governments and industry have created and will continue to create that would benefit from a privacy impact assessment at the design stage and perhaps at later stages too as an iterative process.

contributed article

DOI: 10.1145/1978542.1978569

Cyberwarfare: Connecting the Dots in Cyber Intelligence

Sanjay Goel

Cyberwarfare is a potent weapon in political conflicts, espionage, and propaganda. Difficult to detect, it is often recognized only after significant damage has been done. Gaining offensive capability on the cyber battlefield figures prominently in the national strategies of many countries and is explicitly stated in the doctrines of several, including China, Russia, and the U.S. It is generally understood they are laying the groundwork for potential cyber conflicts by hacking the networks of adversaries and allies alike.

Cyberwarfare incidents are increasing among nation-states as well as terrorists, political/social organizations, and transnational groups. An early example of cyberwarfare was the 1999 targeting of U.S. government Web sites by suspected Chinese hackers in the aftermath of the accidental, as officially reported, U.S. bombing of the Chinese embassy in Belgrade. Cyberwarfare has since been observed largely as nuisance attacks (such as denial-of-service), with occasional incidents of espionage and infrastructure probes.

Future attacks could involve destruction of information and communications systems and infrastructure and psychological operations. Indeed, the cyberattacks against Estonia in 2007 and Georgia in 2008 hinted at the potential of cyberwarfare.

Long Island University • Brooklyn Campus

BLENDED LEARNING M.S. in Computer Science

Fusing online learning with traditional classroom studies, the Brooklyn Campus' new blended program significantly reduces the amount of time you'll spend on campus and maximizes interaction with faculty members and fellow students.

- Emphasis on design and development of large software systems.
- Core curriculum based on Association for Computing Machinery (ACM) recommendations.
- 36-credit program includes small implementation projects and/or programming exercises that provide practical training.
- Complete a large software development project or a thesis.
- Attend 25% - 49% classes online and enjoy face-to-face sessions with professors and classmates.

Blended Learning



For more information, call 718-488-1011
or visit www.liu.edu/brooklyn/mscs.

Call for Nominations

The ACM Doctoral Dissertation Competition

Rules of the Competition

ACM established the Doctoral Dissertation Award program to recognize and encourage superior research and writing by doctoral candidates in computer science and engineering. These awards are presented annually at the ACM Awards Banquet.

Submissions

Nominations are limited to one per university or college, from any country, unless more than 10 Ph.D.'s are granted in one year, in which case two may be nominated.

Eligibility

Each nominated dissertation must have been accepted (successfully defended) by the department between October 2010 and September 2011. Exceptional dissertations completed in September 2010, but too late for submission last year will be considered. Only English language versions will be accepted. Please send a copy of the thesis in PDF format to emily.eng@acm.org.

Sponsorship

Each nomination shall be forwarded by the thesis advisor and must include the endorsement of the department head. A one-page summary of the significance of the dissertation written by the advisor must accompany the transmittal.

Deadline

Submissions must be received by **October 31, 2011** to qualify for consideration.

Publication Rights

Each nomination must be accompanied by an assignment to ACM by the author of exclusive publication rights. (Copyright reverts to author if not selected for publication.)

Publication

Winning dissertations will be published by Springer.

Selection Procedure

Dissertations will be reviewed for technical depth and significance of the research contribution, potential impact on theory and practice, and quality of presentation. A committee of five individuals serving staggered five-year terms performs an initial screening to generate a short list, followed by an in-depth evaluation to determine the winning dissertation.

The selection committee will select the winning dissertation in early 2012.

Award

The Doctoral Dissertation Award is accompanied by a prize of \$20,000 and the Honorable Mention Award is accompanied by a prize of \$10,000. Financial sponsorship of the award is provided by Google.

For Submission Procedure

See <http://awards.acm.org/html/dda.cfm>



Association for
Computing Machinery

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:10.1145/1978542.1978546

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

Stonebraker on NoSQL and Enterprises

Michael Stonebraker discusses several reasons why NoSQL has not caught on with enterprise users.



Michael Stonebraker
"Why Enterprises Are Uninterested in NoSQL"

<http://cacm.acm.org/blogs/blog-cacm/99512>

Sept. 30, 2010

According to a recent ReadWriteWeb blog post by Audrey Watters, 44% of enterprise users questioned had never heard of NoSQL and an additional 17% had no interest. So why are 61% of enterprise users either ignorant about or uninterested in NoSQL? This post contains my two cents' worth on the topic.

At a recent trade show I attended that highlighted NoSQL engines, there were many Web developers, mostly from startups. However, I was struck by the absence of enterprise users. Hence, my (totally unscientific) experience confirms the basic point of the above blog post.

Moreover, in my experience, most information among enterprise users occurs by word of mouth. Hence, if they don't hear about something, it is because their professional network does not pass the word along. In other words, an interested enterprise professional generates additional interest. Non-

interest generates the behavior seen in the above blog post. So why is enterprise interest lacking?

To get more color on the situation, I contacted a very senior technical guru at a large enterprise who is responsible for looking at new database management system (DBMS) technology for his company. I asked him how interested he was in NoSQL and, in effect, how interested his company was. He reported "no interest." I asked him why.

He first said the vast majority of his company's applications are classifiable as online transaction processing (OLTP) where there are frequent small updates to a database of structured records or data warehouses/data marts that assemble historical business data for ad hoc query by analysts. Although there are other applications around the "edges," such as document management, these are not considered important.

He then made one comment about OLTP, one about warehouses, and one general comment. These follow.

No ACID Equals No Interest

Much of the OLTP data kept by this company is mission critical. Screwing it up causes people to lose their jobs. In

his world, ACID is the gold standard for updates to shared datasets. Any system that does not support real transactions is considered a nonstarter in his OLTP environment.

Even if a dataset can get by with single-record transactions now (a common feature of NoSQL DBMSs), he is unwilling to guarantee that it will never need multi-record transactions in the future. Put differently, his company assumes that ACID may be required in the future for any OLTP dataset, and nixes non-ACID systems.

A Low-Level Query Language is Death

Data warehouses are subject to frequent ad hoc queries like "Tell me whether pet rocks are selling better than Barbie dolls in the south?" Ted Codd's pioneering paper, "A Relational Model of Data for Large Shared Data Banks," in 1970 advocated a user interface whereby one stated what data he required instead of writing an algorithm to fetch relevant data from disk. In the subsequent 40 years of DBMS activity, high-level languages, like SQL, have been shown to offer ease of programming for such ad hoc data warehouse inquiries. My enterprise guru's company is rarely interested in the algorithmic record-at-a-time interfaces seen in most NoSQL products, as they are seen as a throwback to the days of IMS and CODASYL.

NoSQL Means No Standards

His company has a large number of databases (apparently more than 10,000), and the company is clearly concerned

with the number of different kinds of interfaces their application programmers have to learn. Hence, standards are important to a large enterprise.

Seemingly, there are north of 50 NoSQL engines, each with a different user interface. Most have a data model, which is unique to that system, along with a one-off, record-at-a-time user interface. My enterprise guru was very concerned with the proliferation of such one-offs. In contrast, SQL offers a standard environment.

I want to close this blog post with a single comment: “Those who do not understand the lessons from previous generation systems are doomed to repeat their mistakes.” In other words, “Stand on the shoulders of those who came before you, not on their toes.”

Disclosure: Michael Stonebraker is associated with four startups that are either producers or consumers of database technology. Hence, his opinions should be considered in this light.

Comments

This blog post makes me wonder why I pay \$100 a year to ACM.

Are you seriously going to sit there and disregard a very viable set of database options just because one person in one enterprise environment says he's uninterested? Or are you pushing your own agenda in the disguise of public opinion?

How do we teach the up-and-coming professionals that they should use the best tool for the job when presumably one of the top DB guys in the industry is waging a war on new technologies in the database field? I say presumably, because your continual dismissal of NoSQL solutions will render you irrelevant.

—Srdjan Pejic

Srdjan,

I am in no position to defend the author but it seems to me that what he is writing here is not NoSQL bashing. This article is a valuable thing; it is making clear to any NoSQL vendor what the barriers are that need to be overcome.

I work for an ISV that sells software to large enterprises and the issues raised here are the issues that would prevent us from using NoSQL. Our customers want to write their own reports using existing data warehouses; they want a RDBMS that fits into their exist-

ing support model.

“How do we teach the up-and-coming professionals that they should use the best tool for the job....” You do that by teaching them to use the best tool for the job; the point is that NoSQL is not going to be the best tool for the job as long as these barriers remain. “The job” is rarely just the application itself; data lives on forever, and enterprises want to use data everywhere and NoSQL vendors need to embrace that reality if they want to be enterprise players.

—Jamison M.

Srdjan,

At the top of the article it was made clear that it isn't “just one person”: “44% of enterprise users questioned had never heard of NoSQL and an additional 17% had no interest. So why are 61% of enterprise users ignorant about or uninterested in NoSQL?” Not to mention the fact that ACM has featured many articles enthusiastic about NoSQL, does that validate your \$100 a year?

In addition, it is quite clear that to an enterprise, NoSQL options are not “viable” for exactly the reasons stated.

I'd have to say, though, that the disclaimer at the bottom of this article is uncalled for, especially since similar disclaimers have not appeared on articles by proponents of NoSQL solutions (who are also financially invested in that tech).

—Jay Wright

Srdjan,

This is why Stonebraker is waging a counterargument to NoSQL: The average NoSQL fan lacks the ability to compare and understand relational database performance vs. NoSQL alternatives.

Nowhere has Mike ever stated, “For specific large dataset problems, SQL continues to outperform NoSQL.” Instead, I've seen him advocate for specific solutions to specific problems. CStore becomes Vertica, H-Store becomes Volt, and those who know better chose Postgres over MySQL.

In my personal growth, I came to understand that most of my startup's scalability problems had been solved before. Any time we started to get excited about Cassandra, BigTable, Dryad-LINQ, PNUTS, or K-V stores like Redis, Tokyo Cab, Couch, or Mongo, a more reasoned voice in our team was able

to educate everyone else that a typical relational SQL solution was still quite scalable while offering far superior consistency or isolation. We saw time and time again that NoSQL hype can easily trend toward uninformed religion.

There are very few people working on problems that really need to care about NoSQL or consistency-relaxed alternatives. Stonebraker's opinion is necessary to seriously question the NoSQL fanboy's understanding; he advocates different flavors of database solutions for different problems. That fact stands in stark contrast to your accusation that he ignores the best tool for the job, or is being rendered irrelevant.

—Jeff Vyduna

Jeff,

First of all, please do not assume I am a NoSQL “fanboy.” Also, how is it that you're sure I lack the ability to “compare and understand relational database performance vs. NoSQL alternatives,” as you put it?

A survey by *InformationWeek* is not a good representative of opinion. Most would say it's actually biased to favor established players like Microsoft and Oracle, so basing an article on those numbers is dubious at best.

Second, since you seem to have not read my comments carefully, I was complaining about the influence of this post by this author on “the best tool for the job” paradigm.

If your startup determines that basing your data store on a relational database is the best way to go, I will fully support you in that choice. Personally, I know that requirements my projects have fit better with a data model based on a K-V store like Mongo for stuff other than e-commerce. The e-commerce portion will go into something like Postgres, because the need for consistency is greater. Again, best tool for the job.

Jay, Why weren't any of these many enthusiastic articles referenced here as a counterpoint? Could it be Mike has an agenda against NoSQL solutions?

As for the stats reference, refer to what I wrote above about *InformationWeek*.

—Srdjan Pejic

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

© 2011 ACM 0001-0782/11/08 \$10.00



DOI:10.1145/1978542.1978547

Scott E. Delman

Why You Should Be Happy to Sign In!

For many of us, being confronted with those five dreaded words “This article is premium content” is nothing but a frustration. Many in the community would prefer all publications just be made freely available without barriers, constraints, or roadblocks that get in our way of downloading exactly what we want to read when we want to read it. And there are those who are not even aware that signing in is required to access all of the articles published in *Communications* or some of the features available on the site, such as commenting or establishing alerts and feeds. And there are still others who may appreciate why signing in is necessary and understand the need for ACM to charge for the services it provides, but are confronted with a more practical problem of remembering all those pesky little passwords. Personally, I fall into the last category, but I know the community at large is probably somewhat evenly distributed among the categories listed here.

What is important to remember is that ACM is not a large for-profit corporation focused primarily on generating large profits for shareholders, but rather is a mid-sized but leading not-for-profit international member organization with a singular focus on advancing the discipline of computing and related technologies for the purposes of education, research, and practice. Or in other words, ACM exists to support you and your work. As an organization, we are a membership-driven body that relies heavily on our more than 108,000 members from over 180 countries around the world. We are an organizer of international conferences, workshops, and symposia on hundreds of technology-related topics each year. As a publisher, we produce approximately 40 peer-reviewed scholarly journals and 10 niche magazines (including *Communications*) on topics that range from interactive design to entertainment computing to eLearning to students in computing. We deliver these materials to over 2,800 educational institutions and companies through an industry-leading online content platform called the ACM Digital Library (<http://www.acm.org/dl>), which is accessed by over 1,250,000 individuals annually.

When you sign in, it shows that you are part of a community that values the work of your colleagues and the services that ACM provides to this community. Providing high-quality publications to you in attractive and readable print and digital formats requires financial resources, and by signing in you are making a statement that you support this community and the work it does.

So, the next time you are confronted with the dreaded “premium content” sign-in page, please think back to this column and remember that the price of admission is not very high and that remembering those sign-in credentials is a small price to pay for everything you get.

the full text of this article is premium content!

sign in

Sign in using your ACM Web Account username and password to access premium content if you are an ACM member, Communications subscriber or Digital Library subscriber.

Username

Password

[Forgot Password?](#) [SIGN IN](#)

get access

Create a Web Account
If you are already an ACM member, Communications subscriber, or Digital Library subscriber, please set up a web account to access premium content on this site.

Join the ACM
Become a member to view this article and take full advantage of ACM's outstanding computing information resources, networking opportunities, and other benefits.

Subscribe to Communications of the ACM
Get full access to 50+ years of CACM content and receive the print version of the magazine monthly.

ACM Member News

GILLES BRASSARD WINS KILLAM PRIZE



The creation of a quantum computer capable of factoring large numbers at

unprecedented speeds would be disastrous for society unless steps are taken to change the way secrets are protected, says Gilles Brassard, winner of Canada's prestigious Killam Prize.

“It would really be a meltdown of electronic commerce,” says Brassard, a professor of computer science at the University of Montreal, and one of five scholars recently awarded a Killam Prize, which includes CN\$100,000, by the Canada Council for the Arts.

If a quantum computer, which uses concepts such as superposition to let a bit simultaneously perform multiple operations, were ever invented, it could quickly break any current encryption scheme that uses classical physics. Brassard says many people assume they can worry about such security concerns when a quantum computer is invented, but a malefactor could be collecting encrypted data now, waiting for the invention of a quantum computer. “If what you're sending today should remain secret for a long time, then you're not being secure,” Brassard says.

His current research involves designing classical cryptographic systems that can withstand quantum attacks. Quantum encryption is currently possible, but the required infrastructure is very expensive and its need is not obvious. Brassard and his team will present a secure classical cryptographic protocol in which it takes longer for an eavesdropper to break the encryption key than it does to generate it, at CRYPTO 2011. The protocol's time difference is small; if it takes N time to generate the key, it takes $N^{1/3/12}$ to break it. “It's not in any way practical or useful, but we're working on some ideas to make this gap longer,” says Brassard.

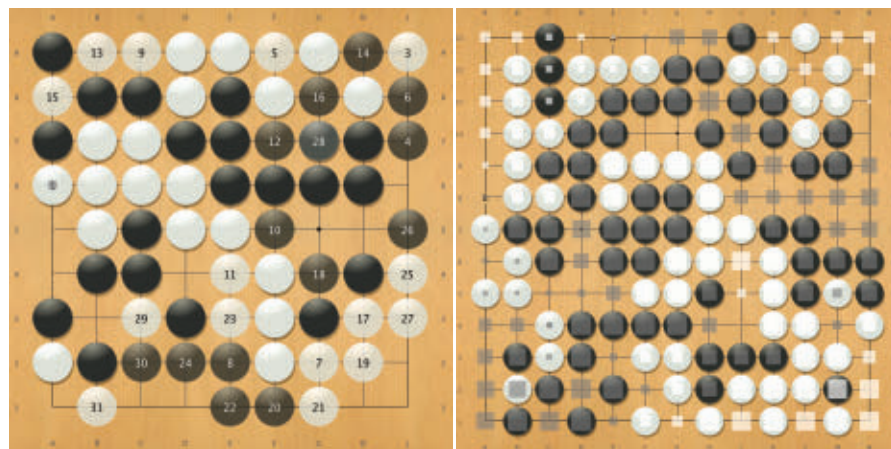
—Neil Savage

A New Benchmark for Artificial Intelligence

Computers are unable to defeat the world's best Go players, but that may change with the application of a new strategy that promises to revolutionize artificial intelligence.

IN 1997, WHEN IBM's Deep Blue beat world champion chess player Garry Kasparov in a five-game match, the media heralded the beginning of a new era in artificial intelligence. While the event undeniably marked a noteworthy milestone in the history of computers, and has served as an enduringly fresh metaphor for the possibilities of technology, what became clear in the years following the event is that many classical programming strategies for AI do not work well when applied to more complex applications. One such application that has emerged as a new benchmark for those conducting research in AI is the board game Go.

Go has proved to be extremely difficult for computers to master. To date, no computer has beaten a professional Go player on the 19×19 board in an even match. On the surface, Go might appear to be much simpler than chess, with players alternating placement of black and white pieces on a square board to capture more territory than their opponent by game's end. But the simplicity of Go's rules are deceiving. From a computer's perspective, Go is much more complex than chess.



Go boards showing a sample of how the Fuego software calculates board positions using Monte Carlo Tree Search. The size of the black or white square on each point indicates the percentage of how often that point belongs to each player at the end of a round of simulated moves.

Instead of dozens of branching movement combinations to evaluate as in chess, the branching search trees for Go may consist of hundreds of options for each move. For the first two Go moves alone, for example, more than 100,000 lines of play are possible, making options for each player's turn highly open-ended.

The approach that has proved in recent years to be the most likely path to victory for computers in Go is a meth-

od called Monte Carlo Tree Search, or MCTS. Only a few years ago, computers couldn't compete well even with amateur Go players, but the use of MCTS has resulted in software that can play near the level of the best professional players on the 9×9 Go board, and can even provide a decent game for seasoned players on the 19×19 board. The demonstrated strength of MCTS applied to Go has drawn attention from many areas of computer science, with



The Huygens supercomputer running MoGo software, which claimed the first nine-stone handicap victory against a professional Go player on a 19x19 board. During the official match, the supercomputer used 800 processors to power MoGo at a speed approximately 1,000 times faster than IBM's Deep Blue.

researchers now showing interest in applying MCTS techniques to other applications and domains.

Brute force methods, such as the classic alpha-beta search technique, are far too slow for Go analysis because of the game's open-ended nature. But with an MCTS algorithm, essentially a form of statistical sampling, it is possible to quickly consider all possible moves on the board, then simulate a set of random games. If a winning combination turns up in the majority of those simulated games, the MCTS algorithm considers that combination a good one. Otherwise, the algorithm

Currently, the most effective MCTS algorithms balance lengthy deep searches for winning Go moves against simulated explorations of new positions.

continues to search. Currently, the most effective MCTS algorithms balance lengthy deep searches for winning moves against simulated explorations of new positions.

One scientist doing work to improve the capabilities of MCTS is Martin Müller, a professor in the department of computer science at the University of Alberta. Müller is co-author of Fuego, a Go program that routinely places in the top ranks at the annual Computer Games Olympiad and was the first program to beat a professional Go player in an official game on the 9x9 board. Müller has worked on Go for more than 20 years, dating back to his Ph.D. dissertation at ETH Zürich, where he used combinatorial game theory to solve endgame problems.

Now using MCTS, he says the method promises to push the boundaries of what is possible in AI. The Fuego MCTS framework, for example, has already been used in many other applications. "As a long time computer Go researcher and Go player, I enjoy gaining new insights from the program's play as it becomes stronger than I am," he says. "This has happened on the 9x9 board, where Fuego has achieved some victories against top-level human competition." But the ultimate challenge, he says, is to scale the program so that it can beat humans on the 19x19 board.

MCTS's Superior Approach

Müller points out that while it has been somewhat of a mystery why MCTS has proved superior to earlier approaches, researchers are beginning to understand why MCTS methods can be more successful in hard domains. "For me, the main reason is that they do not require an evaluation function for arbitrary game positions," he says. "The simulations until the end of the game in MCTS make evaluating the best possible positions much easier."

Even so, the best Go programs are still weak in what Müller calls local situations. Massive parallelization is being used to sidestep this problem, but Müller says it is currently unrealistic to expect supercomputers to be able to resolve a large number of local situations simultaneously within a single global search. "Simply scaling to more and faster processors will not be enough with current techniques," he says. "I think we need one or two further breakthrough ideas in algorithms."

That breakthrough, Müller suggests, might come from MCTS being combined with other methods. For example, most strong algorithms are inherently sequential, meaning the results of all simulations must be derived before determining which simulations are run next. MCTS algorithms can sidestep this process with a technique called virtual loss, which operates on the assumption that simulations that have been started in parallel, but have not yet completed, will lead to a loss. While this technique is fast because it does not search the same simulations repeatedly, it is less effective than sequential search to resolve local situations, and it reaches a performance ceiling, even when parallelized.

What might lead to solving this problem, says Müller, is a method that can supplement global search with a set of effective local searches. "I believe we need local MCTS as well, and to integrate the results with global search," says Müller. "This is a huge technical challenge, but if successful, it would provide a way to scale up much better."

As for applying MCTS to other domains, Müller says the main challenge is to overcome the method's known limitations, particularly in cases where promising states are sparse, making them difficult to discover, or where

planning problems have extremely long solutions. In these cases, hybrid approaches, such as limited-length simulations followed by classical evaluations, might turn out to be the most effective strategies. “We will need to understand how classical techniques and MCTS can best be combined,” he says.

Another researcher working on applying MCTS to Go and other domains is Olivier Teytaud, a computer scientist at the University of Paris-Sud, and coauthor of MoGo, a Go program that has won several significant victories against human players and other Go programs. Teytaud’s early work focused on planning for industrial problems, but he shifted his attention to Go and MCTS several years ago. Like Müller, Teytaud is dedicated to improving the capabilities of MCTS in Go, but with an eye toward applying the ideas to other applications and domains.

“It’s clear that curing cancer, reducing pollution, or automating tedious tasks are more important than playing the game of Go,” he says. “But games are a very clear challenge, providing a great test bed for comparing algorithms.”

Teytaud says one of the reasons he became interested in MCTS is because of its ability to bypass the extrapolation problem. Unlike classical approaches that attempt to generalize domain knowledge to new scenarios, MCTS does not attempt to extrapolate. Instead, MCTS relies on search-tree simulations that require minimal domain knowledge, making it an attractive option for problems in AI. However, like Müller, Teytaud is focused on how best to get MCTS methods to handle local situations that would benefit from the kind of abstraction that humans can do so well.

“We have clearly understood the weaknesses of MCTS, and we are trying many different things to solve them,” Teytaud says. “But we don’t currently know which direction is the best to take.”

Progress in Other Fields

Despite the current mysteries surrounding MCTS, and no clear way to overcome the evident weaknesses of the approach, Teytaud says some progress has been made in applying the method to other applications in which extrapolation is difficult. “Power-plant

The main challenge of applying MCTS to other domains is to overcome the method’s known limitations, particularly in cases where promising states are sparse, making them difficult to discover, or where planning problems have extremely long solutions, says Martin Müller.

management is my favorite application because of its strong ecological and economic importance,” he says. “We would never have been given funding for working on power-plant management without the publicity provided by the work on Go.”

In his most recent effort along these lines, Teytaud is working directly with Artelys, a company that specializes in power-plant management, to develop a system that can respond intelligently to changes in power demand and outages. Managing power allocation, says Teytaud, is a problem similar to Go, where an opponent’s move may be likened to the failure of a plant, the computer’s response likened to switching other plants on or off, and the territory occupied at the end of a Go match likened to the ecological or economic benefit derived from proper power allocation.

Besides being useful for industrial applications, MCTS techniques developed for Go appear to be well suited to multiplayer games. An MCTS-based AI application that Teytaud developed for the massively multiplayer card game UrbanRivals was ranked in the top 1% in a round of matches consisting of some 9,000 players. “The suc-

cess of this application illustrates that the MCTS method can deal with partially hidden information,” he says. “There’s no hidden information in Go, so this application to card games is important.”

As for whether a computer will be able to achieve victory over a top-ranked Go player in an even match, Teytaud says he remains skeptical about it happening soon. “It will probably include some abstract thinking, much more than for chess,” he says. Like Teytaud, Müller says he is uncertain whether or when the technique will be used to beat a professional Go player on the 19×19 board. “I have no crystal ball,” he says, “but I hope to see at least one more fundamental breakthrough in the next decade.”

Even if a landmark victory—on the order of Deep Blue over Kasparov—is not on the horizon for MCTS and the game of Go, the popularity of MCTS likely will continue to grow in domains suited to heuristic search methods, such as industrial-optimization problems, multiplayer games, and other applications that stand to benefit from more effective AI strategies. **C**

Further Reading

Arneson, B., Hayward, B., and Henderson, P. Monte Carlo tree search in hex, *IEEE Transactions on Computational Intelligence and AI in Games* 2, 4, Dec. 2010.

Chaslot, G.M., Winands, M.H., and Herik, H.J. Parallel Monte-Carlo tree search, *Proceedings of the Sixth International Conference on Computers and Games*, Beijing, China, Sept. 29–Oct. 1, 2008.

Enzenberger, M., Müller, M., Arneson, B., and Segal, R. Fuego: An open-source framework for board games and Go engine based on Monte-Carlo tree search, *IEEE Transactions on Computational Intelligence and AI in Games* 2, 4, Dec. 2010.

Rimmel, A., Teytaud, O., Lee, C.S., Yen, S.J., Wang, M.H., and Tsai, S.R. Current frontiers in computer Go, *IEEE Transactions on Computational Intelligence and AI in Games* 2, 4, Dec. 2010.

Winands, H.M., Bjornsson, Y., and Saito, J. Monte Carlo tree search in lines of action, *IEEE Transactions on Computational Intelligence and AI in Games* 2, 4, Dec. 2010.

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

© 2011 ACM 0001-0782/11/08 \$10.00

Supercomputing's Exaflop Target

The twin challenges of parallelism and energy consumption are enlivening supercomputers' progress.

ASIA HAS COME out swinging at the top of June 2011's Top500 list, which rates the world's fastest computers based on the LINPACK benchmark. Leading the list is the K Computer, which achieved 8.2 quadrillion floating-point operations per second (petaflops) to give Japan its first appearance in the much-coveted number-one position since November 2004. It knocks China's Tianhe-1A, at 2.6 petaflops, to second place. The U.S.'s Jaguar (1.75 petaflops) was pushed from second to third place. China's Nebulae (1.27 petaflops) dropped from third to fourth, and Japan's Tsubame 2.0 (1.19 petaflops) slipped from fourth to the fifth position.

Asia's success comes at a time when national governments are reconsidering the value of these ratings—and what defines supercomputing success. The U.S. President's Council of Advisors on Science and Technology (PCAST) recently warned that a focus on such rankings could “crowd out ... fundamental research in computer science and engineering.” Even Jack Dongarra, the list's founder and a computer science professor at the University of Tennessee, believes the rankings need to be seen in a larger context. “You can't judge a car solely by how many RPMs its engine can do,” he says, pointing to the Graph500 list and HPC Challenge Benchmark as other sources of comparative supercomputing data.

Computer scientists are also skeptical about the value of such rankings. Petaflops are not the same as useful work; practical applications need to both preserve and take advantage of their power. Xuebin Chi, director of the Supercomputing Center at the Chinese Academy of Sciences, points out that “programming [for the CPU/GPU



The Top500's leading supercomputer, the K Computer, is more powerful than its five closest competitors combined.

combinations now popular in supercomputing] is more difficult than for commodity CPUs,” and that “changing from sequential to parallel code is not easy.” He predicts that such a transition would take three to five years. But even with superb programming, real-world aspects of data delivery and error correction could significantly reduce application speeds from those reported by the Top500.

Regardless, the November 2010 Top500 list's release, with China's Tianhe-1A in first place, spurred political discussions about national commitment to high performance computing (HPC) throughout the world. In the U.S., 12 senators cited Tianhe-1A in a letter to President Obama warning that “the race is on” to develop supercomputers capable of 1,000 petaflops

(1 exaflop). In asking for funding, they wrote that “Our global competitors in Asia and Europe are already at work on exascale computing technology ... we cannot afford to risk our leadership position in computational sciences.”

David Kahaner, founding director of the Asian Technology Information Program, believes that Tianhe-1A is the leading edge of a Chinese push to not only increase supercomputing speeds, but also domesticate production. “It represents a real commitment from the Chinese government to develop supercomputing and the infrastructure to support it,” he says. “A Chinese domestic HPC ecosystem is evolving. Domestic components are being developed and incorporated, and their use is likely to increase.” Dongarra agrees, noting, “The rate at which they're do-

“If you peek a little bit further into graphics problems, they look a lot like supercomputing problems,” says Sumit Gupta. “Modeling graphics is the same as modeling molecule movement in a chemical process.”

ing that is something we’ve not seen before with other countries.”

CPUs vs. CPU/GPU Hybrids

If the June Top500 list had been a foot race, the K Computer would have lapped the competition. At 8.2 petaflops, it wields more power than the next five supercomputers combined. The K Computer’s name alludes to the Japanese word “Kei” for 10 quadrillions, and represents the researchers’ desired performance goal of 10 petaflops.

Aside from national aspirations, the Top500 list reveals technical trends in HPC research worldwide. Most notable is an increased use of general-purpose graphics processing units (GPUs) in a hybrid configuration with CPUs. The June 2011 list includes 19 supercomputers that use GPU technology; the June 2010 list contained just 10.

GPUs contain many more cores than CPUs, allowing them to perform a larger number of calculations in parallel. While originally used for graphics tasks, such as rendering every pixel in an image, GPUs are increasingly applied to a wide variety of data-intensive calculations. “If you peek a little bit further into graphics problems, they look a lot like supercomputing problems,” says Sumit Gupta, manager of Tesla Products at NVIDIA. “Modeling graphics is the same as modeling molecule movement in a chemical process.”

In today’s supercomputers, GPUs provide the brute calculation power, but rely heavily on CPUs for other tasks. For example, the number-two Tianhe-1A contains two six-core Intel Xeon X5670 CPUs for each 448-core Tesla M2050 GPU (14,336 to 7,168); it also contains a much smaller number of eight-core Chinese-built Feiteng CPUs (2,048). Altogether, GPUs in Tianhe-1A contribute approximately three million cores—30 times as many as are in its CPUs.

But speed is not simply a matter of throwing more cores into the mix, as it is not easy to extract all of their processing power. First, data must be queued and managed to feed them—and to put the results together when they come out. “You need the CPU to drive the GPU,” Chi explains. “If your problem can’t be fit into the GPU itself, data will need to move frequently between the two, hurting performance.” Dongarra correlates this, saying, “The speed of moving data to the GPU and the speed of computing it once it’s there are so mismatched that the GPU must do many computations with it before you see benefits.”

The K Computer, though, bucks this trend. Unlike Tianhe-1A and other recent large supercomputers, it does not utilize GPUs or accelerators. The K Computer uses 68,554 SPARC64 VIIIfx CPUs, each with eight cores, for a total of 548,352 cores. And the Japanese engineers plan to boost the K Computer’s power by increasing the number of its circuit board-filled cabinets from 672 to 800 in the near future.

Asia’s Ascension

Asian researchers appear to be well positioned to exploit GPUs for massively

Top500 List, June 2011.

Rank	Computer	Nation
1	K Computer	Japan
2	Tianhe-1A	China
3	Jaguar	U.S.
4	Nebulae	China
5	TSUBAME 2.0	Japan
6	Cielo	U.S.
7	Pleiades	U.S.
8	Hopper	U.S.
9	Tera-100	France
10	Roadrunner	U.S.

Source: <http://www.top500.org>

Programming

ACM-ICPC World Finals

Students from Zhejiang University won the 2011 ACM International Collegiate Programming Contest World Finals in Orlando, FL, in May, beating student teams from 104 universities in the IBM-sponsored competition.

En route to its championship, Zhejiang University students used IBM’s large-scale analytics and cloud computing technologies to solve eight out of 11 programming problems in five hours.

The only other team to solve eight problems, the University of Michigan, won a gold medal for finishing second.

The University of Michigan team’s success was due to several factors, says coach Kevin Compton, associate professor in the school’s electrical engineering and computer science department. “First, they are very talented programmers and problem solvers,” he says. “A couple of them began entering programming competitions in high school.” In addition, from the time the regional contests were held in October 2010 and the World Finals, the Michigan team practiced intensely.

The Michigan team also created an efficient strategy for solving problems, with team members having clearly defined roles, Compton says. “Each member had a fairly specialized role, either as a coder or debugger and tester,” he says. “By following this strategy, they were able to avoid getting bogged down on a particular problem.”

Tsinghua University and St. Petersburg University finished in third and fourth place, respectively, winning gold medals. The other top 12 finalists, each of which received a silver medal, were Nizhny Novgorod State University (5th), Saratov State University (6th), Friedrich-Alexander-University Erlangen-Nuremberg (7th), Donetsk National University (8th), Jagiellonian University in Krakow (9th), Moscow State University (10th), Ural State University (11th), and University of Waterloo (12th).

—Bob Violino

parallel supercomputing. Kahaner believes China's relative isolation from Western influences may have led to economics that favor such innovations. "They're not so tightly connected with U.S. vendors who have their own perception of things," he says. "Potential bang for the buck is very strong in Asia, especially in places like China or India, which are very price-sensitive markets. If your applications work effectively on those kinds of accelerator technologies, they can be very cost effective."

According to Satoshi Matsuoka, director of the Computing Infrastructure Division at the Global Scientific Information and Computing Center of the Tokyo Institute of Technology, China's comparatively recent entry into HPC may help them in this regard. "Six years ago, they were nowhere, almost at zero," he says. "They've had less legacy to deal with." By contrast, Gupta says, programmers in more experienced countries have to undergo re-education. "Young programmers have been tainted into thinking sequentially," he notes. "Now that parallel programming is becoming popular, everybody is having to retrain themselves."

These issues will only get more complicated as time progresses. Horst Simon, deputy laboratory director of Lawrence Berkeley National Laboratory, says a high level of parallelism is necessary to progress past the 3GHz–4GHz physical limit on individual processors. "The typical one-petaflop system of today has maybe 100,000 to 200,000 cores," says Simon. "We can't get those cores to go faster, so we'd have to get a thousand times as many cores to get to an exaflop system. We're talking about 100 million to a billion cores. That will require some very significant conceptual changes in how we think about applications and programming."

Matters of Energy

Hybrid architectures have historically had another advantage besides their parallelism. They have also usually used less energy than comparable CPU-only systems. In the November 2010 list, hybrid systems generally delivered flops more efficiently than the CPU-only systems.

But the new Top500 list shows that the architectural battle over energy efficiency is still raging. The CPU-based

In today's hybrid CPU/GPU supercomputers, GPUs provide the brute calculation power, but rely heavily on CPUs for other tasks.

K Computer attains an impressive 825 megaflops (Mflops) per watt even as the third-place, CPU-based Jaguar ekes out a so-so 250 Mflops/watt. By comparison, the hybrid Tianhe-1A achieves 640 Mflops/watt, Nebulae gets about 490 Mflops/watt, and Tsubame 2.0 gets 850 Mflops/watt. (The list's average is 248 Mflops/watt.)

The most energy-efficient system is the U.S.'s CPU-based IBM BlueGene/Q Prototype supercomputer, which entered the Top500 in 109th place, with an efficiency of 1,680 Mflops/watt. The IBM BlueGene/Q tops the Green500, a list derived from the Top500 that ranks supercomputers based on energy efficiency. But despite BlueGene/Q's supremacy, eight of the Green500's top 10 are GPU-accelerated machines.

Energy is no small matter. The K Computer consumes enough energy to power nearly 10,000 homes, and costs \$10 million a year to operate. These costs would significantly increase in an exaflop world, notes Simon.

Looking Ahead

Despite the headlines and U.S. senators' statements, Dongarra and colleagues are quick to dismiss the supercomputing competition as a "race." At the same time, he expects to see an increase in Top500 scores, and notes that several projects are aiming for the 10-petaflop target, which could be realized by the end of 2012. But the real prize is the exaflop, which the U.S. government, among others, hopes to achieve by 2020.

Matsuoka believes this goal is possible, but it will be "a very difficult target,"

especially when compared with traditional expectations. "Look at Moore's law," he says. "Computers will get about 100 times faster in 10 years. But going from petascale to exascale in 10 years is a multiple of a thousand." Having said that, he notes that it is been done before—twice. "We went from gigaflops in 1990 to teraflops in about 10 years, and then to petaflops in another 10 years. Extrapolating from this, we could go to exascale in the next 10 years."

But Dongarra warns that we won't reach that stage solely by focusing on hardware. "We need to ensure that the ecosystem has some balance in it. Major changes in the hardware will require major changes in the algorithms and software," he says. "We're looking at machines in the next few years that could potentially have billions of operations at once. How do we exploit billion-way parallelism?"

The payoffs could be enormous. Supercomputing is already widely used in fields as diverse as weather modeling, financial predictions, animation, fluid dynamics, and data searches. Each of these fields embodies several applications. By way of example, Matsuoka says, "You can't do genomics without very large supercomputers. Because of genomics, we have new drugs, ways of diagnosing disease, and crime investigation techniques." While exaflop computers will spawn now-unimagined uses, any current increases in speed as we race toward that goal will greatly benefit many existing applications. ■

Further Reading

Anderson, M.

Better benchmarking for supercomputers, *IEEE Spectrum* 48, 1, Jan. 2011.

Endo, T., Nukada, A., Matsuoka, S., and Maruyama, N.

Linpack evaluation on a supercomputer with heterogeneous accelerators, 2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS), Atlanta, GA, April 19–23, 2010.

Miller, C.

Most popular supercomputing videos, July 13, 2010. <http://www.datacenterknowledge.com/most-popular-supercomputing-videos/>

Top500 list

<http://www.top500.org>

Tom Geller is an Oberlin, OH-based science, technology, and business writer.

© 2011 ACM 0001-0782/11/08 \$10.00

Life, Translated

The Holy Grail of language translation is to develop a machine-based system that can handle the task transparently and accurately.

THROUGHOUT HISTORY, COMMUNICATING between cultures has presented enormous challenges. Kings, prime ministers, presidents, and business executives have long traveled with translators in tow. Tourists have learned to lug along language tapes and phrase books so they can make their thoughts and needs known while visiting a faraway land.

Although numerous gadgets and gimmicks have come and gone over the last several decades, the Holy Grail of translation has always been to develop a machine-based system that can handle the task transparently and accurately. As Kevin Knight, senior research scientist at the Information Sciences Institute at the University of Southern California, notes, “Instantaneous and automated translation would have a profound effect on global communication.”

That future might not be so far away. Machine translation is rapidly moving into the mainstream of society. Web-based services such as Google Trans-

With various services now offering online translation for up to 50 language pairs, the focus is on developing general algorithms that work across commonly used groups of languages.



Smartphone applications like Google Translate rely upon machine learning to provide text-to-text and text-to-speech translation services.

late and Yahoo! Babel Fish make it easy to paste text into a Web browser and almost instantly transform it from one language to another. In addition, a new crop of smartphone apps offer text-to-text and text-to-speech features.

Meanwhile, IBM, Systran, and other companies are developing increasingly sophisticated systems for high-end government and business use. And university researchers are turning to machine translation to decipher ancient languages. “The field is advancing rapidly,” says Salim Roukos, senior manager for Multinational Language Technologies at IBM. “There is enormous demand for having machines translate text and speech.”

Breaking Down Barriers

The idea of using machines to translate languages stretches back to the 1940s. At that time, IBM scientists began exploring the idea of using linguistic and statistical decoding methods to automate language translation. However, the computers of that era were not

nearly powerful enough to accomplish the task. As a result, machine translation mostly languished until the 1980s.

Over the following decade, as more powerful processors and the Internet took hold, the international research community began building the foundation for machine translation. Early studies “gave researchers a clear vision that machine translation was within our grasp,” Knight explains. As a result, he says, “the race was on to improve the underlying algorithms that drive machine translation.” In fact, computer scientists recognized that effective machine translation was as much a mathematical and statistical challenge as a linguistic task.

Then, in 2007, Google introduced free online computer translation based entirely on a statistical model. Older forms of online translation continued to rely on linguistic modeling (some, like Babel Fish, continue to rely on both statistical and linguistics methods). Not surprisingly, the use of automated translation services soared, and

while Google Translate and others are not perfect, they are now widely used to translate Web pages, tweets, product manuals, and more.

Today, researchers are attacking the translation challenge head-on. Some, like Knight and Roukos, are looking for ways to build systems that translate text more accurately in real time. Others, such as Regina Barzilay, associate professor in the Computer Science and Artificial Intelligence Lab at the Massachusetts Institute of Technology, and Benjamin Snyder, assistant professor of computer science at the University of Wisconsin-Madison, are focusing on deciphering ancient languages for clues into the past, as well as insights into how to make machine translation more effective for modern languages.

Both groups face a similar challenge. "There is a lot more to machine language translation than simply building a word-for-word dictionary," Knight says. Not only is syntactic transformation different across different sets of languages, but neologisms, idioms, homonyms, and odd expressions make direct translation a daunting task. As a result, researchers focus on breaking language down into meaningful chunks and picking them apart with specialized programs, Knight says.

Improvements to machine translation systems often take place by trial and error. Developers must tweak

Today, the best machine translation systems boast accuracy rates above 90% when they are used with textbook speech, says Salim Roukos.

and modify algorithms—sometimes based on statistical and probabilistic models—to take into account new or previously overlooked phrases or combinations. The goal is for systems to recognize and compare words for context. This process often relies heavily on word alignment, Roukos says. "It's all about how words correspond with others in a sentence."

For instance, in English the word "fish" is somewhat ambiguous. It could either serve as a noun ("I eat fish") or a verb ("I fish at the stream"). "By contrast, if you look at the French translation for these two words, there are entirely separate meanings," says Snyder. "In French, 'poisson' is the noun version of fish and 'pêcher' is

the verb version. So, some triangulation has to take place for machine translation to work effectively."

With various services now offering online translation for upward of 50 language pairs, the focus is on developing general algorithms that work across commonly used groups of languages. The commercial market has evolved so that online users rely on free and less powerful software than governments and business users. IBM's Roukos says \$15 billion is spent on human translation annually, with heavy users including publishers, law firms, and organizations involved in international commerce.

Global travel sites, for example, frequently procure content in English but cater to an array of markets around the world. A company will typically train the system for specific phrases and terms, run the text through a machine translation system, and, in some cases, have a human review the content before posting it online. Tech firms that offer IT support are also turning to machine translation to provide support services in multiple languages. Others, primarily governments, run reams of printed content through translation systems before giving it to analysts for review.

Windows to the Past

Understanding ancient languages is another aim of researchers. However, in some instances, there are no exist-

Society

Digital Divide Persists, Berkeley Study Shows

Sociologists have known for years that income and education are good predictors of whether people use computers, and the Internet, to access online information. Indeed, more than 10 years ago, U.S. President Bill Clinton warned of a "digital divide" between the information haves and have-nots.

Now a new study from the University of California, Berkeley says an even greater class-based gap exists among the producers of online content, such as bloggers, Facebook and Twitter users, raters of movies, and chat room participants.

Moreover, says Jen Schradie, author of the study, "Given the persistence of these gaps, I believe they are unlikely simply to disappear over time."

A Ph.D. sociology student at Berkeley, Schradie analyzed data about 41,602 American adults surveyed between 2000 and 2008 by the Pew Internet and American Life Project. She examined 10 types of online content creation and found that college graduates were 1.5 times more likely to blog than high school graduates, twice as likely to post to a social network, and three times as likely to write ratings.

Schradie says the results surprised her because the class-based gap exists even among those people who are already online. Also, she says, it cannot be explained by age, gender, or race.

The digital production gap is worrisome, Schradie says, "because people in power rely on the Internet now to search for what matters—whether for politicians who listen to social media campaigns, journalists who rely on Twitter or Facebook posts, or the general public who go to Google for information—but without the voices of the poor and working class, is the Internet

really an accurate picture of America?"

Schradie's recommendations? "First, rather than defunding public libraries, local governments should be expanding [their] Internet services," she says. "Next, I believe that people who work in IT need to stop and think about how a full 25% of American adults have never been online, let alone contributing to the Internet cloud. What this means is that relying on the Internet for either a pulse of opinion or data is inaccurate at best and unjust at worst."

—Gary Anthes

ing translations or a clear understanding of how the language is constructed. In essence, researchers looking for clues into ancient civilizations find themselves working blind. “It is very difficult to decipher an unknown language,” MIT’s Barzilay explains.

This hasn’t stopped her and others from accepting the challenge. In 2010, Barzilay worked with a team of researchers to untangle the ancient Semitic language of Ugaritic. The team built several assumptions into the translation software. First, they decided that Ugaritic would be similar to another language (Hebrew, in this case). Second, they decided that it is possible to map the alphabet of one language to another and find symbols or groups of symbols that correlate between the two languages. Finally, they examined words looking for shared roots.

It was a difficult task, with the computer parsing through the data hundreds and sometimes thousands of times, using probabilistic mapping techniques associated with artificial intelligence. Along the way, the system began to identify commonalities and inconstancies, including shared cognates (similar to “homme” and “hombre” in French and Spanish, respectively). Researchers continued to analyze language combinations until they saw no further improvements.

The result? The software mapped 29 of the 30 Ugaritic alphabet letters to Hebrew. Approximately one-third of Ugaritic words have Hebrew cog-

mates. The researchers correctly identified 60% of these words. Many of the remaining words were off by only a letter or two. A human could easily correct these inconstancies, as well as spot homonyms. In this case, the team could verify the results because Ugaritic has already been deciphered (though they worked blind in their research).

Investigation into ancient languages is not a trivial pursuit. “The process has a great deal of applicability with current languages and translating them correctly,” Barzilay says. Simply put, when researchers learn the DNA of languages they are able to build better models for translation. “We learn how to make linguistic assumptions and build a better refinement cycle,” she points out.

The Final Word

The field of machine translation continues to advance. IBM’s Roukos addresses the linguistic challenges of machine translation using a multidisciplinary team with mathematicians, electrical engineers, computer scientists, linguists, computational linguists, and programmers. Altogether, the group speaks 20 different languages. “We run hundreds of billions of words through computers and analyze statistical models,” he says.

Today, the best machine translation systems boast accuracy rates above 90% when they are used with textbook speech, according to Roukos. Error rates typically double with colloquialisms and informal speech. However, researchers are building larger databases and working to perfect multilingual natural language processing, Roukos says. “Ultimately, the goal is to build systems that can mine text, extract information, and understand how language is used for different situations and applications.”

There is also a push to add languages that have not yet been cataloged. “In Africa there are thousands of languages,” says Knight, “and almost none of them have been touched by machine translation.” Meanwhile, researchers such as Barzilay and Snyder focus on unlocking the mysteries of hundreds of remaining “lost” languages. Tackling the task manually is next to impossible. “It’s a job that only comput-

ers can do,” Barzilay says.

Ultimately, machine translation promises to revolutionize work and life. Google and others are developing image-to-text and image-to-speech translation systems that allow a person to snap a photo of a sign or text and receive a translation. Google Translate on the iPhone already provides speech-to-speech translation across more than two dozen languages.

Meanwhile, the U.S. National Institute of Standards and Technology is testing smartphone-type devices that translate speech instantly between two handsets. The system, dubbed TRANSTEC, has already been used to translate between English and the Afghani tongue Pashto.

More powerful computers and better algorithms promise to further revolutionize the field. “Within the next decade or two we will see remarkable progress in machine translation,” Knight says. “It will likely become a regular part of our lives.” 

Further Reading

Callison-Birch, C., Koehn, P., Monz, C., and Schroeder, J. Findings of the 2009 workshop on statistical machine translation, *Proceedings of the Fourth Workshop on Statistical Machine Translation*, Athens, Greece, March 30–31, 2009.

Chiang, D., Knight, K., and Wang, W. 11,001 new features for statistical machine translation, *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics on NAACL 09*, Boulder, CO, May 31–June 5, 2009.

Ge, N. A direct syntax-driven reordering model for phrase-based machine translation, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, CA, June 2–4, 2010.

Koehn, P. *Statistical Machine Translation*. Cambridge University Press, Cambridge, England, 2010.

Willis, Y. *Machine Translation: Its Scopes and Limits*. Springer, New York, NY, 2010.

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

© 2011 ACM 0001-0782/11/08 \$10.00

Google and others are developing image-to-text and image-to-speech translation systems that allow a person to snap a photo of a sign or text and receive a translation.

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 dark ink that reads "Alain Chesnais". The signature is written in a cursive, flowing style.

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

Economic and Business Dimensions Geography Is Alive and Well in Virtual Teams

Just because a team is virtual, it doesn't mean geography is dead.

THOUGH MEMBERS OF a virtual team can work anywhere, virtual does not end the role of geography. Geography is alive and well. Geographic distance and lack of overlapping work hours impose large coordination burdens on a team. Research has begun to suggest what circumstances generate the largest burdens and why.

For example, members physically separated by equally great distances experience fewer coordination challenges when they are within the same or a few time zones than different time zones. Thus, not surprisingly, members from New York find it easier to work with members from Buenos Aires than Tokyo. However, for team members within the same or a few time zones, physical separation still reduces awareness of what other members are doing. Thus, in order to benefit from working virtually, members need to allocate enough time to the task to overcome these additional coordination obstacles.

Geographical Space and Time

Geographic dispersion in teams is a multidimensional construct. Michael O'Leary and I noted that while people often think of teams as being dispersed or not, dispersion is much better characterized by the *degree* to which members are in different geographic locations.³ Spatial dispersion should be treated as a *separate* dimension than temporal dispersion. Together, geographical space and time yield important insights about working at a distance, including the benefits (for example, accessing member expertise) and costs (for example, constrained member communication); see the accompanying figure.

In the figure, Quadrant A reflects teams with high spatial dispersion but low temporal dispersion. For example, members could be split entirely between North and South America, or between Europe and Africa. Quadrant B shows teams with high spatial and temporal distances. These far-flung teams could have members distribut-

ed across North America, South America, Europe, and Africa. Quadrant C reflects teams with members who are co-located and work at the same time while Quadrant D shows teams with members in the same location but split entirely between two work shifts (for example, 5 A.M.–2 P.M. shift vs. 2 P.M.–11 P.M. shift).

Time zones and coordination delay.

How should this more nuanced view of geography in teams affect thinking about where members work spatially and temporally? For teams in Quadrant A (distributed north-south) versus Quadrant B (distributed east-west), consider the consequence of coordination delay, which occurs when it takes longer than expected for one member to receive progress reports, communication clarification, or issue resolution from another member. Over time, coordination delay among members can lead to worse team performance in terms of completing the work on schedule, completing work within budget, and meeting final requirements.



In a study of 108 teams in a multinational corporation (projects included software development, hardware development, and systems integration), this is what Alberto Espinosa, Cynthia Pickering, and I found.¹

Using company database information on the locations of 675 team members (in 54 cities across 22 countries),

we determined the extent to which each pair of team members was separated by spatial boundaries (same city vs. different city/same country vs. different country). To measure temporal boundaries (non-overlapping work hours), we used time zone data associated with each city along with work hours in a 24-hour time period (and ac-

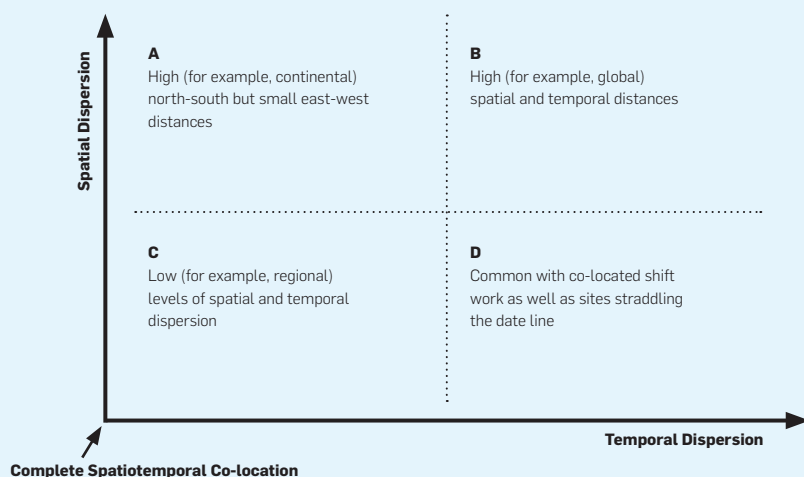
counted for members who shifted their hours to overlap with other members). Some pairs of members had at least one hour of overlap during a nine-hour workday and some pairs of members had no hours of overlap during a nine-hour workday.

Based on surveys, we also assessed the extent to which each member used synchronous communication tools and asynchronous communication tools with every other member of their team. Lastly, we asked about coordination delay with other team members.

The results of our study showed that, as expected, working across spatial and temporal boundaries were both significantly related to coordination.¹ That is, coordination delay was greater for pairs of members in different countries compared to pairs of members in different cities (in the same country) compared with pairs of members in the same city. And coordination delay was greater for pairs of members who had no overlap in their workday compared to pairs of members who had at least one hour of overlap.

We also found that while synchronous tools (such as Web conferencing) were effective in reducing coordination delay when pairs of members

Combinations of spatial and temporal dispersion in teams (source: O'Leary and Cummings²).



Note: The quadrant boundaries are shown approximately at the midpoint, although this is only for ease of presentation, not to indicate any specific significance of the midpoint.

were separated by space but not by time, asynchronous tools (such as email) were *also* effective when pairs of members had high spatial boundaries but low temporal boundaries. Thus, temporal boundaries were more difficult to cross than spatial boundaries given that none of these communication tools reduced coordination delay for pairs of members in different countries who had no overlap in their workday.

Further interviews led us to conclude that teams with better alignment of geography to the work being done (that is, maximizing access to member expertise while minimizing communication constraints) were more effective compared with teams that tried to work through high spatial and temporal boundaries. One virtual team member even reported reorganizing teamwork to take in account being separated by many time zones: *“One of the problems that we have when working with people that are eight hours away in time zones, is the coordination of large projects...the engineers need to work together to talk through problems. So, when there are significant time differences, they just can’t make good solid progress without being able to talk. So, what we started to do in the last couple of years is starting to divide up the project to where each site has a completely separate job, with some minor overlap with the other sites. But in general, they have one complete section of the project all to their own. Otherwise, what we found in trying to manage one functionality across multiple sites is that people were having to work a ridiculous number of hours in order to maintain that communication.”* (interview cited in Cummings et al.¹)

Physical separation and time allocation. Now consider teams in Quadrant A (distributed north-south) versus Quadrant C (co-located), and how members allocate their time to the team. It is common today for members of teams to work on several teams concurrently. As a result, virtual teams are often composed of members who vary widely in the amount of time they work on the team.

To investigate how geographic dispersion impacted the relationship between time allocation and team performance, Martine Haas and I studied

There was an advantage for more dispersed teams when members allocated more time to the team.

285 teams in a large global company (projects focused on product innovation, operational improvement, and customer service). The 2,055 members were located in over 50 countries. We measured the extent to which pairs of members were on the same hallway, different hallway but same floor, different floor but same building, different building but same city, and different city but same country, or different country. We then aggregated this fine-grained location data to the team level to capture geographic dispersion among members.

We also measured the percentage of time that each member contributed to the team, and used independent ratings by company executives to measure performance (for example, value delivery, goal alignment, and tangible results). As expected, we found that teams with members who allocated a greater percentage of their time to the team performed better, on average, than teams with members who allocated less of their time to the team.²

More interesting for the discussion of geography, we showed that there was an advantage for more dispersed teams when members allocated *more* time to the team. That is, teams with members spread across different buildings, cities, and countries benefited more from allocating more time to the team relative to teams with members spread across different hallways and floors in the same building.

Greater physical separation (for example, different floor vs. different building vs. different city) took a greater toll on member attention in terms of keeping track of what others were doing in different locations and managing the communication required to

stay on task. This raises an important question: If virtual teams are more likely to have members who allocate less time, yet virtual teams benefit the most when members allocate more time, how should managers resolve this paradox?

We suggest that when designing teams, managers should explicitly take into account where members work when determining how much time members should allocate (to maximize attention focus when dispersion is greater). For distributed work to be successful, the task should not be added onto what members are already doing, since just because the work is distributed does not mean it is going to be better.

Conclusion

The realities highlighted here regarding the persistence of geography in virtual teams are not intended to offset the enthusiasm around advanced collaboration technologies (for example, Cisco’s Quad, IBM’s Lotus Connections, Salesforce.com’s Chatter, Microsoft’s Lync). Rather, they are intended to clarify where certain trade-offs still exist when working across space and time.

Moving forward, technologies with the best odds of success in assisting virtual teams will need to increase member communication as well as help manage and coordinate their work through better partitioning tasks by location, managing dependencies among tasks that bridge locations, and synchronizing how tasks are integrated across locations. Otherwise, there is a greater chance of an obituary being written for particular technologies than for geography. **C**

References

1. Cummings, J.N., Espinosa, J.A., and Pickering, C.K. Crossing spatial and temporal boundaries in globally distributed projects: A relational model of coordination delay. *Information Systems Research* 30, 3 (Mar. 2009), 420–439.
2. Cummings, J.N. and Haas, M.R. So many teams, so little time: Time allocation matters in geographically dispersed teams. *Journal of Organizational Behavior*. (In press).
3. O’Leary, M., and Cummings, J.N. The spatial, temporal, and configurational characteristics of geographic dispersion in work teams. *MIS Quarterly* 31, 3 (Mar. 2007), 433–452.

Jonathon N. Cummings (jonathon.cummings@duke.edu) is an associate professor of Management at the Fuqua School of Business, Duke University, where he is also faculty director of the Center for Technology, Entertainment, and Media. His research was supported by NSF Award #0603667.

Copyright held by author.

Education

From Interests to Values

Computer science is not that difficult but wanting to learn it is.

COMPUTER SCIENCE IS not that hard. Sure it takes work and perseverance, just like learning a second language or studying political science or a hundred other things. But the media depicts computer science as a gift that some people are born with. Movies would lead us to believe that a magical computer gene enabled Mark Zuckerberg to start coding as a teen with the only major side effects an undying thirst for Mountain Dew and social awkwardness. But computing isn't magic fairy dust in our genes. People become computer scientists because they put in the work and have good strategies for how to learn.

We can teach the strategies about how to learn computer science, but how do we get the students to sign up for a computer science class when they are afraid they don't have some innate talent or that CS is just too hard to bother with? For many groups, this may be part of what is keeping them from participating in equal numbers in CS. The National Science Foundation (NSF) has tried to reach these groups, to broaden participation in computing by increasing the number of women, people with disabilities, and Latino, Native Americans, and African Americans. We should continue these efforts, because without these groups participating fully in CS we are recruiting from only 35% of the U.S. population. We will not be able to fill jobs or compete in the marketplace without approaches that motivate young people from a broad set of demographics to persevere in CS. But more importantly the underlying design of computational devices will be based



Glitch game testers evaluate a pre-release version of NCAA Football by Electronic Arts.

upon the values of a few rather than multiple perspectives that will increase innovation and appropriateness of our designs for the audiences they serve.

What does it take to get these groups interested in and then motivated enough to put in the hard work to succeed in computer science? Beyond what they are interested in, what do they *value*? By this we mean in particular, how do they see their current and future role in the world, and how does that shape what they find compelling and worthwhile? Many interventions aimed to broaden participation in computing begin with what underrepresented youth find “cool” or “fun.” However, most teenagers—especially those from economically less advantaged backgrounds—have issues more pressing than fun on their minds. They

worry about their economic futures in particular. Helping them to imagine new possible futures for themselves is a daunting task. We need to stop looking at what motivates computer scientists and start looking at what motivates our target audience. This may mean segmenting the audience and listening to their needs as groups based on the intersection of gender, race, geography, and economic status. It definitely means that one size does not fit all. A program we have developed, the Glitch Game Testers, is showing promise in this approach by focusing on working with young African-American males to design a CS education program.

The Glitch Project

Glitch is a project of Amy Bruckman's Electronic Learning Communities

Lab and the dissertation work of Betsy DiSalvo in the School of Interactive Computing at Georgia Tech. Glitch is a game testing work program that started in 2009. It is made up of African-American male high school students, with a total of 25 participants over the last two years and little attrition except for those who have graduated. Glitch operates out of a lab in the College of Computing Building at Georgia Tech in partnership with Morehouse College.

Testers are paid to work full-time in the summer and part-time during the school year testing pre-release games for companies like Yahoo!, Cartoon Network, Last Legion, and Kaneva. The program was developed with a demonstration grant from the NSF's Broadening Participation in Computing program under the direction of Jan Cuny. But corporate support has been available from the beginning with Electronic Arts Tiburon helping shape the program and providing Quality Assurance training to the staff and participants, and more recently with support from Yahoo! and Microsoft.

These prestigious companies are working with Glitch, but the testers come from a different world. The majority of testers qualified for free or reduced lunches, attend poorly ranked schools, and are generally not at the top of their class. They were recruited by word of mouth, which resulted in many more applicants than the program could handle. So they were primarily accepted on a first-come, first-served basis. Even though they are coming from rough neighborhoods, these high school students are interacting with developers from game companies they admire. They work just like professional game testers with deadlines and expectations placed on them. The testers are paid hourly, and spend one hour a day on CS class work taught by African-American male undergraduate students from Morehouse and Georgia Tech.

In the first year of class, the testers worked in a computational media approach,⁴ integrating the drag-and-drop programming environment Alice with Python programming. The testers who were entering their second year in the program wanted to do something more challenging, something more

like "real programmers." They were offered an Advanced Placement Computer Science (APCS) class that teaches deeper CS concepts with Java—and all students elected to take it.

Finally, both first and second year students build computers. This serves two purposes beyond learning about computer hardware. First, they needed higher-quality machines to test on—the program was started using surplus machines from Georgia Tech and Morehouse and building them was cheaper than buying them. Second, one of the computers is the final prize in the Glitch "point competitions." Each school year and each summer the participant who earns the most points from bug reports, regression testing, CS assignments, and other tasks earns a computer. This competitive environment plays an important part in motivating students, not only to work hard to earn points but also as a "face saving" excuse when challenged about why they are putting in extra effort. In teen culture, it is sometimes perceived as "not cool" to try hard. But trying to win a computer is a socially acceptable reason to work hard—and our students do.

A number of measures including computer science tests and self-assessments indicate Glitch participants have increased their knowledge about software development and programming. Pre- and post-surveys indicate that Glitch has changed attitudes about studying computing and intent to persist in computing. For example, 22 out of 23 testers last summer reported an increase in their interest in technology. Perhaps the most compelling indication of success is what the students are doing when they leave. Out of the seven students who graduated from high

Being a computer programmer was no longer a mystical quality that was out of their reach...

school in 2010, six are attending college—five of them in computing related majors, (two in CS, one in computer engineering, two in digital media).³ Of the seven students who will be graduating in 2011, all intend to attend college—four have declared their major as CS and the other three are considering CS as their major. Before the program started, of the 14 only one had an interest in CS and one in computer engineering as a major. These are amazing numbers for any type of computer science outreach, but particularly with a group that is usually so difficult to reach.

Why did it work, and how can others do similar things? The Glitch program motivated the students to put in the work to do computing. Being a computer programmer was no longer a mystical quality that was out of their reach but a natural progression from what they were already doing as game testers. They communicated with developers daily, and felt that developers really listened to them, changing games based upon what they had written in bug reports. They already felt like a valuable part of the software development process, so imagining themselves as a bigger part of that process was easy.

Challenges

There are still significant hurdles. From our staff's assessments, four of the seven students who participated in the Glitch APCS classes were prepared to pass the test in May of this year, but only one chose to take the test. We are trying to understand why these young men who had worked so hard all year learning introductory CS would not be interested in sitting for the test. They claim that logistics was part of the reason; the test was offered at 8:00 A.M. at a different school than they attended, on a school day, and none of them wanted to wake up that early to take public transportation there and miss school. But we suspect another factor was intimidation. Perhaps they feared taking the test at school that might be perceived as "better," or not doing well and losing the face of being tech savvy to their family and perhaps themselves. We have started to ask them questions and would like to gain a better understanding of why they did not

take the test. Like all teens, they are not always consciously aware of their own motivations. The challenge is not just to encourage them to share their reasons, but to help them develop greater self-awareness and the ability to articulate their conscious and sub-conscious motivations.

And these questions may have larger implications for African-American teens across Georgia. The Georgia Computes Broadening Participation in Computing Alliance¹ has made increases in the number of APCS test takers (from 389 when the program started in 2004 to 692 in 2010). However, the percentage African-American test takers has remained flat. In 2010, only 9.8% of the test takers were African American compared with 29% of the Georgia population. Even more shocking is how few African-American students pass the APCS exam. Only 16 of 68 (23.5%) African-American students passed in 2010, compared to an overall pass rate of 56%.² Connecting the value of taking the APCS test to the values of our African-American students will be important for our future efforts.

Games are an obvious interest match for young African-American men—they play often and games are reported to be the digital media they interact with most frequently. Creating a competitive environment went along well with their values on competitive gaming practices and the quota-based game testing industry. Finally, these young men needed jobs. They could not justify traveling to Georgia Tech every day, or even every week to just learn about computers. The pragmatic approach they take to their extracurricular activities needs to be reflected in the values of the program.

The challenge is to identify interest and values in other groups and design educational programs around them. While a number of programs use games, robots, or other “fun” computing tools to get kids interested, maybe we need to step back and stop trying to make computing look cool and start looking at audiences’ values. While these young African-American men were attracted to game testing because it was cool, they were able to participate because it was a paying job that met needs they valued more than an interesting thing to do with their time.

They want to continue in computer science because they now see CS as a pragmatic approach of an attainable high-paying career, something they and their families value.

Conclusion

We began to understand the values of young African-American men from underserved neighborhoods through in-depth formative work consisting of observations, participatory design activities, interviews, focus groups and prototyping. And after two years of running the Glitch program, we can begin to speak with authority of what these values are and how they impact learning CS. Many interventions designed to make computing engaging for students focus on their interests. It definitely helps to take elements from popular culture and the arts that appeal to your target audience and leverage those interests. But fewer interventions leverage students’ *values*. What is “fun” and what is “important” for kids are two different things. For teens that grow up in poverty, finding a path to economic security is critical. And it has to be an accessible path—one they can easily imagine themselves on. Values connect to what they view as important, who they see themselves as now, and who they can imagine becoming. The observation that African-American teenage boys often like video games was a first step for our design of Glitch, but perhaps the least important one. Connecting to the deep concept of values is much more important than the surface notion of interests. 

References

1. Bruckman, A. et al. *Georgia Computes!: Improving the Computing Education Pipeline*. Paper presented at the SIGCSE 40th ACM Technical Symposium on Computer Science Education Chattanooga, TN, 2009.
2. CollegeBoard. *Advanced Placement Testing National Summary Reports: 2010*, (2010).
3. DiSalvo, B. et al. *African American Males Constructing Computing Identity*. Paper presented at the ACM CHI Conference on Human Factors in Computing Systems, Vancouver, BC, 2009.
4. Guzdial, M. A media computation course for non-majors. *SIGCSE Bulletin* 35, 3 (Mar. 2003), 104–108.

Betsy DiSalvo (bdisalvo@cc.gatech.edu) is a Ph.D. student in the School of Interactive Computing at Georgia Institute of Technology.

Amy Bruckman (asb@cc.gatech.edu) is an associate professor in the School of Interactive Computing at Georgia Institute of Technology.

Copyright held by author.

Call for Nominations for ACM General Election

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

Suggestions for candidates are solicited. Names should be sent by **November 5, 2011** to the Nominating Committee Chair, c/o Pat Ryan, Chief Operating Officer, ACM, 2 Penn Plaza, Suite 701, New York, NY 10121-0701, USA.

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

Dame Wendy Hall is the Chair of the Nominating Committee, and the members are Anand Deshpande, Ben Fried, Cherri Pancake, and Robert A. Walker.

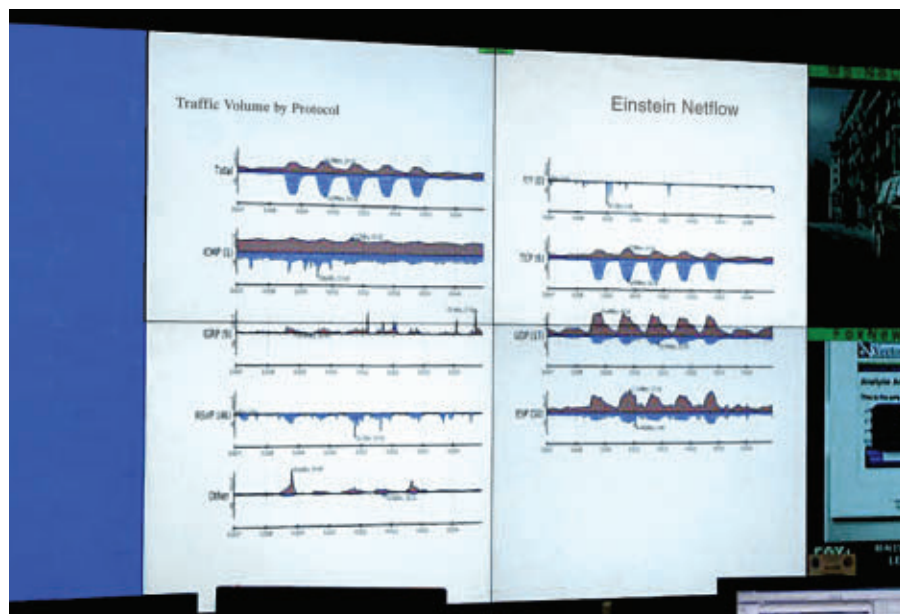


Privacy and Security As Simple As Possible— But Not More So

An assessment of the U.S. government's EINSTEIN project.

THE PROBLEM: SECURE the cybernetwork of a large enterprise. It could be one supplying a vast array of services including disease tracking, astrophysics research, weather predictions, and veterans' health care services to a population of 300 million. Or it could be even bigger, perhaps providing critical infrastructure to the population. Requirements include doing so cheaply and efficiently. The solution? The one the U.S. government has begun to implement—EINSTEIN—involves centralizing its Internet connections so as to perform intrusion detection and prevention on all incoming communications.

Because providing services to the public is a fundamental role for U.S. federal civilian agencies, many agencies turned to the Internet in the 1990s. While confidentiality, integrity, and authenticity dominated early federal thinking about Internet security, agencies faced phishing, IP spoofing, botnets, denial-of-service attacks (DoS), and man-in-the-middle attacks. By the early 2000s, the growing number of attacks on U.S. civilian agency systems could not be ignored. The U.S. government's solution has been to build centralized intrusion detection systems (IDS) and intrusion prevention systems (IPS) at large scale. The project, called EINSTEIN, works at an agency-wide, and in some cases, multi-agency-wide level. Federal civilian systems have two million direct users and serve many more.



EINSTEIN screen depicting volume across U.S. federal government networks by protocol.

Few doubt the value of IDS and IPS as part of a cybersecurity solution, but can a centralized system such as EINSTEIN really work? What attacks will EINSTEIN prevent? What will it miss? What are the privacy implications of using the interception program? These are the questions to which we sought answers. Our answers are provisional because few technical details of the system are public, but these answers have become more important in light of the proposed extension of EINSTEIN to critical infrastructure.² Indeed, as this column went to press, the *Washington Post* reported that three Internet carriers—AT&T, Verizon, and CenturyLink—were filtering traffic to

15 defense contractors using tools developed by the NSA.⁴ Of course critical infrastructure should deploy intrusion detection and intrusion prevention systems, but we are skeptical whether the consolidation and real-time information sharing model central to EINSTEIN 3 can effectively migrate to such privately held systems.

EINSTEIN Project Efforts

The purpose of the 2004 EINSTEIN was to do real-time, or near real-time automatic collection, correlation, and analysis of computer intrusion information. IDSs were to be located at federal agency access points to the Internet. If incoming traffic appeared

“anomalous,” session information would go to US-CERT, the U.S. Computer Emergency Readiness Team, a federal clearinghouse for cyber intrusion information.^a But information sharing did not happen in real time and EINSTEIN’s voluntary nature meant that many agencies did not participate.

Part of the difficulty was Internet connections. Every small, medium, and large federal agency was connected to the network, sometimes in multiple ways, making control of incoming data and real-time information sharing extremely difficult. The government went about reducing the number of federal connections to the public Internet from a few thousand to a few hundred.

The next program, EINSTEIN 2, uses devices located at the Internet access points to monitor traffic coming into or exiting from government networks and to alert US-CERT whenever traffic matching *signatures*, patterns of known malware (for example, the IP address of a server known to be hosting malware or an attachment known to include a virus), were observed in incoming packets.⁷ Participation lagged, but EINSTEIN 2 is now mandatory for federal agencies.

The third effort, EINSTEIN 3, will really up the ante by using intrusion *prevention* systems to stop malware from reaching government sites. EINSTEIN 3 devices will be performing deep packet inspection of content, discarding suspect traffic before it reaches federal systems. (The architecture is such that only communications destined for the federal government are so inspected.) As of this writing, EINSTEIN 3 has been tested only at a single medium-sized federal agency.

Because EINSTEIN IDSs and IPSs would operate on all traffic destined for federal networks, the system would intercept private communications of federal employees (for example, if a federal employee used an agency computer to check a private email account during lunch). But this surveillance is not different from that experienced by

Size is not the only issue in transitioning EINSTEIN systems from federal civilian agencies to the private sector.

employees at regulated industries using company-supplied equipment for their personal communications, and so does seem unreasonable.

EINSTEIN Project Concerns

Can EINSTEIN work? That depends on what “work” means. We have the following concerns.

Scale. Denial-of-service (DoS) attacks can be daunting; they have been measured at 100Gb/s. It is unlikely that the current generation of any network device would be able to resist the DoS attacks at this rate—let alone new attack rates likely in the near future.

Ability to do correlation. Correlation is about discovering previously unknown threats in network traffic in real time as they appear. But this is impossible to do in all but very small networks. No one knows how to use a percentage of the traffic—whether compressed, diarized,^b or sampled—to characterize arbitrary new threats. If one is hoping to deter all threats (and not just previously known ones), *all* incoming data must be correlated and analyzed. That is the crucial point.

One way to think about potential correlation solutions is that architectures can range from highly “centralized” to fully “decentralized” while sensors can be “smart” or “dumb,” that is, having the ability to do lots of computation locally, or not.

If analysis is done locally at the data collection point, then the need

a US-CERT collects information from federal agencies, industry, the research community, state and local governments, and sends out alerts about known malware; see <http://www.us-cert.gov/aboutus.html>.

b “Diarize” is used by the trade to mean making a diary of the data; in the case of a telephone call, this might be the to/from, time, and length of the call, while for IP communications, this would be the metadata of source and destination IP addresses, TCP source and destination ports, and perhaps length of packet.

Calendar of Events

August 15–19

ACM SIGCOMM 2011 Conference,
Toronto, ON Canada,
Sponsored: SIGCOMM,
Contact: Srinivasan Keshav,
Email: keshav@uwaterloo.ca

August 17–19

A Symposium on High Performance Chips,
Stanford, CT,
Contact: Ralph Wittig,
Email: ralph.witting@xilinx.com

August 21–24

The 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,
San Diego, CA,
Sponsored: SIGKDD and SIGMOD,
Contact: Chid Apte,
Email: apte@us.ibm.com

August 22–23

The Second Conference on Auctions, Market Mechanisms, and Their Applications,
New York, NY,
Contact: Sanmay Das,
Email: samnay@cs.rpi.edu

August 23

International Workshop on Smalltalk Technologies,
Edinburgh, United Kingdom,
Contact: Lagadec Loic,
Email: loic.lagadec@univ-brest.fr

August 23–26

International Conference on Queueing Theory and Network Applications,
Seoul, Republic of Korea,
Contact: Kim Bara,
Email: bara@korea.ac.kr

August 24–26

European Conference on Cognitive Ergonomics,
Rostock, Germany,
Contact: Anke Dittmar,
Email: anke.dittmar@uni-rostock.de

August 29–September 2

The 15th IEEE International EDOC Conference – The Enterprise Computing Conference,
Helsinki, Finland,
Contact: Lea Kutvonen,
Email: lea.kutvonen@cs.helsinki.fi

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



Association for
Computing Machinery

to see all incoming data requires that *all* raw signals be sent to *all* sensors. This quickly becomes unmanageable. If there are n sensors, then each sensor must look at the data from $(n-1)$ other sensors, and there are $n(n-1)/2$ pairs of data traversing the network. This is simply unmanageable when n is at all large (EINSTEIN is designed to have between one and two hundred collection points). And the sensors would also need protecting.

An alternative approach would be to centralize the data to perform the correlation. Because summarizing the data cannot solve the problem, all the data must travel through the system to the centralized detector. (We note that in an IP-based environment, packet summary information constitutes 2%–30% of the data; the numbers vary depending on whether the packets are carrying email, which comes in large packets, VoIP, which uses small packets, or something in between. In any case, summarizing does not provide savings at anything like the same scale it would for telephone communications.) This is both enormously costly for a network of any scale, as well as unable to provide the millisecond response needed in a serious attack.

(Of course, one could try a middling solution: neither fully decentralized nor fully sharing signals. Depending on where one sets collection, the problems described here will still occur. The two alternative solutions—dumb sensors and decentralized architectures or smart sensors and centralized architectures—have the worst of both worlds: they would either miss the problems, or involve enormous investment. Neither is viable.)

In short, correlation at the scale and speed at which a system serving two million users is expected to operate is not achievable using common production technology.

Device management. Many EINSTEIN devices will be in non-government facilities, but will need to be remotely controlled by US-CERT. Protecting control mechanisms and pathways against intrusion, disruption, modification, and monitoring will be very challenging.

Signature management. EINSTEIN 3 will use classified signatures devel-

oped by the government as well as unclassified signatures from commercial IDS and IPS vendors. These signatures will have to be protected from the access point operators as well as from Internet-based attackers.

These complexities make it highly unlikely that EINSTEIN can achieve the job for which it is being designed.

We have concerns about cost. If we assume the IDS/IPS function at a federal civilian agency will be similar to that in commercial network defense products built by Narus, for example, a back-of-the-envelope calculation shows each router directing traffic will require 64 times as much equipment to perform EINSTEIN-type filtering.¹ This is clearly a losing battle. In addition, it means that the EINSTEIN program, or at least the instantiation of EINSTEIN 3, will cost approximately \$1 billion just for equipment.

EINSTEIN also raises policy concerns. Any IDS looking for long-term subtle attacks must store large amounts of traffic for non-real-time analysis. System design and configuration will determine what is stored and when. The data EINSTEIN collects will have many possible uses. History has shown that investigatory tools are often misused by those with the tools.^{5,8} There is a significant risk of mission creep for EINSTEIN, and generating detailed logs for all functions that the EINSTEIN 3 device has been configured to do is crucial. Current EINSTEIN 3 documentation does not describe details of the auditing system. Given the size and scope of the EINSTEIN effort, these should be public.

What EINSTEIN can accomplish is limited. EINSTEIN documentation mentions threats of phishing, IP spoofing, botnets, denial-of-service attacks, distributed denial-of-service attacks, man-in-the-middle attacks, or the insertion of other types of malware,⁶ without noting that EINSTEIN-type systems cannot prevent IP spoofing, man-in-the-middle attacks, and some phishing attacks.

Our bigger concern, however, is in the potential of extending the centralized intrusion-detection/intrusion-prevention to critical infrastructure, including communications and public utilities such as the energy smart grid.⁹ This is contradictory—a classified U.S.

federal government program for protecting widely used private-sector systems. Exacerbating this issue is the fact that the architectures and functions of EINSTEIN and privately held critical infrastructure simply don't match.

Federal civilian systems have two million employees, but critical-infrastructure systems in the U.S. serve over 300 million Americans daily. Can a program that effectively protects the communications of federal agencies with 100,000 employees each do the same for communications giants that instead serve 100 million people? The smart grid, with hundreds of transactions per day to hundreds of millions of endpoints, far exceeds the traffic EINSTEIN is designed to handle.

Size is not the only issue in transitioning EINSTEIN systems from federal civilian agencies to the private sector. While the U.S. government can mandate the types of technologies used by federal agencies, typically the types of systems used in the private sector cannot be so mandated. The biggest problem, however, is the lack of applicability of the EINSTEIN technology to privately held critical infrastructure.

Consider the public communication networks. In the 1990s, the rate of communications transmission was sufficiently slow that the communications bits could be effectively examined and stored—at least if one did sampling. That is no longer true. Meanwhile, communications technologies are in a state of constant innovation. For proper functioning, IDS and IPS should be designed to prohibit those types of communications that are not explicitly allowed. Use of this type of block-first technologies would delay deployment of innovative communications technologies. This would have a devastating impact on U.S. innovation and competitiveness.

Or consider the power grid, which is a loosely coupled federation of many independent (sometimes competing) parties with complex trust relationships.³ This architecture vastly complicates consolidation of the type required by EINSTEIN. Even if consolidation were possible, the need for timely delivery of real-time data and the requirement of high reliability make it undesirable to circuitously direct grid control data through a

It is far from clear that this billion-dollar system can deliver sufficient security to be worth the cost.

small number of consolidated access points. In the power grid, function mismatch creates another problem. Centralized IDS/IPS solutions useful for protecting U.S. federal government computer networks may not match well to the power grid. Many parties in the energy grid already have their own IDS/IPS and firewall solutions from a variety of vendors, making the EINSTEIN 3 equipment at least partially redundant. These existing IDS/IPS solutions are often integrated with other important functionality such as quality-of-service, compression, and SCADA^c reports (which are part of Critical Infrastructure Protection requirements for the North American and Federal Energy Regulatory Commission). While these reports are generated by the same equipment that performs IDS and IPS, EINSTEIN 3 equipment cannot realistically subsume this functionality.

Putting it simply, there are deep and fundamental differences between communication networks supporting the U.S. federal government and those supporting private sector critical infrastructure. These differences create serious problems in any attempt to extend EINSTEIN-type technologies to private-sector systems controlling critical infrastructure. This is true in the U.S. and, depending on architecture, may be true elsewhere.

Conclusion

Even implementing EINSTEIN in the restricted environment of federal civilian agency systems is highly com-

plex, and it is far from clear that this billion-dollar system can deliver sufficient security to be worth the cost. In the domain of privately owned critical infrastructure, the potential of EINSTEIN is much less clear. Electronic fences protecting critical infrastructure sound good, but once one examines network architecture more carefully, EINSTEIN's fit is highly questionable. In determining how to protect critical infrastructure, one should keep in mind what Einstein himself was purported to have said: "Everything should be made as simple as possible, but no simpler"—and then develop solutions accordingly. ■

References

1. Bellare, S.M., Bradner, S.O., Diffie, W., Landau, S., and Rexford, J. *Can It Really Work? Problems with Extending EINSTEIN 3 to Critical Infrastructure*. Harvard National Security Journal, to be published.
2. Hoover, N. Cyber command director: U.S. needs to secure critical infrastructure. *Information Week* (Sept. 23, 2010); <http://www.informationweek.com/news/government/security/showArticle.jhtml?articleID=227500515>
3. Juniper Networks. *Smart Grid Security Solution: Comprehensive Network-Based Security for Smart Grid*, 2010.
4. Nakashima, E. NSA allies with Internet carriers to thwart cyber attacks against defense firms. *Washington Post* (June 16, 2011).
5. United States Congress. Senate, Select Committee to Study Governmental Operations with Respect to Intelligence Activities, Final Report of the Select Committee to Study Governmental Operations with Respect to Intelligence Activities: Supplementary Detailed Staff Reports on Intelligence Activities and the Rights of Americans: Book II, Report 94-755, 1976.
6. U.S. Department of Homeland Security, Computer Emergency Readiness Team (US-CERT). *Privacy Impact Assessment for the Initiative Three Exercise*. Washington D.C., 2010.
7. U.S. Department of Homeland Security, Chief Privacy Officer. *Privacy Impact Assessment for EINSTEIN 2*. Washington D.C., 2008.
8. U.S. Department of Justice, Office of the Inspector General (Mar. 2008). A Review of the FBI's Use of National Security Letters: Assessment of Corrective Actions and Examination of NSL Usage in 2006.
9. Zetter, K. Let us secure your network for you or face the "wild, wild west" Internet alone. *WIRED* (May 27, 2010.) <http://www.wired.com/threatlevel/2010/05/einstein-on-private-networks/>

Steven M. Bellovin (smb@cs.columbia.edu) is a professor in the computer science department at Columbia University.

Scott O. Bradner (sob@harvard.edu) is University Technology Security Officer at Harvard University.

Whitfield Diffie (the.whitfield.diffie@gmail.com) is Vice President for Information Security and Cryptography at the Internet Corporation for Assigned Names and Numbers (ICANN).

Susan Landau (susan.landau@privacyink.org) is currently a fellow at the Radcliffe Institute for Advanced Study at Harvard University.

Jennifer Rexford (jrex@cs.princeton.edu) is a professor in the computer science department at Princeton University in New Jersey.

^c SCADA (Supervisory Control And Data Acquisition) systems are used to monitor and control industrial processes.



DOI:10.1145/1978542.1978554

George V. Neville-Neil

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

Code Vicious Storage Strife

Beware keeping data in binary format.

Dear KV,

Where I work we are very serious about storing all of our data, not just our source code, in our source-code control system. When we started the company we made the decision to store as much as possible in one place. The problem is that over time we have moved from a pure programming environment to

one where there are other people—the kind of people who send email using Outlook and who keep their data in binary and proprietary formats.

At first some of us dealt with the horrifically colorful email messages by making our mail server convert all email to plain text before forwarding it, but that's not much help when peo-

ple tell you they absolutely must use Excel, and then store all of their data in it. The biggest problem is that these files take up a huge amount of space in our source-code control system, but we still don't want to store important information outside of it. Many of us are about ready to give up and just stop worrying about these types of files, and allow the company's data to be balkanized, but this doesn't seem like the right answer to me.

Binning Binary Files

Dear Binning,

While the size argument used to be a compelling one—perhaps even as recently as five years ago—we all know that terabyte disks are now cheap, and I would be quite surprised if you told me your company doesn't have a reasonably large, centralized filestore for your source-code control system. I think the best arguments against storing important company data in a proprietary or a binary format—and yes, there are open binary formats—are about control and versioning.

The versioning argument goes something like this. Let's say, for example, that the people who control your data center store their rack diagrams, which show where all your servers and network gear are located, as well as all the connections between that equipment, in a binary format. Even if the program they use to set up the files has some sort of "track changes" feature, you will have no way



of comparing two versions of your rack layouts. Any company that maintains a data center is changing the rack layout, either when adding or moving equipment or when changing or adding network connections. If a problem occurs days or weeks after a change, how are you going to compare the current version of the layout to a version from days or weeks in the past, which may be several versions back? The answer, generally, is that you cannot. Of course, these kinds of situations never come up, right? Right.

The second and I think stronger argument has to do with control of your data. If a piece of data is critical to the running of your business, do you really want to store it in a way that some other company can, via an upgrade or a bug, prevent you from using that data? At this point, if you are a trusting person, you could just store most of your data in the cloud, in something like Google Apps. KV would never do this because KV has severe trust issues. I actually think more people ought to think clearly about where they store their data and what the worst-case scenario is in relation to their data. I am not so paranoid that I do not store anything in binary or proprietary formats, and I am not so insanely religious, as some open source zealots are, as to insist that all data format must be free, but I do think before I put my data somewhere. The questions to ask are:

- What is the impact of not being able to get this data for five minutes, five hours, or five days?

- What is the risk if this data is stolen or viewed by others outside the company?

- If the company that makes this product goes out of business, then how likely is it that someone else will make a product that can read this data?

The history of computing is littered with companies that had to pay exorbitant amounts of money to dead companies to continue to maintain systems so that they would not lose access to their data. You do not wish to be one of those casualties.

KV

Dear KV,

One of the earliest hires in the company I work for seems to run many of our

More people ought to think clearly about where they store their data and what the worst-case scenario is in relation to their data.

more important programs from his home directory. These scripts, which monitor the health of our systems, are not checked in to our source-code control system. The only reason they are even mildly safe is that all of our home directories are backed up nightly. His home-directory habit drives me up a wall, and I'm sure it would aggravate you if you were working here, but I can't really scream at employee number six to clean his home directory of all important programs.

Employee 1066

Dear Employee,

I agree that you can get away with yelling at employee number six only if you are, for example, employee number two. Of course, that's rarely stopped me from yelling at people, but then I yell at everyone, so people around me are used to it. There really is no reason for allowing anyone, including a high-ranking engineer, to run code from a home directory. Home directories are for a person's personal files, checkout from source-code control, temporary files, generated data the person does not need to share, and, of course, pirated music and videos. All right, perhaps that last one should not be there, but it's better than putting it on the central file server!

There are two problems with people running things from their home directories. The first is the issue of what happens when they quit or are fired. At that point you have to lock

them out of the account, but the account has to remain active to run these programs to maintain your systems. Now you have an emergency on your hands, as you immediately have to convert all these programs—without the authors' help—to be generic enough to run in your system. Such programs often depend on accreted bits of the author's environment, including supporting scripts, libraries, and environment variables that are set only when the original author logs into the account.

The second problem is that the user who runs these programs usually has to have a high level of privilege to run them. Even if the person is not actively evil, the consequences of that person making a mistake while logged in as himself/herself are much greater if the person has high privileges. In the worst cases of this, I have seen people who have accounts that, while they aren't named root, have rooty powers when they are logged in, meaning any mistake, such as a stray `rm *` in the wrong directory, would be catastrophic. "Why are they running as root?" I hear you cry. For the same reason that everyone runs as root, because anything you do as root always succeeds, whether or not it was the right thing to do.

I know this is out of character, but if you are not the yelling type, I suggest nagging, cajoling, and even offering to convert the code yourself in order to get it out of this person's home directory.

KV

Q Related articles on queue.acm.org

Beyond Relational Databases

Margo Seltzer

<http://queue.acm.org/detail.cfm?id=1059807>

Sifting Through the Software Sandbox: SCM Meets QA

William W. White

<http://queue.acm.org/detail.cfm?id=1046945>

The Seven Deadly Sins of Linux Security

Bob Toxen

<http://queue.acm.org/detail.cfm?id=1255423>

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

Copyright held by author.

Viewpoint

Journals for Certification, Conferences for Rapid Dissemination

Rethinking the role of journals in computer science.

THE PUBLICATION CULTURE in computer science is different from that of all other disciplines. Whereas other disciplines focus on journal publication, the standard practice in CS has been to publish in a conference and then (sometimes) publish a journal version of the conference paper. Indeed, it is through publication in selective, leading conferences that the quality of CS research is typically assessed.

Why should a researcher publish a journal version of a paper? In fields other than CS, which place no value on conference publication, there are two main reasons:

- **Certification.** Publication in a peer-reviewed journal is a signal to the world that the paper has passed a minimal level of competence; publication in a leading journal confers even more prestige.

- **Publicity.** Journal publication can be an effective way to tell the world (including policymakers and science advisors, not just colleagues) about the research, particularly publication in a leading journal like *Science* or *Nature*.

In CS, the situation is different. In many subdisciplines of CS, having a paper accepted at a leading conference already gives as much of a certification as getting it into a leading journal. Conferences play the role of publicity as well. The best way to get your subdiscipline to know about your results is to publish them in the leading con-

We can already see the beginning of a shift in the conference and journal landscape.

ference for that subdiscipline.

There is increasing debate about the role of conferences in our field.^{1,5,8} Fortnow³ argues that our field suffers from the current use of conferences for certification.^a Two particular problems he cites are those of quality and innovation suffering because we end up living in a deadline-driven world, and the splintering of the field into multiple conferences (so that there are enough publication venues), leading to conferences failing to act as a broad forum and bring their communities together. Fortnow suggests conferences should be held less frequently, and accept every reasonable paper for presentation without proceedings.

^a We remark that Fortnow's Viewpoint is not universally accepted; see, for example, the different reactions noted in in Vardi's March 2010 *Communications' Editor's Letter*.⁶

Even in the current situation, journals do play a role in providing more relaxed page limits, which allow authors to include more discussion, more expository details, details of proofs, additional experimental results, and the time to submit a more polished, thoughtful paper. For theoretical papers, the certification issue remains significant because it is rare that conference reviewers review proofs as thoroughly as journal reviewers. Publication in a journal also adds value through a paper going through a strict review process with several iterations. Finally, journals also provide publicity for interdisciplinary work.

We can already see the beginning of a shift in the conference and journal landscape. Part of the shift involves journals publishing conference proceedings as special issues. For example, *ACM Transactions on Graphics (TOG)* publishes every SIGGRAPH and SIGGRAPH Asia technical paper in its biannual conference issues, which replace traditional conference proceedings. If a paper is conditionally accepted for presentation at SIGGRAPH, then the paper undergoes a second review (by one of the original reviewers) to ensure all changes requested by the reviewers are made; it is then also published in *TOG*. Similarly, papers accepted at this year's International Conference on Logic Programming (ICLP) will appear in *Theory and Practice of*

Logic Programming. However, note that this approach means these journal issues suffer from the same problems that conference publications suffer from: papers are subject to the conference length restrictions and paper submission deadlines that make it difficult to do serious revisions.

The database community has taken this one step farther. Currently the only way to submit a paper to the VLDB (Very Large Database) conference is to submit it to the journal *Proceedings of the VLDB Endowment (PVLDB)*. Continuous submissions are accepted throughout the year, reviews guaranteed within two months, a full review cycle including checking of final versions by responsible editors is supported, and papers accepted by a specified date are offered a presentation slot in the next VLDB conference. This change makes it possible to have a “revise and resubmit review,” but there still remain serious page limitations. (Interestingly, *PVLDB* also allows the publication of an extended version of a *PVLDB* paper in another journal.)

The *TOG/SIGGRAPH* relationship has another facet, which illustrates another possibility for a conference-journal relationship: *TOG* allows any author of a published paper to present the work at SIGGRAPH, while operating without deadlines and less strict page limits.

These experiments suggest that the CS community needs to think through the intertwined role of conferences and journals, especially in light of the growing amount of research at the intersection of computer science and other fields. Our unique distinction of being a conference-led field leads to a particular problem for multidisciplinary work because, outside of CS, journals typically have all the power, and are very reluctant to take papers where versions have appeared previously. For example, some leading biology journals are unwilling to publish work that has appeared in RECOMB (Conference on Research in Computational Molecular Biology) or ISMB (Conference on Intelligent Systems for Molecular Biology); this has caused a problem for researchers in computational biology. Some interdisciplinary conferences are sensitive to this issue (including ACM Electronic Com-



merce), and allow full papers to be submitted and reviewed, but then published as a one-page abstract in the proceedings. However, this is not the norm.

Future Role of Journals

We offer the following vision of the future role of journals within CS, with some thoughts on how to make it come about. Many of these ideas have been suggested and, indeed, some have even been tried. But more serious experimentation is needed. Our vision of the future includes the following elements.

Papers will be available on public Web archives such as CoRR, the Computing Research Repository (see <http://arxiv.org/corr>), the CS part of the arXiv.

This is increasingly common now, as researchers are discovering the advantages of posting papers on managed archives rather than just having them on their own home pages. But if all papers are available in one place, then making a paper stand out from the pack will become more significant. One of the best ways of doing this will be via certification.

Journals will be the main “certification” authorities, because they can operate without deadlines or strict page limits, allow for a careful review cycle with checking of results, are compatible with other scientific disciplines, and promote thoughtful work. (By “journal” we simply mean an editor-in-chief together with an editorial board recognized by the community as a certifying authority. Journals like *Journal of Artificial Intelligence Research (JAIR)* and *Logical Methods in Computer Science (LMCS)* demonstrate that a group of community members can start viable, well-respected journals without the support of a publisher.) To paraphrase Churchill, we believe that journal reviews are the worst form of certification, except for all the other ones that have been proposed now that papers are available online. For example, citation counts (and page rank style variants of citation counts) suffer from well-known problems, including the fact that different fields have different citation rates, they can be influenced by fads,

and the counts depend on the database of papers being used (see van Noorden⁴).^b They cannot, for example, tell an economist that a computer science paper on game theory is relevant to economists, nor can they certify the correctness of results.^c Nevertheless, citation counts are becoming increasingly important. An advantage of greater CS use of journals is that it would allow CS citations to be more comparable to those of other fields. This is particularly significant when CS researchers compete for, say, funding at the national level.

Journals will need to be much faster in reviewing papers to play an important role in CS. Indeed, this will be essential in supporting the promising trend toward using journals also as a deadline-free path to conference presentation, in addition to protecting their more traditional role. Of course, de-emphasizing the importance of conferences will help to achieve this, by significantly reducing the work load on conference reviewers. Here are some additional suggestions for speeding up the review cycle:

- It seems sensible to adopt a page limit for papers that require fast review, for example, to facilitate presentation at an upcoming conference. The resulting process would still allow for a full review cycle and continuous submission throughout the year. (Having said that, we believe it is critical that there be enough journals that do not have significant page limitations and allow for longer, thoughtful articles.)

- For conferences that maintain their own review process, better coordination with journals would allow the same reviewer to read both the conference and an extended journal version.

^b A recent *Nature* editorial assessing assessment and a collection of metrics-related articles are available at <http://www.nature.com/metrics>.

^c Another certification approach that has been suggested is to have people just write reviews of papers, and attach them to the papers, without the need for recognized certification authorities. Again, while we believe that such reviews can play a useful role, we are not aware of any such system that has succeeded. Part of the problem is that the people whose reviews we would most like to read are busy; another is that a rather idiosyncratic set of papers will be reviewed this way.

Our unique distinction of being a conference-led field leads to a particular problem for multidisciplinary work because, outside of CS, journals typically have all the power, and are very reluctant to take papers where versions have appeared previously.

- More cooperation between journals would also be helpful. For example, a journal could agree to pass on reviews of a paper and the names of the reviewers to another journal at the request of authors (subject to the agreement of the reviewers). Halpern followed this policy as editor-in-chief of the *Journal of the ACM*; authors of rejected papers could request reviews to be passed on to the journal of their choice.

Each certification could come with a “cost”: For every paper that is reviewed by a journal, some author of that paper must be available to review another paper (Crowcroft et al.² make a similar point) and/or there could be a cost for submission. Both approaches are in fact used in the *B.E. Journal of Theoretical Economics* (see <http://www.bepress.com/bejte/policies.html>). Currently, authors can choose either to commit to reviewing two papers in a timely way (within 21 days of receiving it) or paying \$350 when they submit a paper. If an author agrees to review a paper and his/her review is late, then there is a financial penalty (currently \$200).

Certifications need not be mutually exclusive. We believe the commu-

nity should experiment with different forms of certification. For example, those who work in interdisciplinary areas may choose to write two versions of a paper, targeted to different communities, and then get certification from the appropriate journals for each of the two versions. It may even make sense to get certifications from two (or more!) different communities for the same paper.^d

The conference culture has served CS well up to now. Conferences provide authors with useful feedback; they are also a great forum for meeting colleagues. We should debate whether conferences should continue in the same role in the future, now that CS has matured and is making connections to so many other fields. As we have tried to make clear, the role of journals, and how certification will be carried out, needs to be an important part of this debate. ■

^d We do not believe copyright issues will present a serious impediment (any more than they are now in CS when different versions of a paper appear in both a journal and conference), and expect to see a continuing trend where authors do not give exclusive copyright to journals, giving them an assent to publication instead.

References

1. Birman, K. and Schneider, F.B. Program committee overload in systems. *Commun. ACM* 52, 5 (May 2009), 34–37.
2. Crowcroft, J., Keshav, S. and McKeown, N. Scaling the academic publication process to Internet scale. *Commun. ACM* 52, 1 (Jan. 2009), 27–30.
3. Fortnow, L. Time for computer science to grow up. *Commun. ACM* 52, 8 (Aug. 2009), 33–35.
4. van Noorden, R. A profusion of measures. *Nature* 465:864–866 (2010).
5. Vardi, M.Y. Conference vs. journals in computing research. *Commun. ACM* 52, 5 (May 2009).
6. Vardi, M.Y. Revisiting the publication culture in computing research. *Commun. ACM* 53, 3 (Mar. 2010), 5.
7. Wallach, D. Rebooting the CS publication process. Freedom to Tinker, June 2010; <http://www.freedom-to-tinker.com/blog/dwallach/rebooting-cs-publication-process>.
8. Wing, J.M. Breaking the cycle. *Commun. ACM* 52, (Dec. 2009), 8–9.

We thank Kavita Bala, Ken Birman, Lance Fortnow, Nir Friedman, Johannes Gehrke, Paul Ginsparg, Jim Hendler, John Hopcroft, Doug James, Christoph Koch, Jon Kleinberg, Daphne Koller, Steve Marschner, Michael Mitzenmacher, Andrew Myers, Greg Morrisett, Moshe Vardi, and two anonymous reviewers of this Viewpoint for their comments.

Joseph Y. Halpern (halpern@cs.cornell.edu) is a professor in the computer science department at Cornell University, Ithaca, NY.

David C. Parkes (parkes@eecs.harvard.edu) is a Gordon McKay Professor of Computer Science in the School of Engineering and Applied Sciences at Harvard University, Cambridge, MA.

Copyright held by author.

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

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

Seeking a middle ground.

BY ERIC ALLMAN

The Robustness Principle Reconsidered

“BE CONSERVATIVE IN what you do, be liberal in what you accept from others.” (RFC 793)

In 1981, Jon Postel formulated the Robustness Principle, also known as Postel’s Law, as a fundamental implementation guideline for the then-new TCP. The intent of the Robustness Principle was to maximize interoperability between network service implementations, particularly in the face of ambiguous or incomplete specifications. If every implementation of some service that generates some piece of protocol did so using the most conservative interpretation of the specification and every implementation that accepted that piece of protocol interpreted it using the most generous interpretation, then the chance that the two services would be able to talk with each other would be maximized. Experience with the Arpanet had shown that getting

independently developed implementations to interoperate was difficult, and since the Internet was expected to be much larger than the Arpanet, the old ad hoc methods needed to be enhanced.

Although the Robustness Principle was specifically described for implementations of TCP, it was quickly accepted as a good proposition for implementing network protocols in general. Some have applied it to the design of APIs and even programming language design. It’s simple, easy to understand, and intuitively obvious. But is it correct?

For many years the Robustness Principle was accepted dogma, failing more when it was ignored rather than when practiced. In recent years, however, that principle has been challenged. This isn’t because implementers have gotten more stupid, but rather because the world has become more hostile. Two general problem areas are impacted by the Robustness Principle: orderly interoperability and security.

Standards and Interoperability

Interoperability in network protocol implementations is a Hard Problem.TM There are many reasons for this, all coming down to the fundamental truth that computers are unforgiving. For example, the specification may be ambiguous: two engineers build implementations that meet the spec, but those implementations still won’t talk to each other. The spec may in fact be unambiguous but worded in a way that some people misinterpret it. Arguably some of the most important specs fall into this class because they are written in a form of legalese that is unnatural to most engineers. The specification may not have taken certain situations (for example, hardware failures) into account, which can result in cases where making an implementation work in the real world requires violating the spec.

In a similar vein, the specification may make implicit assumptions about the environment (for example, maxi-

BE

CONSERVATIVE

IN WHAT YOU DO,

BE LIBERAL

IN WHAT YOU

ACCEPT FROM

OTHERS.

mum size of network packets supported by the hardware or how a related protocol works), and those assumptions are incorrect or the environment changes. Finally, and very commonly, some implementers may find a need to enhance the protocol to add new functionality that isn't defined by the spec.

Writing standards (that is, any specification that defines interoperability between different implementations) is an art. Standards are essentially contracts, in the legal sense, but the law has the advantage (or perhaps disadvantage) of a long history of definition, redefinition, and refinement of definition, usually in case law. The goal of a standard is to make interoperability possible. That requires both precision (to avoid ambiguity) and clarity (to avoid misinterpretation). Failure in either way results in a lack of interoperability. Unfortunately, these two goals are sometimes at odds, as noted.

Our normal human language is often ambiguous; in real life we handle these ambiguities without difficulty (or use them as the basis for jokes), but in the technical world they can cause problems. Extremely precise language, however, is so unnatural to us that it can be hard to appreciate the subtleties. Standards often use formal grammar, mathematical equations, and finite-state machines in order to convey precise information concisely, which certainly helps, but these do not usually stand on their own—for example, grammar describes syntax but not semantics, equations have to be translated into code, and finite-state machines are notoriously difficult for humans to understand.

Standards often include diagrams and examples to aid understandability, but these can actually create problems. Consider the possibility that a diagram does not match the descriptive text. Which one is correct? For that matter, any time the same thing is described in two places there is a danger that the two descriptions may say subtly different things. For example, RFC 821 and RFC 822 both describe the syntax of an email address, but unfortunately they differ in minor ways (these standards have since been updated to fix this and other problems). A common solution is always to include



For many years the Robustness Principle was accepted dogma, failing more when it was ignored rather than when practiced. In recent years, however, that principle has been challenged.



necessary duplicate language “by reference” (that is, including a reference to another document rather than an actual description). Of course, taken to an extreme, this can result in a rat's nest of standards documents. For example, the OSI recommendations (standards) for message handling (email) are contained in about 20 different documents filled with cross-references.

Even using examples can be controversial. Examples are never *normative* (standards buzzword for *authoritative*); that is, if there is a conflict between an example and the body of the text, the text wins. Also, examples are seldom complete. They may demonstrate some piece of the protocol but not all the details. In theory if you removed all the examples from a standard, then the meaning of that standard would not change at all—the sole *raison d'être* being to aid comprehension. The problem is that some implementers read the examples (which are often easier to understand than the actual text of the standard) and implement from those, thus missing important details of the standard. This has caused some authors of standards to eschew the use of examples altogether.

Some (usually vendor-driven) standards use the “reference implementation” approach—that is, a single implementation that is defined to be correct; all other implementations are in turn correct if and only if they work against the reference implementation. This method is fraught with peril. For one thing, no implementation is ever completely bug-free, so finding and fixing a bug in the reference implementation essentially changes the standard.

Similarly, standards usually have various “undefined” or “reserved” elements—for example, multiple options with overlapping semantics are specified at the same time. Other implementations will find how these undefined elements work and then rely on that unintended behavior. This creates problems when the reference implementation is extended to add functionality; these undefined and reserved elements are typically used to provide the new functions. Also, there may be two independent implementations that each work against the refer-

ence implementation but not against each other. All that said, the reference implementation approach could be useful in conjunction with a written specification, particularly as that specification is being refined.

The original InterOp conference was intended to allow vendors with Network File System (NFS) implementations to test interoperability and ultimately demonstrate publicly that they could interoperate. The first 11 days were limited to a small number of engineers so they could get together in one room and actually make their stuff work together. When they walked into the room, the vendors worked mostly against only their own systems and possibly Sun's (since as the original developer of NFS, Sun had the reference implementation at the time). Long nights were devoted to battles over ambiguities in the specification. At the end of those 11 days the doors were thrown open to customers, at which point most (but not all) of the systems worked against every other system. By the end of that session the NFS protocol was much better understood, many bugs had been fixed, and the standard was improved. This is an inevitable path for implementation-driven standards.

Another approach to standards is to get a bunch of smart people in a room to brainstorm what the standard should do, and only after the standard is written should the code be implemented. This most closely matches conventional software engineering, where a specification is written before the code. Taken to extreme, this is the waterfall model. The problem with producing standards this way is the same as occurs with the waterfall model: the specification (standard) sometimes mandates things that are only marginally useful but are difficult or impossible to implement, and the cost of going back and modifying the specification goes up exponentially with time.

Perhaps the best situation of all is where the standards and implementations are being developed in parallel. When SMTP (Simple Mail Transfer Protocol) was being developed, I was in the unusual position of developing the Sendmail software contemporaneously with the standard itself. When

updates to the draft standard were proposed, I was able to implement them immediately, often overnight, which allowed the standard and the implementation to evolve together. Ambiguities in the standard were exposed quickly, as were well-meaning features that were unnecessarily difficult to implement. Unfortunately, this is a rare case today, at least in part because the world has gotten sufficiently complex that such quick updates in standards are no longer easy.

Ambiguity and Extensibility in Standards

As an example of ambiguity, consider the following excerpt from a (mythical) standard:

If the *A* option is specified in the packet, field *X* contains the value of the parameter.

This assumes a protocol that has a fixed-size header. *A* is probably a bit in some flags field, and *X* is some field in the packet. On the surface this description seems pretty clear, but it does not specify what field *X* means if the *A* option is *not* specified. A better way to word this might be:

If the *A* option is specified in the packet, field *X* contains the value of the parameter; otherwise field *X* must be zero.

You might be thinking that this wording should be unnecessary—*of course* *X* should be zero, so why bother being that explicit? But without this detail, it could also mean: “If the *A* option is not specified in the packet, field *X* is ignored”—or, perhaps, “field *X* is undefined.” Both of these are substantially different from the “must be zero” interpretation. Furthermore, the difference between these two wordings is trivial but significant. In the former case “ignored” might mean “must be ignored” (that is, under no circumstances should field *X* be used if option *A* is not specified). But the latter case allows the possibility that field *X* might be reused for some other purpose.

Which (finally) brings us back to the Robustness Principle. Given the “must be zero” specification, to be most robust any implementation would be sure to zero the *X* field before sending a packet (be conservative in what it sends) but would not check the *X* field

upon receipt (be liberal in what it accepts).

Now suppose our standard is revised (version 2) to add a *B* option (which cannot be used in conjunction with option *A*) that also uses the *X* field. The Robustness Principle has come to our rescue: since “robust” version 1 implementations should not check the value of field *X* unless option *A* has been specified, there will be no problem adding an option *B*. Of course, version 1 receivers won't be able to provide the option *B* functionality, but neither will they be upset when they receive a version 2 packet. This is a good thing: it allows us to expand protocols without breaking older implementations.

This also clarifies what to do when passing on a packet—implementations should *not* clear field *X*, even though that is the most “conservative” thing to do, because that would break the case of a version 1 implementation forwarding a packet between two version 2 implementations. In this case the Robustness Principle must include a corollary: implementations should silently ignore and pass on anything that they do not understand. In other words, there are two definitions of “conservative” that are in direct conflict.

Now let's suppose that our mythical standard has another field *Y* that is intended for future use—that is, in a protocol extension. There are many ways to describe such fields, but common examples are to label them “reserved” or “must be zero.” The former does not say what value a compliant implementation should use to initialize reserved fields, whereas the latter does, but it is usually assumed that zero is a good initializer. Applying the Robustness Principle makes it easy to see that when version 3 of the protocol is released using field *Y* there will be no problem, since all older implementations will be sending zero in that field.

An example protocol diagram.

A	B	...
X	Y (reserved)	
undefined	must be zero	

The Dark Side

But what happens if there are implementations that do not set field *Y* to zero? The field is never initialized, so it is left containing whatever garbage happened to be previously in memory. In such a situation the incorrect (insufficiently conservative) implementation might happily survive and interoperate with other implementations even though it did not technically match the specification. (Such an implementation is also probably leaking information—a security problem.) Alternatively, an implementation might commandeer the *Y* field for some other use (“after all, it’s not being used, so I might as well use it”). The effect is the same.

What has happened here is the bad implementation has survived because of the liberality of all the other implementations. This will never be detected until version 3 of the protocol comes along—or some implementation violates the “accept” side of the Robustness Principle and starts doing more careful checking. In fact, some protocol implementations have special testing modules that are “conservative in what they accept” in order to ferret out these problems, but many do not.

The final indignity may occur when version 3 is being developed and we discover that too many implementations (or one very popular one) are not being sufficiently conservative in what they generate. Lest you think that this must be rare, there are innumerable cases where a vendor has found a convenient “reserved” field and commandeered it for its own use.

I’ve framed this example as though it involved low-level network packets (à la Figure 3 in Transmission Control Protocol RFC 793),² but this can easily be applied to common protocols such as XML. The same difficulties (commandeering a tag that later gets used, removing unrecognized attributes, and so on) all apply here. (For a good argument about why the Robustness Principle shouldn’t be applied to XML, see Tim Bray’s “On Postel, Again.”¹)

Security

The Robustness Principle was formulated in an Internet of cooperators. The world has changed a lot since



Sometimes interoperability and security are at odds with each other. In today’s climate they are both essential. Some balance must be drawn.



then. Everything, even services that you may think you control, is suspect. It’s not just user input that needs to be checked—attackers can potentially include arbitrary data in DNS (Domain Name System) results, database query results, HTTP reply codes, you name it. Everyone knows to check for buffer overflows, but checking incoming data goes far beyond that.

► You might be tempted to trust your own corporate database, but consider how the data was added in the first place. Do you trust every piece of software that might update that database to do strict checking? If not, you should do your own checking.

► Do you get your data over a TCP connection that goes through your firewall? Have you considered the possibility of connection hijacking? Security-critical data should be accepted only over encrypted, signed connections. Other data should be carefully checked.

► Do you trust connections that come in from computers inside your firewall? Have you ever heard of viruses? Even machines that you think are under your control may have been subverted.

► Do you trust command-line flags and environment variables? If someone has managed to get an account on your system, these might be used for privilege escalation.

The atmosphere of the Internet has changed so much that the Robustness Principle has to be severely reinterpreted. Being liberal in what you accept can contribute to security problems. Sometimes interoperability and security are at odds with each other. In today’s climate they are both essential. Some balance must be drawn.

A Liberality Too Far

Errors can be made on both sides of the Robustness Principle. The previous “zeroing the *X* field” example is a case of being too conservative in what you generate, but most of the reevaluation is coming from the “be liberal in what you accept” side.

The problem occurs in how far to go. It’s probably reasonable not to verify that the “must be zero” fields that you don’t have to interpret are actually zero—that is, you can treat them as undefined. As a real-world example, the

SMTP specification says that implementations must allow lines of up to 998 characters, but many implementations allow arbitrary lengths; accepting longer lines is probably OK (and in fact longer lines often occur because a lot of software transmits paragraphs as single lines). Similarly, although SMTP was defined as a seven-bit protocol, many implementations handle eight-bit characters with no problem, and much of the world has come to rely on this (there is now an SMTP extension to specify eight-bit characters that formalizes this behavior).

On the other hand, an implementation that has to deal with digital signatures should probably be very strict in interpreting public certificates. These are usually encoded as BASE64, which uses 65 characters for encoding (the upper- and lowercase characters, the digits, “+”, “/”, and “=”, all of which can be represented in seven-bit US-ASCII). If other characters occur in a certificate, it could result from a security attack and hence should probably be rejected. In this case, an overly liberal implementation might just ignore the unrecognized characters.

In fact, one of the principles of software reliability and security is always to check your input. Some people interpret this to mean user input, but in many cases it means checking everything, including results from local “cooperating” services and even function parameters. This sound principle can be summarized as “be conservative in what you accept.”

Generality

I’ve described this problem as though I were looking at a protocol such as TCP, with fixed-bit fields in packets. In fact, it is much more common than that. Consider some real-world examples:

- Some MIME implementations interpret MIME header fields such as Content-Type or Content-Transfer-Encoding even when no MIME-Version header field has been included. Inconsistencies in these implementations are one of the causes of mangled messages being received.

- Far too many Web servers accept arbitrary data from users and other services and process them without first checking the data for reasonableness. This is the cause of SQL injection attacks.

Note that this can also be taken to the opposite extreme: many Web servers won’t allow you to specify email addresses with “+” in them, even though this is completely legal (and useful). This is a clear example of being too conservative in what you accept.

- Older minicomputers often did useful things when the instructions contained bit patterns that were supposed to be illegal (or at least undefined). Instead of throwing a fault (which was expensive), the hardware would do whatever was convenient for the hardware designers. Overly tricky software engineers would sometimes discover these oddities and use them, which resulted in programs that would not work when you upgraded the hardware. The DEC PDP-8 was an example of such a machine. It turns out that it was possible to ask the machine to shift left and shift right in one instruction. On some models this had the effect of clearing both the high-order and low-order bits, but on other models it did other things.

- Sendmail has been criticized for being too liberal in what it accepts. For example, Sendmail accepted addresses that did not include a domain name on the From header field value, “fixing” them by adding the local domain name. This works fine in most corporate settings but not in hosted environments. This liberality puts very little pressure on authors of mail submitters to fix the problem. On the other hand, there are those who say that Sendmail should have accepted the bogus address but not attempted to correct it (i.e., it was too conservative, not too liberal).

- Many Web browsers have generally been willing to accept improper HTML (notably, many browsers accept pages that lack closing tags). This can lead to rendering ambiguities (just where does that closing tag belong, anyhow?), but is so common that the improper form has become a de facto standard—which makes building any nontrivial Web page a nightmare. This has been referred to as “specification rot.”

All of these are, in one way or another, examples of being too liberal in what you accept. This in turn allows implementations to perpetuate that

aren’t conservative enough in what they generate.

What Now?

So, what to do? Is the Robustness Principle just wrong? Perhaps it is more robust in the long run to be conservative in what you generate and even more conservative in what you accept. Perhaps there is a middle ground. Or perhaps the Robustness Principle is just the least bad of a bunch of bad choices.

Well, not really. Remember that being liberal in what you accept is part of what allows protocols to be extended. If every implementation were to insist that field *Y* actually be zero, then rolling out version 3 of our protocol would be nearly impossible. Nearly every successful protocol needs to be extended at some point or another, either because the original design didn’t account for something or because the world changed out from under it. Change is constant, and the world is a hostile place. Standards—and implementations of those standards—need to take change and danger into account. And like everything else, the Robustness Principle must be applied in moderation. ■

Related articles on queue.acm.org

The Deliberate Revolution

Mike Burner

<http://queue.acm.org/detail.cfm?id=637960>

Toward a Commodity Enterprise Middleware

John O’Hara

<http://queue.acm.org/detail.cfm?id=1255424>

Wireless Networking Considered Flaky

Eric Allman

<http://queue.acm.org/detail.cfm?id=957735>

References

1. Bray, T. On Postel, Again (2004; <http://www.tbray.org/ongoing/When/200x/2004/01/11/PostelPilgrim>).
2. Transmission Control Protocol RFC 793, Figure 3 (1981); <http://datatracker.ietf.org/doc/rfc793/>.

Acknowledgments

Thanks to Kirk McKusick for the details of the first InterOp and to George Neville-Neil, Stu Feldman, and Terry Coatta for their helpful suggestions.

Eric Allman is the cofounder and chief science officer of Sendmail, one of the first open-source-based companies. He was previously the lead programmer on the Mammoth project at UC Berkeley. He was chief programmer on the INGRES database management project and was involved with the early Unix effort at Berkeley. Allman is a founding member of the *ACM Queue* Editorial Advisory Board.

© 2011 ACM 0001-0782/11/08 \$10.00

Heterogeneous systems allow us to target our programming to the appropriate environment.

BY SATNAM SINGH

Computing Without Processors

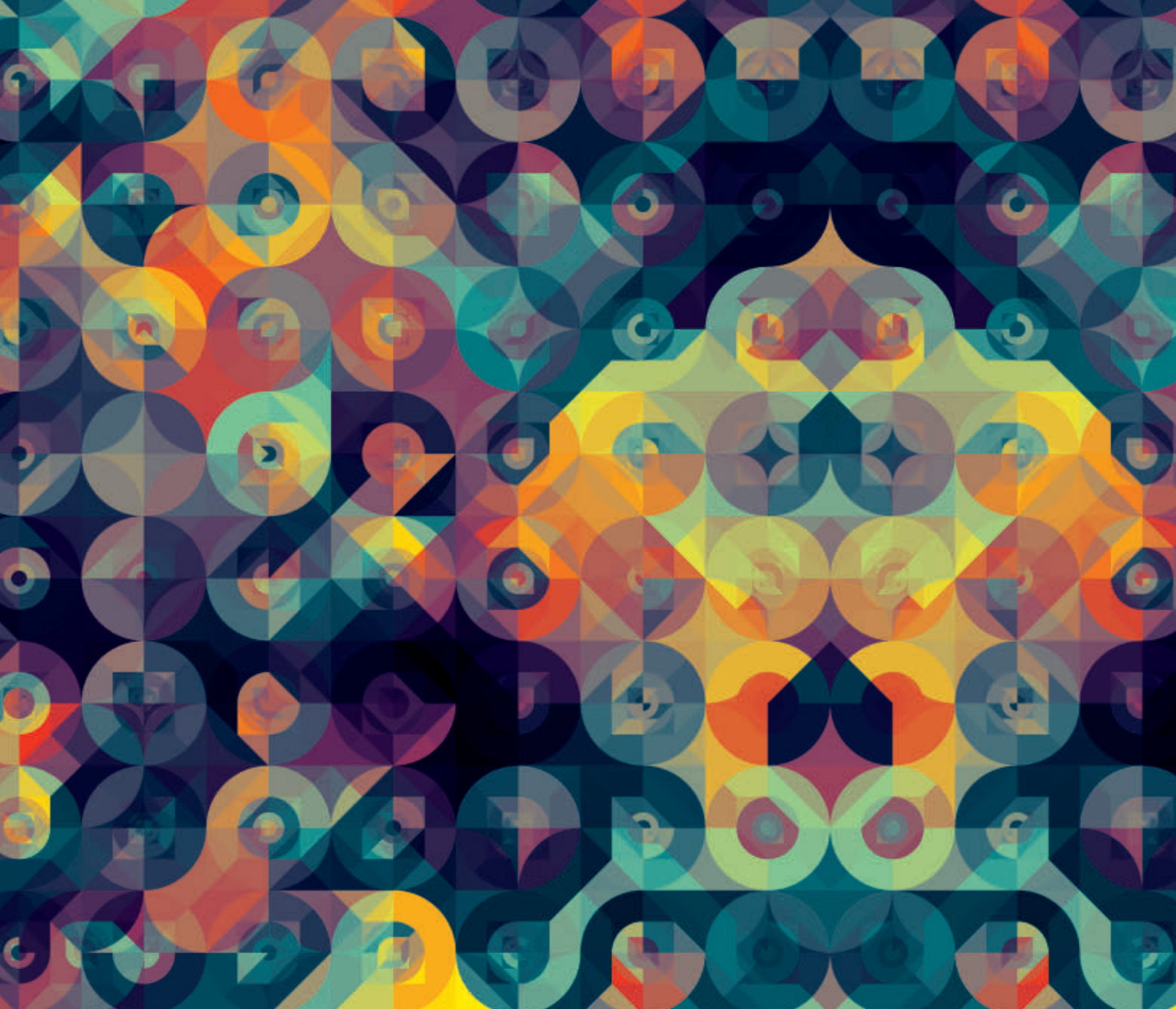
FROM THE PROGRAMMER'S perspective the distinction between hardware and software is being blurred. As programmers struggle to meet the performance requirements of today's systems they will face an ever increasing need to exploit alternative computing elements such as graphics processing units (GPUs), which are graphics cards subverted for data-parallel computing,¹¹ and field-programmable gate arrays (FPGAs), or soft hardware.

The current iteration of mainstream computing architectures is based on cache-coherent multicore processors. Variations on this theme include Intel's experimental Single-Chip Cloud Computer, which contains 48 cores that are not cache coherent. This path, however, is dictated by the end of frequency scaling rather than being driven by requirements about how programmers wish to write software.⁴ The conventional weapons available for writing concurrent and parallel software for such multicore

systems are largely based on abstractions developed for writing operating systems (for example, locks and monitors). However, these are not the correct abstractions to use for writing parallel applications.

There are better ways to bake all that sand. Rather than composing many elements that look like regular CPUs, a better approach, from a latency and energy-consumption perspective, is to use a diverse collection of processing elements working in concert and tuned to perform different types of computation and communication. Large coarse-grain tasks are suitable for implementation on multicore processors. Thousands of fine-grain data-parallel computations are suitable for implementation on GPUs. Irregular





fine-grain tasks that need to run with extreme requirements for performance or reduced energy consumption are suitable for implementation as digital circuits running on an FPGA chip. The future is heterogeneous.

The emergence of such heterogeneous systems challenges the comfortable distinction that has been established between the design processes for hardware and software. The instruction set architecture (ISA) of the CPU has acted as a convenient interface layer. The software industry was predicated on the assumption that there is only one kind of hardware (the CPU) with a standardized interface (the ISA). This allowed the hardware industry to continually innovate a series of microarchitectures that con-

formed to the fixed ISA interface. This cozy demarcation is now being eroded as programmers seeking the absolute maximum performance or reduced energy consumption resort to alternative processing elements such as GPUs and FPGAs.

One place we are likely to see the deployment of heterogeneous computing systems is in the cloud, where we can imagine racks populated with not only multicore processors but also GPUs and FPGAs. Amazon has already created the Elastic Compute Cloud that allows computations to be executed on GPUs. Some kinds of computations can be executed on GPUs and FPGAs at a performance-per-dollar ratio that is significantly better than what is achievable with a CPU. As energy use becomes

more of a limiting factor in the growth of data centers the deployment of heterogeneous architectures to help reduce latency and energy consumption will be inevitable. Such systems will need programming models and runtime infrastructures that allow the remapping of computations to different types of processing hardware, depending on the computational demands and nonfunctional requirements (energy versus latency). These systems will even open up the opportunity to charge based on energy consumption rather than time or cycles.

To make heterogeneous computing in the cloud a reality there are still many technical challenges to overcome: for example, how to virtualize GPU and FPGA computations so they

can be moved to different physical resources and how to guard against security violations.⁹

Another challenge introduced by heterogeneous systems will be the design of data structures that are suited for use on multicore processors and heterogeneous computing elements connected with a complex memory hierarchy or on-chip network. This will force us to rethink structures such as stacks and queues and instead consider unordered concurrent constructs that allow greater flexibility for memory access and layout.¹²

What Is Possible Today

Modern heterogeneous systems are evolving toward a future of intriguing computing possibilities. Today's systems are in various stages of development in achieving that goal, including the refinement of architectures that allow parallel processing, innovations to existing programming languages, and the expansion of language portability to GPUs, AVX (Intel's Advanced Vector Instructions), and FPGAs.

More than nested parallel for loops. The most popular heterogeneous computing resource today is the GPU, which provides architecture with a high aggregate-memory

bandwidth and the ability to perform many more data-parallel operations than is possible with a conventional processor. Other heterogeneous computing elements such as FPGAs can also efficiently realize data-parallel operations, as well as supporting high-speed irregular operations such as processing XML queries at "line speed" (that is, in real-time as data flows over a high-speed network).

The exploitation of new data-parallel computing elements has naturally drawn on the experience of parallel programming in high-performance scientific computing (for example, MPI and OpenMP). Many useful applications are not data parallel yet do permit other kinds of parallelization (for example, asynchronous communicating threads where each performs a different task).

To make effective use of heterogeneous computing programmers need to look beyond the problem of how to parallelize nested for loops and consider how to compute as much as possible in parallel in the highly irregular structure of modern applications. Techniques that make it easier to exploit irregular parallelism include software transactional memory (STM) and automatic mutual exclusion (AME).¹

Data-parallel computing with GPUS.

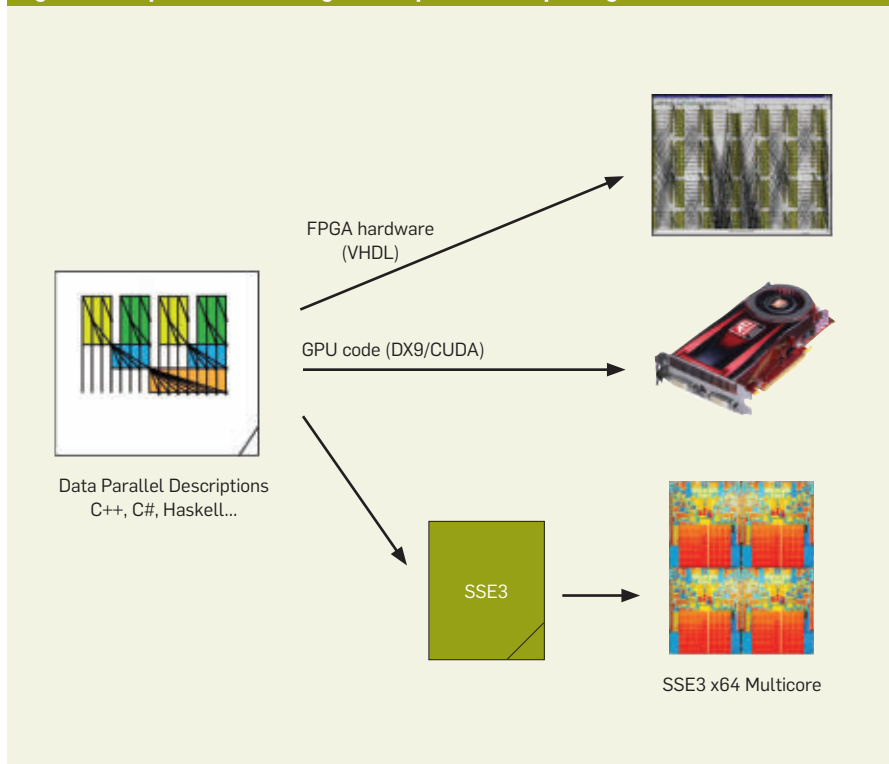
Although GPUs can offer substantial performance improvements for data-parallel algorithms, they are far from trivial to program even with special GPU programming languages and systems such as Nvidia's Compute Unified Device Architecture (CUDA).¹⁰ To get good performance from GPUs, one undergoes a "Back to the Future" experience where tomorrow's silicon is programmed using yesterday's techniques—that is, explicit data placement, data movement, and data synchronization. Modern GPU architectures have a memory hierarchy that needs to be explicitly programmed to obtain good performance.

Today most people who make effective use of GPUs undergo a steep learning curve and are forced to program close to the machine using special GPU programming languages. Do programmers really need a special set of languages to write data-parallel programs for GPUs and a different set of languages to write data-parallel programs for multicore processors? At a high level of abstraction the task at hand is the same in both scenarios.

The software industry is reacting to the emergence of multicore systems by adding innovations to existing languages to help express concurrent and parallel computing. The use of data-parallel computing on GPUs, however, could not wait for languages such as C++ to be modified to support parallelism better, so systems such as CUDA were developed to fill the void.

As C++ is extended to support data parallelism (for example, through the introduction of data-parallel for loops), there will be new back ends for C++ compilers that can target GPUs (and eventually FPGAs), and some of the programs written today in CUDA will be written tomorrow in data-parallel versions of C++ and other mainstream languages such as Java and C#. To extract the best performance from GPUs, however, one will still have to use systems such as CUDA that expose low-level architectural features; but for many classes of algorithms and users, the higher-level compilation path from data-parallel conventional languages will be sufficient. Ultimately, we will need to devise new languages that provide effective memory models for modern computing systems.²

Figure 1. Compilation from a single description to multiple targets.



Language-neutral, multitarget data-parallel programming. To meet today's need for a higher-level data-parallel programming system that can target heterogeneous systems, one can look to the Accelerator system from Microsoft.¹³ It allows certain kinds of data-parallel descriptions to be written once and then executed on three different targets: GPUs, multicore processors using SSE3 vector instructions, and FPGA circuits, as illustrated in Figure 1. (Microsoft Accelerator can be downloaded from <http://research.microsoft.com/en-us/projects/Accelerator/>)

In general, we cannot hope to devise one language or system for programming heterogeneous systems that allows us to compile a single source into efficient implementations on wildly different computing elements such as CPUs, GPUs, and FPGAs. Such parallel-performance portability is difficult to achieve. If the problem domain is sufficiently constrained, however, it is possible to achieve good parallel performance from a single source description. Accelerator achieves this by constraining the data types used for parallel programming (to whole arrays that cannot be explicitly indexed) and by providing a restricted set of parallel array access operations (for example, in order, in reverse, with a stride, shifted, transposed).

Not all kinds of data-parallel algorithms are a suitable match (for example, matrix inversion does not fit this model). For many classes of algorithms (for example, stencil-style computations⁸), however, the Accelerator system makes it possible to compile a single description to three very different kinds of hardware and obtain good performance. One can always make a faster implementation using CUDA, SSE3 intrinsics, or Verilog, but this performance improvement comes at significantly higher design cost.

A key aspect of Accelerator is that it is not a new language. The system works just as a library that can be used from any language that supports linking with a C calling interface under Windows. Applications that use the Accelerator system have been written in several languages including C++, C#, F#, and the functional language Haskell.

Accelerator is an example of an embedded domain-specific language (EDSL). The Accelerator system provides the user with an abstract programming language that encodes notions of arrays and whole array operations and certain memory transforms. This abstract language is embedded in a conventional concrete language, and programs of the abstract language are expressed as data structures in the concrete language. This trick is a valuable technique for programming heterogeneous systems, allowing you to write data-parallel descriptions in a language-agnostic way and cross-compile to a variety of hardware architectures.

Consider the task of performing an element-wise data-parallel addition of two arrays: [1, 2, 3, 4, 5] and [6, 7, 8, 9, 10]. This very simple Accelerator program is shown in Figure 2 as a C++ program.

This regular C++ program can be compiled with the unmodified Visual Studio compiler and linked with the Accelerator library. When it executes it generates GPU code (using the DX9 system) at runtime on the host processor by JITing (using just-in-time compilation), transfers the generated code and data to the GPU for execution, and then transfers the results back to the host for display. The JITing model allows the system to hide many of the low-level details about the generation of GPU code and the communication with the GPU hardware. It is important to note that the data-parallel addition of the arrays *x* and *y* is specified by using an overloaded definition for the “+” operator that builds a data structure in the heap to express the intended computation:

FPA operator+ (FPA a1, FPA a2)

Figure 2. Accelerator C++ data-parallel element-wise addition program that uses a GPU.

```
#include <iostream>
#include "Accelerator.h"
#include "DX9Target.h"

using namespace std;
using namespace ParallelArrays;
using namespace MicrosoftTargets;

int main()
{
    // Create a GPGPU computing resource
    DX9Target *tgtDX9 = CreateDX9Target();

    // Declare some sample input arrays
    float xvalues[5] = {1, 2, 3, 4, 5};
    float yvalues[5] = {6, 7, 8, 9, 10};

    // Create data-parallel versions of inputs
    FPA x = FPA(xvalues, 5);
    FPA y = FPA(yvalues, 5);

    // Specify data-parallel computation
    FPA z = x + y; // Computation does not occur here...

    // Allocate space for the result array
    float* zvalues = (float*) malloc (5 * sizeof(float));

    // Execute the data-parallel computation on the GPU
    tgtDX9->ToArray(z, zvalues, 5); // z = x + y is now evaluated

    // Write out the result
    for (int i = 0; i < 5; i++)
        cout << zvalues[i] << " ";
    cout << endl;

    return 0;
}
```

When executed this program produces the output:

```
7 9 11 13 15
```

This code represents the input and output data-parallel arrays in “texture memories” and uses a data-parallel add instruction over these arrays. The user of the Accelerator system is entirely insulated from the graphics-level view of the computation.

To run the same computation as SSE3 vector instructions on a multi-core processor split across multiple processing cores, you would write exactly the same program but specify a different target. To emphasize the language-neutral aspect of Accelerator, the example in Figure 3 switches the source language to C# for the multicore SSE3 version.

When this program is run, it dynamically generates (JITs) SSE3 vector instructions and splits the data across multiple cores, then fires up vector instructions on each core to deliver exactly the same result as the GPU version.

Mapping explicit data-parallel computations onto FPGAs. When the performance requirements cannot be met by a multicore SSE3 implementation or

by a GPU implementation, you can try to produce an FPGA implementation by writing the same description of the parallel computation and specifying the use of an FPGA target. Again, to emphasize the language-neutral aspect of Accelerator the example in Figure 4 switches the source language to F#.

Note that this example does not write out the results of executing the data-parallel description on an FPGA chip. We cannot currently support a JITing model for the FPGA Accelerator target because the design tools that process hardware descriptions into FPGA programming bitstreams (the machine code of FPGA circuits) typically take a very long time to execute (tens of minutes or even hours). This target works in an offline mode and generates a VHDL (VHSIC hardware description language) hardware description that is deposited in a file (adder.vhd) that can be used as input to special design tools provided by FPGA vendors.

A more realistic example of an Accelerator program is a convolver that

computes a weighted average over a stream or image, as shown in Figure 5. A straightforward way to implement this algorithm is to represent the computation with nested for loops, which use explicit array indexing to capture the notion of a sliding window. However, this is a poor starting point for a system that wants to generate a parallel implementation or one that can be targeted to a heterogeneous collection of processing cores or digital hardware because it is difficult to map the arbitrary array indexing operations into efficient memory access operations on various targets.

A much better way to express this computation is in terms of a whole array operation combined with explicit memory-access transformations. An example of a memory transform operation is `shift`, which is shown in Figure 6. This operation takes an input array `x` and produces an output array that may be shifted right (`-1`) or left (`+1`).

Figure 7 shows an application of a shift operation used to describe successively shifted versions of the input array, and then whole array multiplication is performed in parallel followed by a parallel reduction (of add operations) to yield the convolved output.

In Accelerator the whole array description of the convolution operation can be expressed as shown in Figure 8.

This description is parameterized on the “target” (that is, it can be used to generate multithreaded SSE3 instructions, GPU code, or FPGA circuits, depending on the actual value of `computeTarget`). Note that the array of weights `a` is a regular C# array, because this represents compile-time constants rather than data that is only known at runtime. The other array parameter `x` has a different type: `FloatParallelArray`. This represents a data-parallel array that may live in a different address space from the host C# program (for example, on a memory bank of the GPU or on an embedded memory on an FPGA chip). These arrays do not permit explicit array indexing but instead offer memory-transform operations such as `Shift`.

To illustrate the kind of performance improvement that might be expected from such an approach the Accelerator version of a two-dimensional

Figure 3. Accelerator V# data-parallel element-wise addition using SSE3 vector instructions.

```
using System;
using Microsoft.ParallelArrays;
namespace AddArraysPointwiseMulticore
{
    class AddArraysPointwiseMulticore
    {
        static void Main(string[] args)
        {
            var x = new FloatParallelArray (new[] {1.0F, 2, 3, 4, 5});
            var y = new FloatParallelArray (new[] {6.0F, 7, 8, 9, 10});
            var multicoreTarget = new MulticoreTarget();
            var z = x + y;
            foreach (var i in multicoreTarget.ToArray1D (z))
                Console.WriteLine(i + " ");
        }
    }
}
```

Figure 4. Accelerator C# data-parallel element-wise addition using an FPGA circuit.

```
open System
open Microsoft.ParallelArrays
let main(args) =
    let x = new FloatParallelArray (Array.map float32 [|1; 2; 3; 4; 5|])
    let y = new FloatParallelArray (Array.map float32 [|6; 7; 8; 9; 10|])
    let z = x + y
    use fpgaTarget = new FPGATarget("adder")
    fpgaTarget.ToArray1D(z)
```

convolver computation is compared against a sequential version written as two separable passes.

The Accelerator version works by applying two one-dimensional convolutions and takes as a parameter the specific target to be used for the data-parallel execution of the algorithm in Figure 9.

The sequential version was run on a computer with two Intel Xeon E7450 processors, each clocked at 2.4GHz. The machine had 32GB of memory and ran Windows Server 2008 R2 Enterprise. The Accelerator version was run using the SSE3 multicore target on the same machine, testing the same code using two different graphics cards: an ATI Radeon HD5870 and an Nvidia 470 GTX using the DX9 target. The input matrix for the convolution was set to 8,000 by 8,000, and the experiment varied the size of the convolution kernel. The speedups achieved over the sequential baseline version are shown in Figure 10. The execution time for each of the Accelerator experiments includes the end-to-end time that measures the cost of JITing and data movement to and from the GPU or to and from thread-local state in the SSE3 case.

Figure 10 also shows that as the convolution kernel gets larger both the SSE3 multicore target and the GPU targets show improving performance. Cache effects, however, cause the speedup of the SSE3 multicore target to degrade after the kernel size grows to be larger than 40. This experiment shows the viability of using a single-source description that can be mapped to two very different targets to yield meaningful performance improvements.

The Importance of Specifying Communication

A key aspect of designing descriptions is the ability to describe memory-transform operations at a suitable level of abstraction. Descriptions that make too many assumptions about memory organization and access (for example, random array indexing) are difficult or impossible to port to other architectures that have wildly different memory-access economics.

Shift is a powerful memory-access abstraction because it reveals what in-

formation to reuse so it is worth caching, storing in a register, or placing into local shared memory on a GPU (see Figure 11). For example, when the FPGA target is used to implement the convolver, the shift operations are used to infer delays (and anti-delays) that allow you to reach each data item once but use it three times. (In an actual implementation extra delays are added to cancel the anti-delays, which are hard

to come by in the real world.) For a two-dimensional convolver this advantage is even more pronounced.

The delays introduced by the shift operator can be redistributed through the circuit to yield a transposed implementation of the convolver automatically (shown in Figure 12). This is a particularly good fit for FPGA technology because of the abundance of registers for implementing pipeline delays. A

Figure 5. One-dimensional convolution.

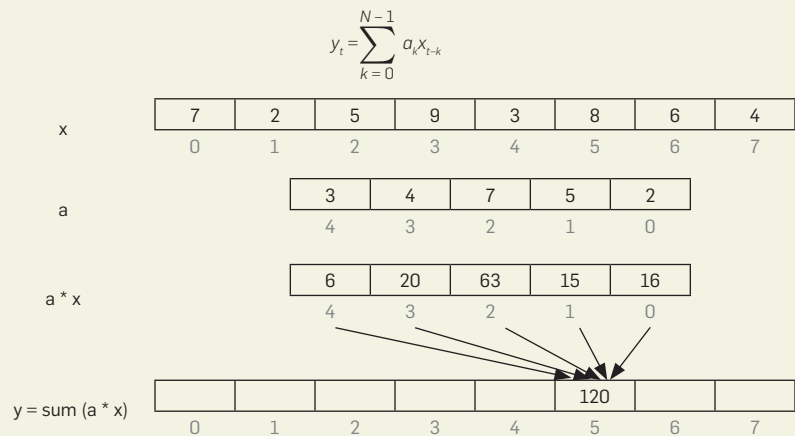


Figure 6. The shift memory-access transform.

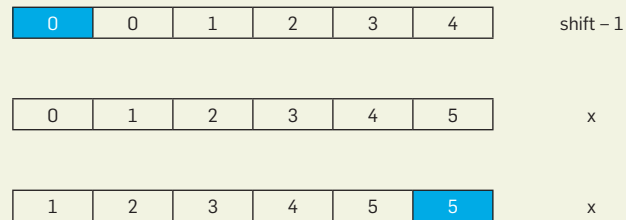
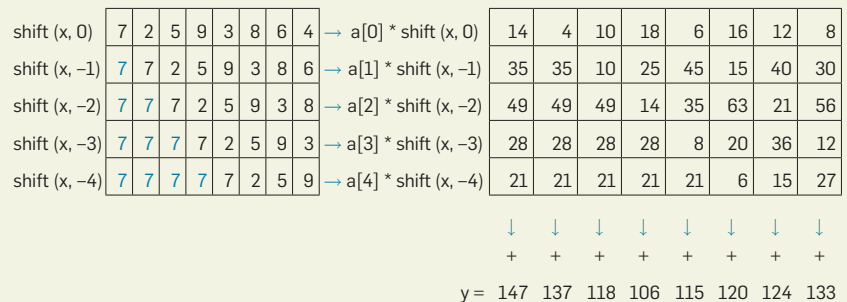


Figure 7. One-dimensional convolution expressed with shifts.



typical FPGA implementation of this circuit produces a system that has a delay of around five nanoseconds (for integer values), which allows 200 million elements to be convolved in a second. Many of these convolvers can be laid down on an FPGA circuit, and each can run in parallel.

Multiple targets. The Accelerator model is limited to mapping a single description completely onto one of several targets. A natural extension is to imagine a system that compiles a single data-parallel description onto a mixture of processing elements. For example, certain subexpressions could be mapped to GPU hardware and others could be mapped to FPGA

hardware, and the compilation process would also generate the software needed to orchestrate the movement of data between the different types of hardware elements and coordinate their execution. This capability is useful for producing a family of implementations for given functions that represent different price/performance points or different energy/latency trade-offs.

Mapping a single computation onto a heterogeneous set of cooperating processing elements can be taken one step further by allowing a computation to migrate from one type of processing hardware (say, a GPU) to another (say, an FPGA) in a transparent manner.

What We Might See in 18 Months

Turning programs into circuits has not been a glorious form of alchemy and has failed to produce what many developers want (a technique for genuinely converting real software into circuits). Instead it has provided digital designers with merely the ability to write highly stylized circuit descriptions that are a heartbeat away from low-level register transfer-level descriptions, which are usually cast in VHDL or Verilog. These techniques may be useful for improving the programming productivity of hardware engineers, but they do not form the basis of an approach that effectively converts realistic software into hardware circuits (or code for other kinds of heterogeneous computing elements).

Just as some alchemists eventually uncovered enough scientific and theoretical basis for their craft and then rebranded themselves as socially acceptable chemists, there are signs that a similar transmutation is occurring in the field of genuine high-level synthesis. Two very encouraging projects that seriously tackle the task of converting real programs into circuits are the Liquid Metal project at IBM, based on the Lime language that extends Java with special concurrency constructs,³ and the Kiwi project at the University of Cambridge by David Greaves and at Microsoft Research Cambridge (which works on unmodified standard .NET programming languages and relies on multithreaded programs to describe important parallel execution economics).⁶

Multicore processors and GPU architectures are beginning to move closer to each other. Intel's Sandy Bridge processors have AVX (Advanced Vector Extensions), which provide data-parallel operations over 256-bit registers. AMD's Fusion program promises ever tighter integration between CPUs and GPU cores on the same die. There is already a proof-of-concept single-package chip that combines an Intel Atom processor and an Altera FPGA. This could be the stepping-stone to systems that put FPGA-like logic on the same die as processor cores. The shift to a heterogeneous world looks inevitable.

Although the focus of this article has been heterogeneous *processing*, it is

Figure 8. Accelerator C# program for a one-dimensional convolution.

```
using Microsoft.ParallelArrays;
using A = Microsoft.ParallelArrays.ParallelArrays;
namespace AcceleratorSamples
{
    public class Convolver
    {
        public static float[] Convolver1D(Target computeTarget,
                                           float[] a, FloatParallelArray x)
        {
            var n = x.Length;
            var y = new FloatParallelArray(0.0f, new [] { n });
            for (int i = 0; i < a.Length; i++)
                y += a[i] * A.Shift(x, -i);
            float[] result = computeTarget.ToArray1D(y);
            return result;
        }
    }
}
```

Figure 9. Accelerator C# program for a two-dimensional separable convolution.

```
public static float[,] Convolver2D(Target computeTarget, float[] a, float[,] x)
{
    var xpar = new FloatParallelArray(x);
    var n = x.GetLength(0);
    var m = x.GetLength(1);
    var zpar = new FloatParallelArray(0.0f, new[] { n, m });
    var shiftBy = new[] { 0, 0 };
    for (var i = 0; i < a.Length; i++)
    {
        shiftBy[1] = -i;
        zpar += a[i] * A.Shift(xpar, shiftBy);
    }
    shiftBy[1] = 0;
    var ypar = new FloatParallelArray(0.0f, new[] { n, m });
    for (var i = 0; i < a.Length; i++)
    {
        shiftBy[0] = -i;
        ypar += a[i] * A.Shift(zpar, shiftBy);
    }
    return computeTarget.ToArray2D(ypar);
}
```

important to note that currently evolving heterogeneous systems consist of many other kinds of heterogeneous elements (for example, storage, memory, and networking). The management of different kinds of storage (for example, physical versus solid state), memory (for example, dealing with different types of memory with different access economics and latencies and reliability characteristics), and networking (for example, optical interconnects versus copper and wildly varying protocols) imposes formidable challenges to the design of systems software and application software for modern heterogeneous systems.

Further Out

There has also been progress in converting C programs that manipulate heap data structures with pointers into circuits using recent advances in shape analysis and region types. The conventional wisdom in the high-level synthesis field would lead one to believe that such transformations are impossible, but today you can deduce a surprising amount of information symbolically about what is happening in the heap of C programs and can determine with a far greater degree of fidelity which parts of the heap any given instruction may touch (compared with conventional point-to analysis).

These advances in the theory of program analysis arm us with the power tools needed to make the transmutation of programs into circuits (or GPU code) possible. These techniques allow a blurring of the distinction between hardware and software because they permit a computation cast as software working over a heap using pointers to be recast as a computation over an array (when certain bounds can be automatically computed) and then the code to be remapped to GPUs or FPGAs. This ability to convert an algorithm to preserve its meaning but refactor it to work over a different representation of its data that is more appropriate for a different execution environment is a key weapon that makes the retargetable programming of heterogeneous systems possible.

Researchers such as Luca Cardelli⁵ and Masami Hagiya⁷ are devising techniques for programming with DNA, pushing the frontiers of what it

is possible to compute with. Such research may shed light on how to build systems that can not only repair themselves but also assemble themselves. Although mainstream programmers are not likely to be programming with DNA anytime soon, such projects remind us that there are many things in the world that have the capability to compute. We face an exciting challenge in exploring how best to program them.

Conclusion

Programming until now has largely focused on cajoling CPUs into implementing our algorithms. This has involved significant assumptions about the economics of the underlying execution architecture and in particular on the architecture of the memory system. Increasingly, we will need to program computational engines that have radically different architectures from regular CPUs in order to meet our require-

Figure 10. Performance improvement from single source mapped to multiple targets.

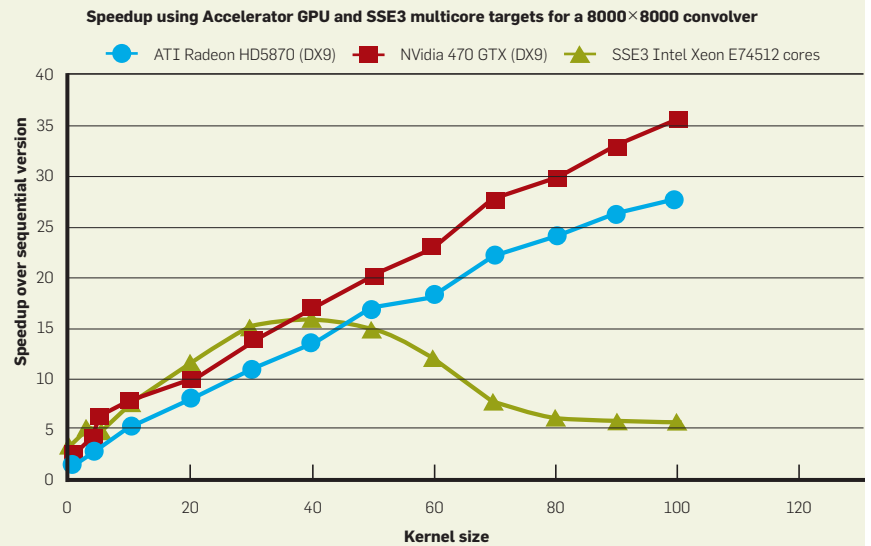


Figure 11. Shift: reusing rather than rereading data.

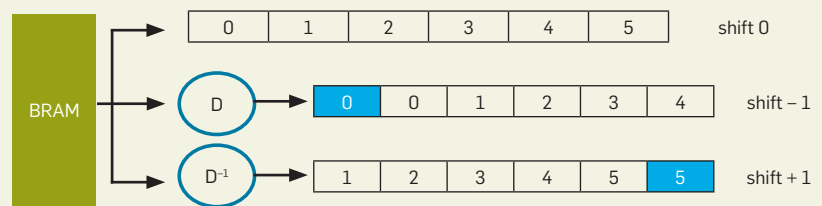
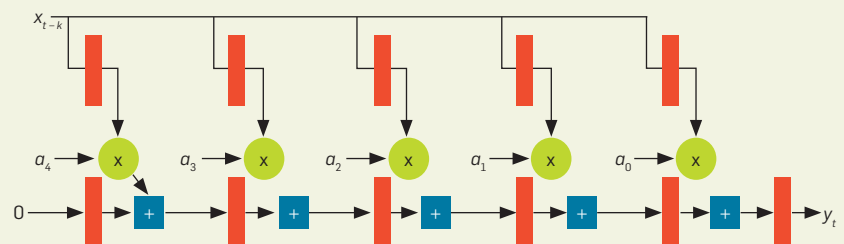


Figure 12. Automatically generated transposed 1D convolver.



ments for performance improvement or to reduce energy consumption.

Mapping the same computation onto several different computing elements typically involves rewriting algorithms to achieve an efficient execution on each architecture. This typically involves using a different programming language and a very different model of execution compared to the original algorithm that was developed for a CPU.

This is an expensive operation and although it will be necessary in many cases it is possible to identify a constrained class of problems that share some key architectural assumptions. This knowledge can then be exploited to permit an automatic compilation from one description onto multiple targets, for example, CPU vector instructions, GPUs, and FPGAs.

The Accelerator system is an example of this approach that provides an embedded domain-specific language for stencil-style computations that can be written in C, C++ or any language that supports a C calling interface, for example, C#, F#, or Haskell. In the case of Accelerator, the constraints are the use of data-parallel arrays that permit only whole array operations (no arbitrary array indexing) and the provision of explicit memory transform operations (shifting, rotation, strides). This avoids baking in assumptions about a particular memory model and permits the efficient mapping of Accelerator data-parallel computations onto very different execution targets.

As the number of heterogeneous processing elements grows we will need to develop a collection of such embedded domain-specific languages, each of which captures the essence of the computational requirements for a specific domain (stencil computations, XML parsing, regular expression matching). For each embedded domain-specific language we can then implement a mechanism for compiling descriptions onto a specific execution architecture.

By combining embedded domain specific languages that are used through library calls in mainstream programming languages we can take an important step toward developing mechanisms that make the pro-



The Accelerator system provides the user with an abstract programming language that encodes notions of arrays and whole array operations and certain memory transforms.



gramming of heterogeneous systems tractable while avoiding the cost of re-implementation or the cost of developing and adopting new languages. ■

Related articles on queue.acm.org

Blurring Lines Between Hardware and Software

Homayoun Shahri

<http://queue.acm.org/detail.cfm?id=644267>

The Cost of Virtualization

Ulrich Drepper

<http://queue.acm.org/detail.cfm?id=1348591>

Toward Energy-Efficient Computing

David J. Brown, Charles Reams

<http://queue.acm.org/detail.cfm?id=1730791>

References

1. Abadi, M., et al. Semantics of transactional memory and automatic mutual exclusion. *ACM Transactions on Programming Languages and Systems* 33, 1 (2011).
2. Adve, S.V. and Boehm, H.-J. Memory models: A case for rethinking parallel languages and hardware. *Commun. ACM* 53, 8 (Aug. 2010).
3. Auerbach, J., et al. Lime: A Java-compatible and synthesizable language for heterogeneous architectures. In *Proceedings of the ACM International Conference on Object-oriented Programming Systems Languages and Applications (OOPSLA 2010)*.
4. Borkar, S. and Chien, A.A. The future of microprocessors. *Commun. ACM* 54, 5 (May 2011).
5. Cardelli, L. Strand algebras for DNA computing. *Natural Computing* 10, 1 (2009), 407.
6. Greaves, D. and Singh, S. Designing application-specific circuits with concurrent C# programs. In *Proceedings of the 8th ACM/IEEE International Conference on Formal Methods and Models for Codesign (2010)*.
7. Hagiya, M. Towards autonomous molecular computers. In *Proceedings of 3rd Annual Genetic Programming Conference (1998)*.
8. Lesniak, M. PASTHA—Parallelizing stencil calculations in Haskell. In *Proceedings of the 5th ACM SIGPLAN Workshop on Declarative Aspects of Multicore Programming (2010)*.
9. Madhavapeddy, A. and Singh, S. Reconfigurable data processing for clouds. MSDN Blogs (2011); http://blogs.msdn.com/b/satnam_singh/archive/2011/01/18/reconfigurable-data-processing-for-clouds.aspx.
10. Nvidia Corporation. Nvidia CUDA (2007); <http://developer.nvidia.com/cuda>.
11. Owens, J.D., et al. GPU computing. In *Proceedings of the IEEE* 96, 5 (2008).
12. Shavit, N. Data structures in the multicore age. *Commun. ACM* 54, 3 (Mar. 2011).
13. Tarditi, D., Puri, S. and Oglesby, J. Accelerator: Using data parallelism to program GPUs for general-purpose uses. *ASPLOS* (2006).
14. Trancoso, P., Othonos, D. and Artemiou, A. Data-parallel acceleration of decision support queries using Cell/BE and GPUs. In *Proceedings of the 2009 ACM International Conference on Computing Frontiers*.
15. Xu, L. and Wan, J.W. Real-time intensity-based rigid 2D-3D medical image registration using rapidmind multi-core development platform. In *Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (2008)*.

Satnam Singh (s.singh@acm.org) is a researcher at the Microsoft Research Cambridge UK laboratory, where he investigates high-level design techniques for low-level systems. He is also chair of Reconfigurable Computing at the University of Birmingham.



It's no easy task for NoSQL.

BY OREN EINI

The Pain of Implementing LINQ Providers

I REMEMBER SITTING on the edge of my seat watching videos from the 2005 Professional Developers Conference that first showed Language Integrated Query (LINQ). I wanted LINQ: it offered just about everything I could hope for to make working with data easy.

The impetus for building queries into the language is quite simple; it is something that is used all the

time; and the promise of a unified querying model is good enough, even before you add all the language goodies that were dropped on us. Being able to write in C# and have the database magically understand what I am doing? Awesome! Getting compilation errors from Visual Studio, rather than runtime errors at testing (or worse, production)? Wonderful! Getting rid of most SQL injection issues? Amazing!

Then .NET 3.5 rolled around, and we had LINQ, and I found myself in quite a pickle. There are actually two sides to the LINQ story: the consumer side and the implementer side. For

the consumer, it is all sunshine and flowers, but for the implementers it is doom and gloom with just a tad of despair. And even for the consumers, past the initial love-at-first-sight phase, they are issues that require handling. Let me focus first on the consumer side.

Despite appearances, LINQ isn't actually SQL in C# (or VB.NET). It is just a syntax transformation from a set of well-known keywords to well-known methods. The usage of `Select`, `Where`, and `GroupBy` has made it taste very similar to SQL, but there are drastic differences between the

way LINQ works conceptually and the set-based relational logic that rules SQL databases.

Let us examine a simple LINQ query:

```
from user in users
where user.IsActive
select user;
```

We do not know the type of the users variable, but in general it would be either an `IEnumerable<T>` or an `IQueryable<T>`. `IEnumerable<T>` is the basic interface for everything in memory enumeration and iteration. But `IQueryable<T>`, along with the syntax transformation done by the compiler, is something else altogether. It allows implementers of `IQueryable<T>` to access the expression tree (the compiler abstract syntax

tree) for the query.

Let us look at how this query actually translates. Table 1 shows the transformation depending on the type of the users variable.

Because `IQueryable<T>` implementers are called with the expression tree of the query, they have the chance to decide, based on that expression tree, how actually to execute that query. In many cases, the `IQueryable<T>` implementation will translate the expression tree into the querying medium of the underlying data store (for example, SQL and XPath) and not execute the code directly.

The LINQ to Objects (querying against an `IEnumerable<T>`) behavior is considered to be the definitive implementation for how a LINQ query should behave. That implementation

is actually executing the code in memory, and as such, has access to the entire Base Class Library and can execute any operation it wishes. That is not the case with other LINQ implementations, since it is the `IQueryable<T>` implementation that determines which operations are supported and which are not. The problem is, however, that beyond the most trivial of queries, it is actually quite hard to map the intent of the user to a query that makes sense on the target data store. Operations that make perfect sense when all of the data is accessible in main memory make poor choices when the data is distributed in different database tables or even different servers.

For example, consider the following nontrivial LINQ query, which finds all the incomplete late tasks for the active users on projects that they own:

```
from user in users
where user.IsActive
from task in user.Tasks
where
    task.Project.Owners.Contains(user) &&
    task.DueDate < DateTime.Now &&
    task.Completed == false
select new { task.Title,
            task.Status, user.UserName };
```

This is a very clean way of expressing the intent, and it reads quite nicely. When running on a small data set in memory, it also performs beautifully. The problem is, for the most part, we are *not* running on small data sets or in memory.

Consider for a moment the effect of such a query on a large data set, even in memory. With LINQ to Objects, most primitive operations have a $O(N)$ cost. In the previous query, we have composite calls that feed on one another.

What this leads to is that even when we are talking about pure in-memory models, large data sets require us to take optimized approaches. We could build an `IQueryable<T>` implementation that would use in-memory indexes for performing this query. For the most part, however, we are not writing such queries against an in-memory representation; we are writing such queries against a database. A database can handle large data sets easily; that is what it is *for*.

Table 1. Compiler transformation for `IEnumerable<T>` queries and `IQueryable<T>` queries.

Our code	Compiler transformation result
<pre>IEnumerable<User> users = null; var activeUsers = from user in users where user.IsActive select user;</pre>	<pre>IEnumerable<User> users = null; var activeUsers = users.Where(delegate(User user) { return user.IsActive; });</pre>
<pre>IQueryable<User> users = null; var activeUsers = from user in users where user.IsActive select user;</pre>	<pre>IQueryable<User> users = null; var isActiveProperty = Expression.Parameter(typeof (User), "user"); var activeUsers = users.Where(Expression.Lambda<Func<User, bool>>(Expression.Property(isActiveProperty, typeof (User).GetProperty("IsActive"), new[] { isActiveProperty })));</pre>

Figure 1. A query example translated into SQL using the LINQ to Entities framework.

```
SELECT [Extent1].[Id] AS [Id],
       [Filter1].[Title] AS [Title],
       [Filter1].[Status] AS [Status],
       [Extent1].[UserName] AS [UserName]
FROM   [dbo].[Users] AS [Extent1]
       INNER JOIN (SELECT [Extent2].[UserId] AS [UserId],
                          [Extent3].[Title] AS [Title],
                          [Extent3].[Status] AS [Status],
                          [Extent3].[ProjectId] AS [ProjectId]
FROM         [dbo].[UsersTasks] AS [Extent2]
              INNER JOIN [dbo].[Tasks] AS [Extent3]
ON [Extent3].[Id] = [Extent2].[TaskId]
WHERE      ([Extent3].[DueDate] < CAST(sysdatetime() AS DATETIME2))
AND (0 = [Extent3].[Completed]) ) AS [Filter1]
ON [Extent1].[Id] = [Filter1].[UserId]
WHERE ( [Extent1].[IsActive] = 1 )
AND ( EXISTS (SELECT 1 AS [C1]
FROM         [dbo].[ProjectOwners] AS [Extent4]
WHERE      ([Filter1].[ProjectId] = [Extent4].[ProjectId])
AND ([Extent4].[UserId] = [Extent1].[Id]) ) )
```

Using the LINQ to Entities framework, Figure 1 depicts the previous query example translated into SQL.

So far, so good, but it is very easy to write LINQ queries that no provider can translate, simply because there is no mapping between whatever it is you want to do in the query and the operations provided by the underlying database. For example, consider something as simple as:

```
from user in users
where user.UserName[5] == 'A'
select user;
```

With LINQ to Objects, this query runs just as you would expect, but trying this query using LINQ to Entities would throw an error. That error would happen only at runtime, because the code is perfectly fine, and the compiler has no way of knowing if a specific `IQueryable<T>` implementation supports this. Or, what is actually worse, the provider will be able to translate it, but into a query that is going to perform poorly.

There is no technical reason why LINQ to Entities cannot support queries on string indices, but consider what it would mean from the database point of view. That would be translated into something like `SUBSTRING(Username, 5,1) = 'A'`, and that would mean forcing a table scan to answer this query.

The problem, as always, is that when you have a layer of abstraction, it can easily abstract away important details. The result is that while in theory you could move your queries from one data store to another, in practice you have to be well aware of the actual behavior of the LINQ provider and the underlying database.

It isn't that LINQ generates inefficient queries, or even that it allows the generation of inefficient queries. This is simply an issue of abstraction hiding what is really happening. LINQ gives a lot of latitude to the users, and it is easy to create queries that simply cannot be executed efficiently by the underlying data store. That is mostly because there is a big gap between the semantics of querying in memory collections (which is the abstraction that LINQ uses) and querying data stores such as relational and nonrelational databases.

As a consumer of LINQ, I absolutely adore it, despite some of the issues just discussed. But *consuming* LINQ is the part that is all sunshine and roses. The task of *implementing* a LINQ provider is one of Herculean proportion and involves much use of fertilizer, sweat, and hard work.

Implementing can be Painful

Since .NET 3.5 came out, I've had to implement a LINQ provider several times. I implemented the first one for NHibernate, and I implemented a few that did not target a relational database, the most important of which is the LINQ provider for RavenDB (<http://ravendb.net>).

ravendb.net).

In each and every case, the process was extremely painful. To build a LINQ provider, you first have to understand how the compiler looks at the language, because that is all a LINQ provider is. You are given an expression tree, which is just the compiler view of a piece of code, and are told: "Make something of that and give me a result of T."

The problem is, there is very little relation between what the compiler hands to you and what the user actually wrote. The compiler view of the code is often drastically different than what you would expect.

Figure 2. The expression tree from a simple query.

```
1. MethodCallExpression
  a.Method : MethodInfo : "Where"
  b.Arguments : ReadOnlyCollection
    i. ConstantExpression
      1. Value : Object : "LINQConsoleApplication1.User[]"
      2. NodeType : ExpressionType : "Constant"
      3. Type : Type : "EnumerableQuery"
    ii. UnaryExpression
      1. Operand : ExpressionLambda
        a.Expression>
          i.Body : ExpressionEqual
            1.BinaryExpression
              a. Left : ExpressionMemberAccess
                i. MemberExpression
                  1. Expression : ExpressionParameter
                    a. ParameterExpression
                      i. Name : String : "user"
                      ii. NodeType : ExpressionType : "Parameter"
                      iii. Type : Type : "User"
                  2. Member : MemberInfo : "Int32 Id"
                  3. NodeType : ExpressionType : "MemberAccess"
                  4. Type : Type : "Int32"
                    a. Right : ExpressionConstant
                      i. ConstantExpression
                        6. Value : Object : "1"
                        7. NodeType : ExpressionType : "Constant"
                        8. Type : Type : "Int32"
                  ii. Parameters : ReadOnlyCollection
                    1.ParameterExpression
                      a. Name : String : "user"
                      b. NodeType : ExpressionType : "Parameter"
                      c. Type : Type : "User"
                  iii. NodeType : ExpressionType : "Lambda"
                  iv. Type : Type : "Func"
                2. Method : MethodInfo : null
                3. IsLifted : Boolean : "False"
                4. IsLiftedToNull : Boolean : "False"
                5. NodeType : ExpressionType : "Quote"
                6. Type : Type : "Expression>"
              b.NodeType : ExpressionType : "Call"
              c.Type : Type : "IQueryable"
```

Consider this simple LINQ query:

```
from user in users
where user.Id == 1
select user;
```

Though this is probably the very simplest LINQ query possible, the compiler translates it to something that looks drastically different and quite complex, as shown in Figure 2. This is one of the more problematic aspects of writing a LINQ provider. You get something that does not look very much like what the user provided.

If you are used to writing compilers, this all makes perfect sense. When you have more complex queries, however, even queries that do not seem much different on the surface, the result is drastically different and far more complex. The following query produces an expression tree that goes on for five pages:

```
from user in users
where user.IsActive
from task in user.Tasks
where
    task.Project.Owners.Contains(user) &&
    task.DueDate < DateTime.Now &&
    task.Completed == false
select new { task.Title,
            task.Status, user.UserName };
```

The first query is three lines and has an expression tree that goes on for about a page. The second is five lines, and has an expression that

goes on for five pages. You can see the full expression trees for both queries in the files, http://queue.acm.org/downloads/2011/Trivial_Query.html and http://queue.acm.org/downloads/2011/Non_Trivial_Query.html, respectively. For compiler writers, this just makes sense. For developers who aren't well versed in compiler theory, this is a big stumbling block.

Most mature LINQ providers actually use a two-stage process. In the first part of the process the provider runs over the expression tree, compiling its own in-memory representation of the query, as extracted from the expression tree. The second part is performing whatever it is that the user has actually wanted to do.

While it might not surprise you, almost all of the hard work involves translating the expression tree the compiler gave us into something that actually closely resembles what the user initially provided.

The start of a better approach can be seen in the re-linq framework (<http://relinq.codeplex.com>), which takes care of most of the work of translating from the internal compiler representation and the user intent. Personally, I would have constructed an AST (abstract syntax tree) mimicking the actual query semantics from the get-go instead of providing `IQueryable<T>` with the raw compiler output. Figure 3 is an example of what the trivial query AST looks like when using the NRefactory project (<http://wiki.sharpdevelop.net/NRefactory.ashx>) to process it.

Working with something like this is drastically simpler than working with the compiler output, because we now have the relevant meaning. As it stands, we have to extract the meaning ourselves, which is not trivial.

There are two lessons to take from this. The first is that giving developers the raw compiler output as an expression tree is a fairly poor choice from the point of view of the developers needing to build LINQ providers on top of those. Even fairly minimal work (such as seen in Figure 3) would have made creating LINQ providers drastically easier. It isn't that I think that you could make it work for all cases; I believe that you would still have this complexity in some cases, if only because we need to support arbitrarily complex inputs.

The vast majority of cases, however, falls into fairly predictable patterns, and solving this problem at the compiler/framework level would have made the cases of complex expression trees rare rather than everyday occurrences. That is why I strongly encourage the use of third-party libraries, such as the re-linq project, that take on a lot of the work of putting the common transformations into a more reasonable format.

There are other issues in implementing a proper LINQ provider:

- Different languages provide different expression trees for the same logical code. For example, equality is handled differently in C# and VB, depending on whether you are comparing to a string or not.

- Different versions of the compiler can emit different expression trees for the same queries.

- There is absolutely no documentation about expression trees that helps to build a LINQ provider.

- Your input is basically any legal expression in C# or VB.Net, which means that the option space you have to deal with is huge. You can throw an exception if you don't understand the query, of course, but that means that users will discover unsupported scenarios only at runtime, defeating much of the point of having queries integrated into the language.

To make matters worse, users often come up with the most complex que-

Figure 3. Trivial query AST, as generated by NRefactory.

```
• FromClause=QueryExpressionFromClause
  o Identifier=user
  o InExpression
    • IdentifierExpression Identifier=users
  o MiddleClauses
    • QueryExpressionWhereClause Condition
      • BinaryOperatorExpression
        • Left
          • MemberReferenceExpression
            • TargetObject Identifier=user
            • MemberName=Id
        • Op=Equality
        • Right
          • PrimitiveExpression
            • Value=1
    o SelectOrGroupClause
      • QueryExpressionSelectClause Projection
        • IdentifierExpression Identifier=user
```

ries and expect them to work. Because of the reasons outlined above, implementing a LINQ provider is mostly a matter of implementing it one query at a time. The reason that you have to approach building a LINQ provider in this way is that you cannot really treat many operations as black boxes where given X output Y . For example, consider the case of the `Count()` operation. It can be applied to the query globally, to subqueries in the query, to a collection property as part of the projection or processing the query, or inside the group by clause. Each case needs to be handled differently. Furthermore, the user usually comes up with queries that may require re-architecting the way you are doing things.

LINQ to NoSQL

So far, I have talked mostly about the problems inherent in implementing any LINQ provider, but this article is particularly focused on building a LINQ provider for a NoSQL data store.

One of the problems with building such a LINQ provider is that quite a bit of the LINQ API makes absolutely no sense outside of a relational database. For example, how do you handle grouping in a document database? Or ordering in a key/value store? You can certainly do either one (maybe not very efficiently, but you can). The LINQ API makes it seem as though it is just as easy to handle as if it were all in memory. There are ways to mark off parts of the API as unusable for a particular provider, but it is an opt-out approach and requires quite a bit of repetitive coding.

Now, given the intent in building LINQ (allow querying over anything), there really isn't a good way *not* to have this problem. Different data stores have different semantics, and it is fairly difficult to consider all of those semantics in a generic API. There are ways provided to do so, and that is sufficient, if annoying.

Before we get to marking parts of the LINQ API as outside the scope of our providers, we still have to get to the first order of business: How do you handle *querying*?

As I mentioned, I had the chance to build a few LINQ providers. Among those, RavenDB and RavenLight are two recent implementations that pro-

vide two very similar yet drastically different approaches to implementing a LINQ provider.

The RavenLight LINQ Provider

RavenLight is an embedded document database meant to run inside Silverlight applications. In other words, it is running completely on the client side with no server-side component whatsoever (it can replicate to and from a RavenDB instance, but that is beside the point for this article). Figure 4 shows a typical Silverlight application using RavenLight.

When building RavenLight's LINQ provider, we made the following assumptions:

- ▶ All the data is available locally.
- ▶ The total amount of the data in the data store is likely to be small (up to tens of thousands of items, not millions or more).
- ▶ Most Silverlight queries will be done in response to a user action.

I'll add a tautological statement as well: the broader the support for the LINQ API, the better it is.

With RavenLight, we actually decided to skip the bother of implementing a LINQ provider and just use LINQ to Objects.

How does this work? Because all the data is available locally and the amount of data stored is small, we are going to run queries directly against the data store. What this means in

practice is that we can do something like what is shown in Figure 5.

As you can see, we are not actually implementing anything here. The `ReadAll<T>()` method simply does a first-level filtering based on the requested type, and that is it. I simplified the implementation drastically to make the point, obviously, but this is how it works at a conceptual level. Presto, the entire LINQ API is supported, because we are actually returning `IEnumerable<T>` under the covers.

This approach has some limitations, but we left the option open to sit down at a later date and implement a LINQ provider that does not perform a linear scan operation all the time. That said, when N is small, $O(N)$ approach is quite reasonable and efficient.

For that matter, if we want to get a

Figure 4. A typical application using RavenLight.

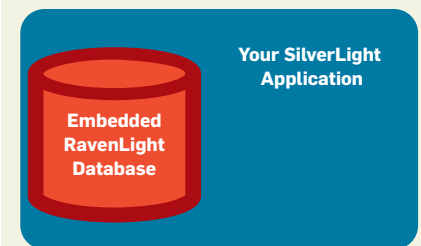


Figure 5. Simplified example of RavenLight LINQ provider code.

```
public class RavenLight
{
    // other stuff

    public IQueryable<T> Query<T>()
    {
        return ReadAll<T>().AsQueryable();
    }

    private IEnumerable<T> ReadAll<T>()
    {
        var fullName = typeof(T).FullName;
        foreach (var item in datastore["ByType"]
            .SkipTo(fullName)
            .TakeWhile(x => x.Type == fullName))
        {
            yield return item.Deserialize<T>();
        }
    }
}
```

quick speed boost we can probably do this:

```
public IQueryable<T> Query<T>()
{
    return ReadAll<T>().AsQueryable().AsParallel();
}
```

This option is efficient in terms of both developer time and actual usage. Remember, in a Silverlight application, we are literally running at the client desktop. We do not have to worry much about performance beyond ensuring that we can respond to the query in a time shorter than the human response time. That gives us *tens of milliseconds* at least, and that amount of time means that we can “waste” a lot of computing power.

This approach, however, is suitable only if you are able to meet all of the requirements I have specified. For the most part, I think you will find that is not the case.

Far more often, I see people turning to one of two options: LINQ serialization and full-blown LINQ providers. Serializing LINQ queries over the wire assumes you have something on the other hand that can process them efficiently. Usually, this isn’t the case, and even if it is, real-world data set sizes would make this a very inefficient choice.

Querying NoSQL Data Stores

A relational database allows you to make just about any query that you want. A NoSQL data store will generally put limits on your querying capability. For example, you may be able to query only on specific fields (predetermined) or in a specific way.

With a NoSQL solution such as key/value store (Memcached, Redis, among others), you can query only by ID. With a BigTable data store, your querying options are limited to querying by key or key range and so on.

In many cases, it literally does not

pay to try to provide a LINQ provider abstraction on top of highly limited querying capabilities. If we are using something like Redis (key/value store, only querying option is by ID), there really is not much point in trying to create a LINQ provider. It would serve only to create the illusion that we can support additional query operations.

With RavenDB, the querying capability allows us to perform most queries (property equal to value, range queries on properties, and so on) without much trouble, which means it makes a lot of sense to support a LINQ provider. That said, there are actually many operations we cannot support. For example, a GroupBy operation is not something that can usually be defined on the fly.

Building the LINQ provider for RavenDB was not an easy task; in fact, we budgeted more time to it than to building the actual database. Yes, it took less time to build the database and the querying mechanism than to build the LINQ provider for that same database.

RavenDB allows you to issue queries such as those shown in Table 2.

As you can imagine, the task of the LINQ provider is to take the LINQ query expressions and emit the RavenDB syntax. That is more or less what we have done. Building a trivial LINQ query provider is difficult but not too difficult.

Processing a single expression is actually quite easy once you have the framework in place to support it. It gets somewhat more difficult with compounded expressions, but it is still manageable.

The problem really begins to hit when you start considering how to (or even if you should) support operations that are not really provided by the underlying data store. For example, joining and grouping are two operations that RavenDB does not do, but they are native parts of LINQ (and are actually part of the language). There are simpler examples. One of the features of

RavenDB is it performs absolutely no computation during queries; all the data for the query is already prepared. That means we can achieve very good querying speeds.

The issue is the user can specify a statement such as:

```
from user in users
where user.Id % 2 == 0
select user.Name;
```

Again, this is mostly an issue of determining how much you are going to support. Obviously, we can’t send that query to RavenDB, which has no built-in support for math operation during queries. You can add that support to the client portion of the API, of course, but that means your provider implementation has to become more and more sophisticated, and it already took more time than just building the database.

Because of those issues, we have decided we are not going to try to support all of the options available in the LINQ API. Instead, we are going to define a common set of operations that we will support, and anything outside that is not going to work.

For example, consider a query for a user in a particular role:

```
var q = from user in users
from role in user.Roles
where role == "Admins"
select user;
```

We can’t really support this query; it is too complex for us to understand easily (and it opens up quite a lot of possibilities that we would have to support). Figuring out that role came from user is actually pretty complex, because of the way this is handled internally. You have to keep track of aliases, transparent identifiers, re-names, handle multiple nested statements, and so on.

We still wanted to support this, however, so we provided an alternative way of doing that:

```
var q = from user in users
where user.Roles.
Any(role => role == "Admin")
select user;
```

The reason this is so much easier

Table 2. RavenDB LINQ provider results.	
LINQ Query	RavenDB Syntax
where user.Name == "Ayende"	Name: Ayende
where user.Age > 18	Age: [18 TO NULL]

than the previous query is simple. We have all of the information for processing the expression directly in the node. In the previous query, we would have to spend quite a bit of effort just figuring out from where we got `role`. What we would get is actually something called transparent identifier, and we would have to track down from where it originated.

From the user perspective, there is not much of a difference, but there is a huge difference in the amount of complexity involved in understanding this statement compared with the previous one.

We have made similar decisions in other places, and we have thoroughly documented them. Users get to use LINQ, with all the benefits associated with it (strong typing and intelligence among the most important), but we do not have to tackle the entire complexity involved in trying to support anything that can be written as a LINQ query.

Summary

As mentioned, it took more time to build the LINQ provider for RavenDB than it took to build RavenDB itself. Indeed, by far the most complicated piece of code in RavenDB even now is the LINQ provider implementation.

Nevertheless, when talking about the .NET framework, if you are creating a database or a library to access a database, you pretty much have no choice but to provide a LINQ implementation. Even after saying that, I would recommend evaluating carefully whether you really need a LINQ provider or, to be more exact, whether a LINQ provider implementation would be worthwhile.

If you are connecting to a key/value store or something else that has limited querying options, you probably shouldn't bother. Having a LINQ query available will only hint at things you can do that you actually cannot. It is easier not to have one than to explain why this or that operation is not supported.

If you do decide to implement a LINQ provider against a NoSQL database, I would strongly encourage you to choose a conventional way to define each supported operation. You will be much better off if your users' input can be made to follow specific patterns.

If you do decide to implement a LINQ provider against a NoSQL database, I would strongly encourage you to choose a conventional way to define each supported operation.

Another option is to selectively disable parts of the LINQ API for your LINQ provider (by creating your own LINQ methods and marking them with the [Obsolete] attribute).

Finally, there are libraries such as `re-linq` (<http://relinq.codeplex.com/>) that can help build LINQ providers. `Re-linq` in particular takes care of doing most of the work of going from a LINQ expression tree to something far more palatable to work with.

As I said in the beginning of this article, I really like LINQ, but only from the consumer side. Writing LINQ providers is not a task for the faint of heart and should not be attempted unless you know what you are doing.

You might be better off with a limited LINQ implementation instead. Consider creating a provider that supports very few operations, maybe even one that supports only equality tests. That can be quite impressive, and it is not that hard to do.

It is when you want to support more that the complexity starts to pile up, so think carefully about precisely what you want to do, the costs associated with each option, and the benefits that will be gained.

Users *love* LINQ, after all... 

Related articles on queue.acm.org

A Co-Relational Model of Data for Large Shared Data Banks

Erik Meijer, Gavin Bierman

<http://queue.acm.org/detail.cfm?id=1961297>

A Conversation with Erik Meijer and Jose Blakeley

<http://queue.acm.org/detail.cfm?id=1394137>

The Next Big Thing

George Neville-Neil

<http://queue.acm.org/detail.cfm?id=1317398>

Oren Eini has more than 15 years of experience in the development world, concentrating on the Microsoft and .NET ecosystem. He is the founder of Hibernating Rhinos LTD, an Israeli-based company offering friction-free data-management solutions. Under his pseudonym, Ayende Rahien, Eini is a frequent blogger at <http://www.ayende.com/Blog/>

DOI:10.1145/1978542.1978559

Unite neuroscience, supercomputing, and nanotechnology to discover, demonstrate, and deliver the brain's core algorithms.

BY DHARMENDRA S. MODHA, RAJAGOPAL ANANTHANARAYANAN, STEVEN K. ESSER, ANTHONY NDIRANGO, ANTHONY J. SHERBONDY, AND RAGHAVENDRA SINGH

Cognitive Computing

WHAT IS THE mind? Neither scientists nor philosophers agree on a universal definition or specification. Colloquially, we understand the mind as a collection of processes of sensation, perception, action, emotion, and cognition. The mind can integrate ambiguous information from sight, hearing, touch, taste, and smell; it can form spatiotemporal associations and abstract concepts; it can make decisions and initiate sophisticated coordinated actions.

Cognitive computing aims to develop a coherent, unified, universal mechanism inspired by the mind's capabilities. Rather than assemble a collection of piecemeal solutions, whereby different cognitive processes are each constructed via independent solutions, we seek to implement a unified computational theory of the mind. AI pioneer Allen Newell described it as "a single set of mechanisms for all of cognitive behavior. Our ultimate goal is a unified theory of human cognition."

Historically, many disparate fields have taken radically different approaches to exploring mind-like computation, some of which we cover here. On the one

hand, strong artificial general intelligence, or AI,¹⁰ a branch of cognitive science, takes a system-level approach to synthesizing mind-like computers. Since the mind arises from the wetware of the brain, neuroscience¹⁸ takes a component-level approach to understanding how it gives rise to the mind. Proceeding top-down in a reductionist fashion, cognitive neuroscience⁹ seeks to integrate theoretical cognitive science with experimental psychology and organism-level neuroscience. In contrast, proceeding bottom-up in a constructive fashion, systems neuroscience¹⁸ seeks to combine experimental data at multiple spatial and temporal scales. The diversity of thought implicit in this plurality of approaches is essential, given the profound technological importance and scientific difficulty of the mind-brain problem. Science thrives on multiple groups taking different, complementary, parallel perspectives while working at different levels of abstractions.

Against this backdrop, our novel, promising approach is to operationalize vast collections of neuroscience data by leveraging large-scale computer simulations. Today, a thoughtful selection from the riches of neurophysiology and neuroanatomy can be combined to produce near real-time

» key insights

- **Cognitive computing will lead to novel learning systems, non-von Neumann computing architectures, programming paradigms, and applications that integrate, analyze, and act on vast amounts of data from many sources at once.**
- **A new white-matter long-distance network spanning the entire Macaque monkey brain consisting of 383 regions and 6,602 connections opens fresh ways to analyze, understand, and eventually, imitate the network architecture of the brain.**
- **Path-breaking developments in cortical simulation algorithms enable cat-scale cortical simulations on Lawrence Livermore National Laboratory's Dawn Blue Gene/P supercomputer with 147,456 CPUs and 144TB of main memory.**

Measuring White-Matter Pathways in the Human Brain

The white matter of the human brain comprises more than 150 kilometers of long-range projections. Understanding the architecture of these projections (the “projectome”) is important for understanding brain function and has led to fundamental discoveries in normal and pathological brains (see Figure 1). Despite these findings, the bulk of the human projectome, or complete map of axonal projections in the brain, remains unexplored, with many exciting questions to answer.

Recent advances in diffusion-weighted magnetic resonance imaging (DW-MRI) have allowed noninvasive measurement of the human white-matter network across the entire brain. DW-MRI acquires an aggregate description of

the microscopic diffusion of water molecules, along many directions within millimeter-size chunks of brain tissue. The dense packing of axon bundles within the white matter imposes oriented obstacles faced by water molecules. By measuring diffusion patterns produced by these obstructions, DW-MRI can help determine the location and orientation of axon bundles.

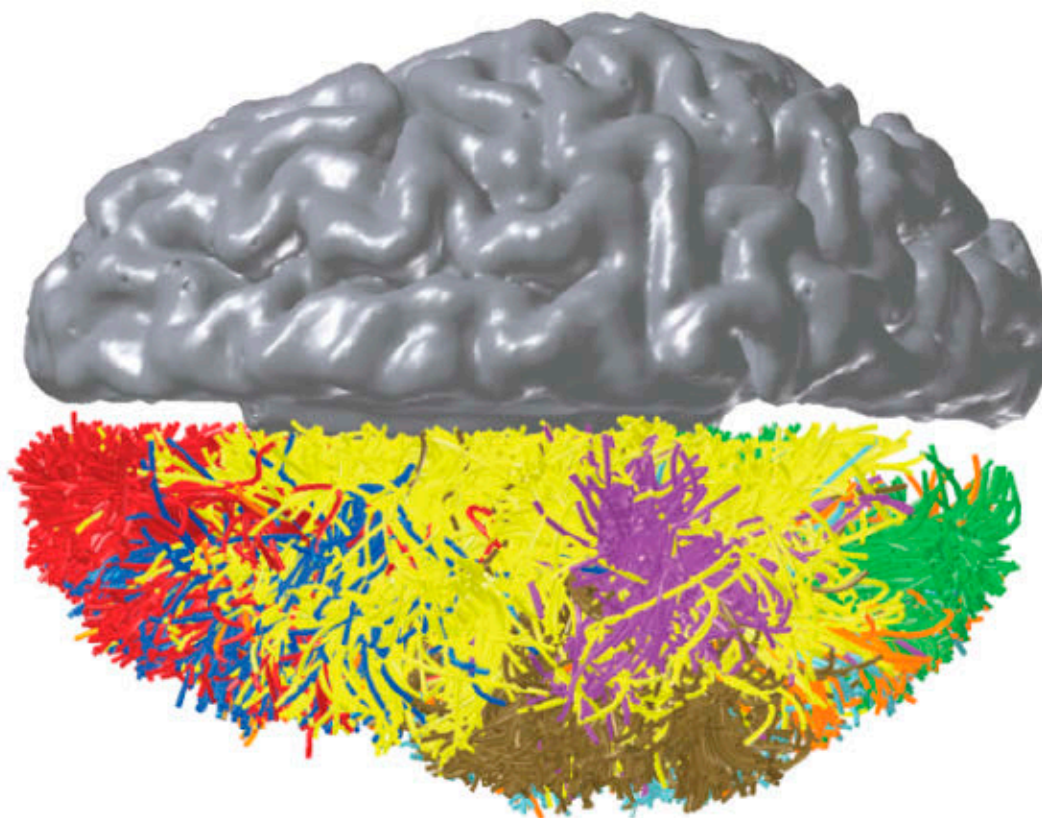
Unfortunately, the aggregation of the microscopic diffusion measurements at the millimeter-scale spatial resolution introduces ambiguity in the inference of the underlying projectome. Resolving it demands evaluation of an enormous number of potential pathways in order to estimate the full projectome from DW-MRI data.

As a consequence, DW-MRI axonal-tracing techniques often estimate only one projection at a time, attempting to trace a single fiber through the white matter using only local measurements, with no regard for the paths of other fibers. These local, greedy optimization methods are not well suited for estimating the entire projectome, as they ignore critical global criteria (such as data prediction, where the model predicts the diffusion data that matches the measurements, and physical-volume constraints, where white-matter volume is finite).

To address these shortcomings we have developed a parallel algorithm for global projectome evaluation that

uniquely accounts for global prediction error and volume conservation.³⁴ Leveraging the IBM Blue Gene/L supercomputing architecture, the algorithm first creates a massive database of 180 billion candidate pathways using multiple local tracing algorithms, then employs a global-optimization algorithm to select a subset of these candidates as the projectome. The estimated projectome (in the figure) accounts for 72 million projections per square centimeter of cortex and is therefore the highest-resolution, volume-conserved, collaborative projectome of the human brain. This surpasses previously achieved projectome resolutions by a factor of at least a thousand.

Figure 1. Top view of a subset of the human-brain projectome created using our algorithm. Shown are major long-range connections of the brain, estimated through diffusion-weight MRI, bottom, with several important groups of pathways assigned distinct colors; the cortex (gray area) is for reference in the opposite hemisphere.



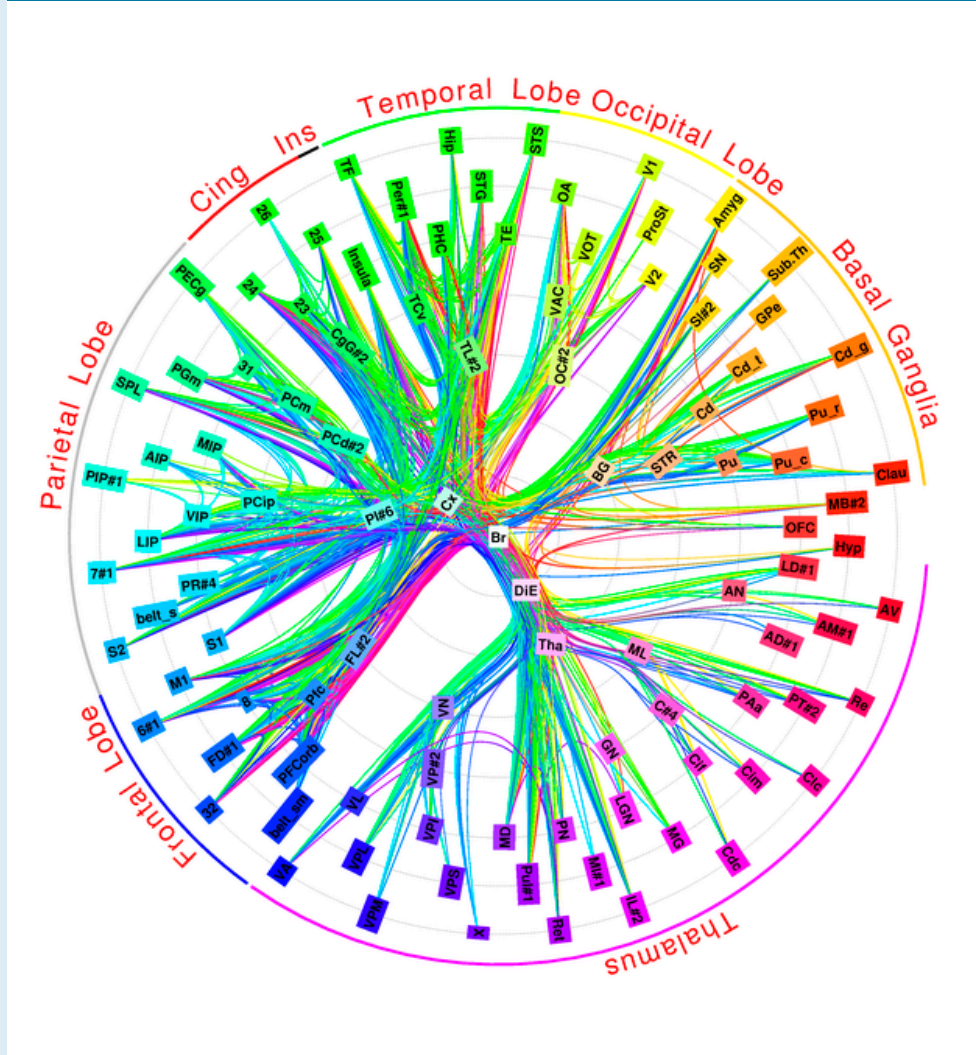
Analyzing White-Matter Pathways in the Macaque Monkey Brain

Anatomical tracing in experimental animals has historically been the pervasive technique for mapping white-matter pathways. In these experiments, a dye is injected in one brain area and its percolation studied to discover white-matter projections to other brain areas. Thousands of such measurements, collected over decades, have generated a vast, but sometimes inconsistent, database of projections. We undertook the challenge of constructing, visualizing, and analyzing a unified, consistent white-matter graph spanning the macaque brain.²⁸

We derived a novel white-matter graph incorporating 410 published anatomical tracing studies of the macaque brain from the neuroinformatic database CoCoMac.¹⁹ Our graph consists of 383 hierarchically organized areas spanning cortex, thalamus, and basal ganglia; it also has 6,602 directed edges and captures well-known cortico-cortical, cortico-subcortical, and intra-subcortical white-matter pathways. This graph is three times larger than the largest previous white-matter network of the macaque brain and is eight times larger than one of the most commonly analyzed white-matter networks of the macaque brain.

We have unearthed several critical insights by leveraging the unprecedented scale of our graph and state-of-the-art tools from network theory, which has also proved invaluable to understanding the hidden structure of graphs (such as the Web, metabolic pathways, and social networks). The degree distribution of the graph is consistent with an exponential distribution and is not scale-free, thus settling a much-debated, foundational open question. The graph has six degrees of separation, is a small-world network, and is characterized by the principle of organized complexity. Additionally, the graph revealed that the prefrontal cortex, the seat of

Figure 2. A white-matter graph in the macaque monkey brain consisting of 102 hierarchically organized super-areas and 1,138 connections. Each node is represented by a colored rectangle, labeled with an abbreviation from CoCoMac nomenclature. Each edge is represented by a colored curve such that the color transitions smoothly from the color of its target node to the color of its source node. Bundles of white-matter projections criss-cross the expanse of the brain.



executive function, contains the lion's share of topologically central areas. Finally, the graph embodies a tightly integrated core circuit that corresponds extremely well with a network believed to be the substrate for higher cognition and consciousness. It is quite remarkable and reassuring that the graph recapitulates critical known fiber pathways in the visual system, the dorsal-ventral pathways,

thalamocortical relays, and numerous corticocortical, corticosubcortical, and subcortical fiber systems implicated in specific cognitive functions. Simulation of circuits incorporating these fiber systems may yield specific insights into these cognitive functions.

To compactly visualize the large, high-resolution graph and make it amenable to simulation, we aggregate

neighboring areas into a smaller number of super-areas, thus sacrificing resolution. A connection between two brain areas in the original graph results in a connection between corresponding super-areas. The smaller, low-resolution graph (in the figure) contains 102 super-areas and, after eliminating duplicates and self-loops, 1,138 connections.

simulations at the scale of small mammalian brains. Though we have only humble achievements to report, our aspirations are lofty. We seek nothing less than to discover, demonstrate, and deliver the core algorithms of the brain and gain a deep scientific understanding of how the mind perceives, thinks, and acts. This will lead to novel cognitive systems, computing architectures, programming paradigms, practical applications, and intelligent business machines.

Rationale

Our rationale was aptly and eloquently captured by Churchland and Sejnowski, writing, “It would be convenient if we could understand the nature of cognition without understanding the nature of the brain itself. Unfortunately, it is difficult if not impossible to theorize effectively on these matters in the absence of neurobiological constraints. The primary reason is that computational space is consummately vast, and there are many conceivable solutions to the problem of how a cognitive operation could be accomplished. Neurobiological data provide essential constraints on computational theories, and they consequently provide an efficient means for narrowing the search space. Equally important, the data are richly suggestive in hints concerning what might really be going on and what computational strategies evolution might have chanced upon.”

Neuroscience today is rich in detailed biological observations, as reflected in the sheer size—1,414 pages—of *Principles of Neural Science*, a modern introductory textbook by Kandel et al.¹⁸ As neuroscientists, we view these observations as a web of clues to the biological mechanisms of cognition. As engineers, we view them as something else entirely. The brain is an example solution to the problem of cognitive computing, and the observations of neuroscience are a partial set of constraints on the form of that solution. The trick to leveraging neuroscience in the name of cognitive computing is to separate the wheat from the chaff.

Here, we explore the fundamental neuroscientific constraints on building a functional simulation of the brain,

first describing structural constraints learned from the wiring diagram of the brain. The central message is the brain’s neuronal network is a sparse, directed graph organized at multiple scales. In particular, local, short-range connections can be described through statistical variations on a repeating canonical subcircuit, whereas global, long-range connections can be described through a specific, low-complexity blueprint. We highlight what neurophysiology has taught us about the dynamics of computation and communication within this network. Our thesis is that the computational building blocks of the brain (neurons and synapses) can be described by relatively compact, functional, phenomenological mathematical models, and that their communication can be summarized in binary, asynchronous messages (spikes).

The overarching motivation of our approach is the fact that the behavior of the brain apparently emerges via non-random, correlated interactions between individual functional units, a key characteristic of organized complexity. Such complex systems are often more amenable to computer modeling and simulation than to closed-form analysis and often resist piecemeal decomposition. Thus, empowered by strides in supercomputing-based simulation, the rationale for our approach rests in our conviction that large-scale brain simulations, at the appropriate level of abstraction, amount to a critical scientific instrument, offering opportunities to test neuroscientific theories of computation and to discover the underlying mechanisms of cognition.

A critical judgment must be made as to the appropriate level of abstraction for simulation. This conundrum must be faced when modeling any physical system. If we would choose too high a level of abstraction, the black boxes within the model will themselves be hopelessly complicated and likely map poorly onto reality as our understanding grows. If we abstract away too little and work at too high a resolution, we will squander computational resources and obscure our own understanding with irrelevant detail. Unfortunately, no oracle exists to instruct us as to the correct balance

between abstraction and resolution at the outset. The only solution is to experiment and explore as a community. It is a virtue that different schools of thought have emerged,^{13,14,16,25,38} each with an argument for its own chosen level of abstraction in conceptualizing and modeling the brain. The most established traditions are at relatively high levels of abstraction and include efforts in AI, cognitive science, visual information processing, connectionism, computational learning theory, and Bayesian belief networks.^{4,5,9,21–24,27,30,35,37} Meanwhile, other efforts have sought to pin down the opposite end of the spectrum, striving for ever-higher levels of reductionist biological detail in simulating brain tissue.^{20,36} Here, we strike a balance between these extremes¹¹ and advocate a middle path,^{12,15,17} one more faithful to the neuroscience than to an abstract connectionist model, yet less detailed than an exhaustive, biophysically accurate simulation. Depending on context, a telescope, a microscope, and binoculars each has a place in a scientist’s repertoire.

Neuroanatomy

A central tenet of neuroscience, sometimes called the “neuron doctrine,” posits that specialized cells in the brain, the neurons, are the biological substrate of brain computation. The function of individual neurons is covered later in the section on neurophysiology, but for now, neuronal function can be abstracted to receiving, integrating, and sending binary messages. These messages are communicated at points of contact, dubbed synapses by Sir Charles Sherrington in England in 1897. Through messaging, neurons collaborate to form networks that engender powerful capabilities, vastly more sophisticated than the processing capacity of individual neurons. To understand brain function, it is crucial to understand the organization of neural circuitry.

Connectivity in the brain is sparse. Adult humans have about 100 trillion synapses, six orders of magnitude less than would be required to completely and directly connect the tens of billions of neurons that make up the brain. Moreover, there is strong evidence that biology has a relatively compact algo-

rhythm for assembling this sparse network; animals with much larger brains and higher cognitive abilities (such as apes) do not have proportionately larger swaths of their genome devoted to neural function than animals with more modest brains and abilities (such as rodents). Perhaps it is unsurprising that neuroanatomists have not found a hopelessly tangled, arbitrarily connected network, completely idiosyncratic to the brain of each individual, but instead a great deal of repeating structure within an individual brain and a great deal of homology across species.

At the surface of the brains of all mammals is a sheet of tissue a few millimeters thick called the cerebral cortex, or simply cortex, thought to be the seat of higher cognition. It is organized at multiple scales, including layers running horizontally, vertically integrated columns spanning its depth, large functionally defined areas consisting of millions of columns, and ultimately networks of multiple areas that are tightly linked. Within this system, neurons are connected locally through gray-matter connec-

tions, as well as through long-range white-matter connections that leave the cortex to travel to distant cortical regions or sub-cortical targets (see Figures 1 and 2).

One of the earliest discoveries suggesting structure within cortex was the six distinct horizontal layers spanning the thickness of the cortical sheet. A specific network of connections between and within these cortical layers has been identified and studied,⁶ giving rise to characteristic patterns of interlaminar activity propagation. We adapted this canonical laminar cortico-thalamic architecture into an archetypical gray-matter network amenable to simulation (see Figure 3).

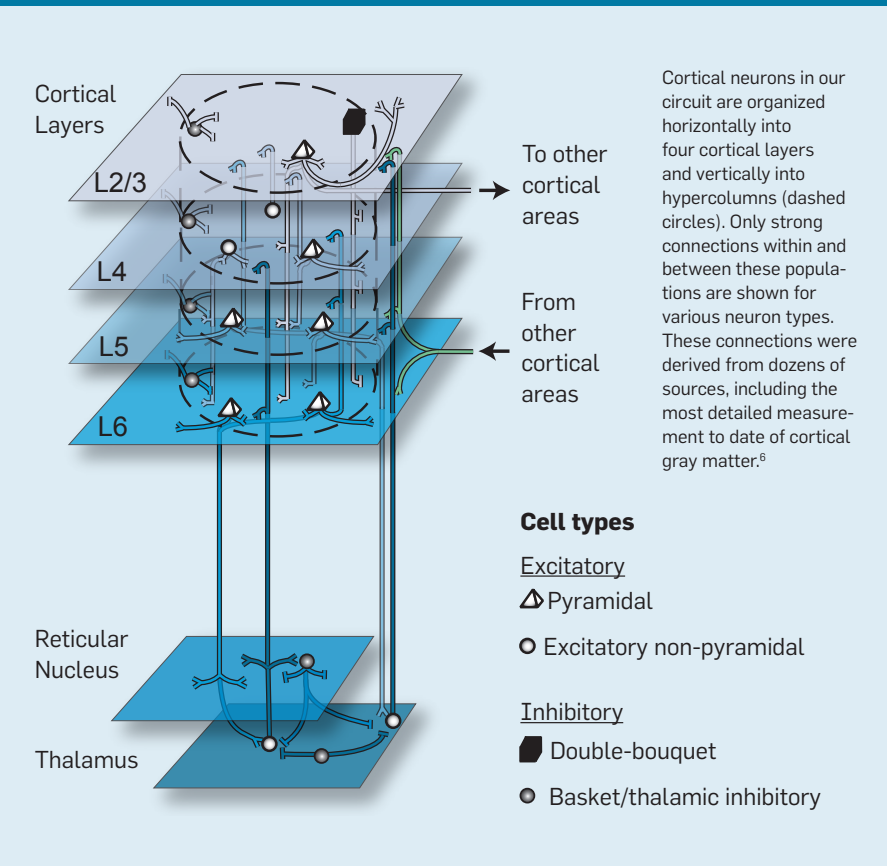
The connections between layers are principally vertical, with limited lateral spread, leading to a columnar structure tens or hundreds of microns in diameter, referred to as a “cortical column.” In many cortical areas, it has been demonstrated that neurons within the same column share related functional characteristics, suggesting that columns are functional, as well as structural, entities.^{6,29} The information

collected by measurements at the columnar scale has been instrumental in creating our large-scale brain models, as in Figure 3.

Cortical columns organize into cortical areas that are often several millimeters across and appear to be responsible for specific functions, including motor control, vision, and planning. Suggesting the possibility of a specific cortical circuit for each function, the famous Brodmann atlas, *Localization in the Cerebral Cortex*, offers a segmentation of the brain into cortical areas based on cellular density variations within the six cortical layers.¹⁸ For example, Brodmann area 17 has been definitively linked to core visual-processing functionality. Decades of work by hundreds of scientists have focused on understanding the role each cortical area plays in brain function and how anatomy and connectivity of the area serve that function.

While overwhelming evidence in the 20th century supports the functional specialization of cortical areas, the brain also demonstrates a remarkable degree of structural plasticity. For example, it has been demonstrated that an area normally specialized for audition can function as one specialized for vision, and vice versa, by rewiring the visual pathways in the white matter to auditory cortex and the auditory pathways to visual cortex in the developing ferret brain. This astonishing natural reconfigurability gives hope that the core algorithms of neurocomputation are independent of the specific sensory or motor modalities and that much of the observed variation in cortical structure across areas represents a refinement of a canonical circuit; it is indeed this canonical circuit we wish to reverse engineer. The existence of such a canonical microcircuit is a prominent hypothesis,²⁹ and while a great deal about the local cortical wiring has been measured,⁶ the exact form of this microcircuit remains unknown and its role in neurocomputation undemonstrated. Even if a base canonical circuit can be found, to unlock its potential we must also identify and implement the accompanying plasticity mechanisms responsible for tailoring, refining, and elaborating the canonical circuit to its specific function during development and in adult

Figure 3. A circuit diagram of the thalamocortical system simulated on the C2 cortical simulator.



learning. We later revisit the topic of plasticity, specifically its possible local, synaptic mechanisms.

At the coarsest scale of neuronal system organization, multiple cortical areas form networks to address complex functionality. For example, when reading, the brain executes a deft series of intricate eye movements that scan and fixate within words to extract a series of lines and edge combinations (letters) forming intricate spatiotemporal patterns. These patterns serve as keys to unlock a tome of linguistic knowledge, bathing the brain in the sights, sounds, smells, and physicality of the words' meaning. It is astounding that this complex functionality is mediated by a small network of tightly connected, but spatially distant, brain areas. This gives hope that distinct brain functions may be supported by signature sub-networks throughout the brain that facilitate information flow, integration, and cooperation across functionally differentiated, distributed centers. In 2009 and 2010, our group at IBM Research-Almaden achieved two breakthroughs^{28,32} in measuring and analyzing the white-matter architecture of macaque and human brains as a means of furthering our understanding of networks of brain areas (see the sidebars "Analyzing White-Matter Pathways in the Macaque Monkey Brain" and "Measuring White-Matter Pathways in the Human Brain").

Neurophysiology

The adaptation of a biological cell into a structure capable of receiving and integrating input, making a decision based on that input, and signaling other cells depending on the outcome of that decision is a truly remarkable feat of evolution. Such cells, the neurons, were proposed to underlie information processing and cognitive function within the brain by Nobel laureate Santiago Ramón y Cajal in 1891. Neuronal function arises from a neuron's three main structural components: dendrites, tree-like structures that receive and integrate inputs; a soma, where decisions based on these inputs are made; and an axon, a long narrow structure that transmits signals to other neurons near and far. It is quite impressive that while cells are typically



The adaptation of a biological cell into a structure capable of receiving and integrating input, making a decision based on that input, and signaling other cells depending on the outcome of that decision is a remarkable feat of evolution.



microscopic, axons can grow to more than a meter in length.

At the root of signal integration and transmission within a neuron are fluctuations in the neuron's membrane potential, the voltage difference across the membrane that separates the interior and exterior of a cell. These fluctuations occur when ions cross the neuron's membrane through channels that can be opened and closed selectively. If the membrane potential crosses a critical threshold, the neuron generates a spike (its determination that it has received noteworthy input), which is a reliable, stereotyped electrochemical signal sent along its axon. Spikes are the essential information couriers of the brain, used in the sensory signals the retina sends down the optic nerve in response to light, in the control signals the motor cortex sends down the spinal cord to actuate muscles, and in virtually every step in between.

When a spike arrives at the end of its axon, the nature of the signal changes. Synapses are tiny structures that bridge the axon of one neuron to the dendrite of the next, transducing the electrical signal of a spike into a chemical signal and back to electrical. The spiking neuron, called the "presynaptic neuron" in this arrangement, releases chemicals called neurotransmitters at the synapse that rapidly travel to the other neuron, called the "postsynaptic neuron." The neurotransmitters trigger ion-channel openings on the surface of the postsynaptic cell, subsequently modifying the membrane potential of the receiving dendrite. These changes can be either excitatory, meaning they make target neurons more likely to fire, or inhibitory, making their targets less likely to fire. Both the input spike pattern received and the neuron type determine the final spiking pattern of the receiving neuron. Through this process, the essentially digital electrical signal of the spike sent down one neuron is converted first into a chemical signal that can travel between neurons, then into an analog electrical signal that can be integrated by the receiving neuron.

The magnitude of this analog postsynaptic activation, called "synaptic strength," is not fixed over an organ-

Cortical Simulator Design and Implementation

Since 2007, we have been developing the C2 near-real-time mammalian-scale cortical simulator^{2,3} to harness the distributed memory multiprocessor architecture of IBM Blue Gene systems (see Figure 4). Here, we discuss the core architecture of the simulator and highlight key innovations along the dimensions of memory, computation, and communication.

The cortical simulator includes a clock-driven component with discrete time steps, as well as an event-driven component. In the former, the state of the neurons is updated once every time step, typically either one millisecond or one-tenth of one millisecond of simulated time. In the latter, when a neuron fires, it creates a spike event that is then delivered to the synapse of a target neuron after a tunable axonal delay. The spike event has two essential functions: change the membrane potential of the target neuron and possibly trigger a change to the strength of the synapses on the axon and dendrites of the spiking neuron.

The entire state of the simulation (consisting of neurons, synapses, and transient spike messages) is evenly distributed among the local memories of the multiprocessor system. Each processor maintains the state of a group of neurons and all synapses providing input to these neurons. A notable C2 innovation is the memory-efficient representation of synaptic state, facilitating significantly increased model scales.

C2 harnesses a large number of processors while fully exploiting the computational capacity of each processor to achieve near-real-time simulation. Its design ensures that the number of computational operations at every time step is proportional to the number of spikes, rather than to the vastly larger number (typically a thousandfold) of synapses.

Most notably, C2 employs a novel synchronization technique requiring only two communication steps, in sharp contrast to previous algorithms that used communication steps in proportion to number of processors. When simulating with more than a hundred thousand processors, such communication optimizations are indispensable.

ism’s lifetime. Thus, the influence one neuron has on another can change, altering the functional relationships within a network of neurons. Canadian psychologist Donald O. Hebb’s famous conjecture for synaptic plasticity is “neurons that fire together, wire together,” or that if neuron A and B commonly fire spikes at around the same time, they will increase the synaptic strength between them. One modern refinement of Hebb’s idea is that synaptic strengths may change depending on the relative timing of pre- and post-synaptic spikes through a mechanism called “spike-timing dependent plasticity,” or STDP,³³ so neuron A strengthens its connection to neuron B if A tends to fire just before B fires, while connection strength is weakened if the firing order is reversed. There are also

ongoing research efforts to link neuro-modulatory chemicals, like dopamine, to more complex mechanisms for synaptic plasticity that resemble update rules from reinforcement learning.^{31,34} The mechanisms of synaptic plasticity are a focus of active research, but no one can say for certain which mechanisms are most prevalent or most significant. However, it is widely believed among brain researchers that changes in synaptic strength underlie learning and memory, and hence that understanding synaptic plasticity could provide crucial insight into cognitive function.

While this provides a rough outline of neuron behavior, neuroscientists have uncovered a much more detailed picture of neuron function, including a host of different ion channels that pro-

duce oscillatory changes in membrane potential and regulate firing patterns, different synapse types that operate over a range of time courses, neuro-modulators that produce changes in neuron behavior, and many other features that influence function.¹⁸ Many different types of neurons can be distinguished based on these features, which have been captured in a number of models.

It should be noted that though it is widely agreed that spikes are the brain’s primary information couriers, considerable debate concerns how spikes encode information. The dominant view has been that cortical neurons encode information in terms of their instantaneous firing rates, and the relative timing between spikes is essentially irrelevant. Studies have shown there is additional value in the precise timing of spikes, though the lion’s share of the information is available in the spike rate. Further, recent evidence suggests the brain is able to detect and exploit artificially induced precise spiking timing.

We embrace the proposition that, because spikes are the universal currency of neuronal communication, a simulated network that reproduces the brain’s temporal pattern of spiking must necessarily constitute a sufficient simulation of neural computation. In an idealized thought experiment, this simulation would predict the exact temporal pattern of spikes across the entire brain (including neuron bursting, correlations between neurons, and temporal synchrony) in response to arbitrary stimuli and contextual placement. It is uncontroversial to say that such an achievement, were it possible, would implicitly recapitulate biological neurocomputation. However, many researchers dispute that this implies that spikes are the correct level of abstraction at which to study and simulate the nervous system. Those who believe that working at a higher level of abstraction is preferable argue that the details of such a spiking simulation are irrelevant to the fundamental principles of cognition and actually obscure the key algorithms of brain-based computation.²⁷ On the other hand, those who believe that working at a finer resolution is required assert

Neurons and synapses in representative mammals.

	Mouse	Rat	Cat	Monkey	Human
Billions of Neurons	0.016	0.055	0.763	2	20
Trillions of Synapses	0.128	0.442	6.10	16	200

that a precise reproduction of spike trains, though sufficient, is unattainable without including in the simulation the detailed dynamics of dendritic compartments, ion concentrations, and protein conformations. However, few brain researchers dispute that neurons are the fundamental cellular units of computation and that spikes are the messages passed between them. Because spikes therefore constitute a preferred level of description of neural communication, we focus on simulations dealing directly with spikes.^{2,3,15}

Supercomputing Simulations

Neuroanatomy and neurophysiology, together, have produced a rich set of constraints on the structure and the dynamics of the brain. In our own work, we have aimed to integrate and operationalize a judiciously chosen subset of these constraints into a single computational platform, a mammalian-scale brain simulator.

The essential ingredients of the simulator include phenomenological model neurons exhibiting a vast behavioral repertoire, spiking communication, dynamic synaptic channels, plastic synapses, structural plasticity, and multi-scale network architecture, including layers, minicolumns, hypercolumns, cortical areas, and multi-area networks, as in Figure 3. Each of these elements is modular and individually configurable, so we can flexibly test a multitude of biologically motivated hypotheses of brain structure and dynamics. The ensuing enormous space of possible combinations requires that simulations run at speeds that permit rapid, user-driven exploration.

Neural simulations have a rich history dating to the 1950s. Since then, research in cortical simulations (see Figure 3) has progressed along two paths: detail and scale. Several publicly available simulators, including NEURON and GENESIS, allow for detailed simulation of a small number of neurons.⁷ Unfortunately, incorporating such fine biophysical detail renders the task of near-real-time simulations at mammalian scale computationally impractical. On the other hand, using compact phenomenological neurons, other studies have demonstrated sim-

ulations of millions of neurons and billions of synapses.^{3,15} Our objective is to push the boundaries of the state of the art along the dimensions of model scale and neuroanatomical detail while achieving near-real-time simulation speed.

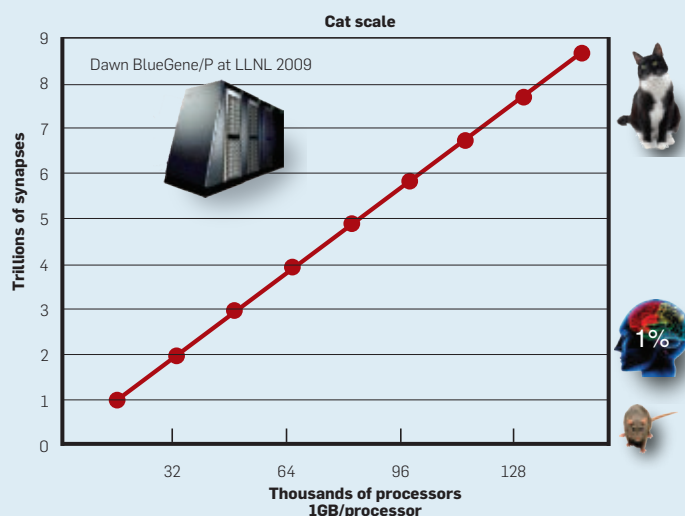
Simultaneously achieving scale, speed, and detail in one simulation platform presents a formidable challenge with respect to the three primary resources of computing systems: memory, computation, and communication. For example, the cat cerebral cortex has almost a billion neurons and more than six trillion synapses (see the table opposite). Since synapses outnumber neurons by a factor of 10,000, memory requirements representing the state of the simulation scale in direct proportion to the number of synapses. Consequently, even if we could represent the state of a synapse in 1B, a cat-scale simulation would require at least 6TB of main memory; efficient synaptic data structures require about 16B of memory per synapse. Further, assuming that each neuron is updated once every millisecond, the dynamical difference equations governing neuronal state evolution must be computed one trillion times per second. With a biologically plausible average neuron firing rate

of once per second, most synapses would receive a spike once a second, so six trillion spike messages must be communicated across the network. To meet this demand, we leverage magnificent strides in supercomputing, coupled with key innovations in algorithms and software architecture.

Along the hardware dimension, the Blue Gene supercomputer system offers large numbers of computational processors, vast amounts of main memory in a distributed architecture, and low-latency, high-bandwidth communication subsystems. Along the software dimension, we have developed a cortical simulator we call C2 that exploits the distributed-memory multiprocessor architectures. We have performed simulations of increasing scale and incorporated progressively richer neurophysiological and neuroanatomical constraints in our simulations (see the sidebar “Cortical Simulator Design and Implementation”).

Since 2007, Our simulations have grown steadily in scale, beginning with early work at a scale of mouse and rat³ cortices. We obtained our most recent result (see Figure 4) in May 2009 in collaboration with Lawrence Berkeley National Laboratory using the Dawn Blue Gene/P system, achieving

Figure 4. Scaling cortical simulations with C2.



Simulations on a Blue Gene/P supercomputer, including a 1% human-scale cortical model with 1.97 trillion synapses employing 32,768 processors and 32TB of main memory, culminating in a cat-scale cortical model with 8.61 trillion synapses using 147,456 processors and 144TB of main memory.

the newsworthy milestone of cat-scale cortical simulations, roughly equivalent to 4.5% of human scale, fully utilizing the memory capacity of the system.² The networks demonstrated self-organization of neurons into reproducible, time-locked, though not synchronous, groups.³ The simulations also reproduced oscillations in activity levels often seen across large areas of the mammalian cortex at alpha (8Hz–12Hz) and gamma (> 30Hz) frequencies. In a visual stimulation-like paradigm, the simulated network exhibited population-specific response latencies matching those observed in mammalian cortex.² A critical advantage of the simulator is that it allows us to analyze hundreds of thousands of neural groups, while animal recordings are limited to simultaneous recordings of a few tens of neural populations. Taking advantage of this capability, we were able to construct a detailed picture of the propagation of stimulus-evoked activity through the network; Figure 5 outlines this activity, traveling from the thalamus to cortical layers four and six, then to layers two, three, and five, while simultaneously traveling laterally within each layer.

The C2 simulator provides a key integrative workbench for discovering algorithms of the brain. While our simulations thus far include many key

features of neural architecture and dynamics, they only scratch the surface of available neuroscientific data; for example, we are now incorporating the long-distance white-matter projections (see the first two sidebars and Figures 1 and 2), other important sub-cortical structures (such as the basal ganglia), and mechanisms for structural plasticity. We remain open to new measurements of detailed cortical circuitry offered by emerging technologies.

The realistic expectation is not that cognitive function will spontaneously emerge from these neurobiologically inspired simulations. Rather, the simulator supplies a substrate, consistent with the brain, within which we can formulate and articulate theories of neural computation. By studying the behavior of the simulations, we hope to reveal clues to an overarching mathematical theory of how the mind arises from the brain that can be used in building intelligent business machines. In this regard, the simulation architecture we built is not the answer but the tool of discovery, like a linear accelerator, laying the groundwork for future insight into brain computation and innovations in neuromorphic engineering.

Prospective

The quest for intelligent machines ulti-

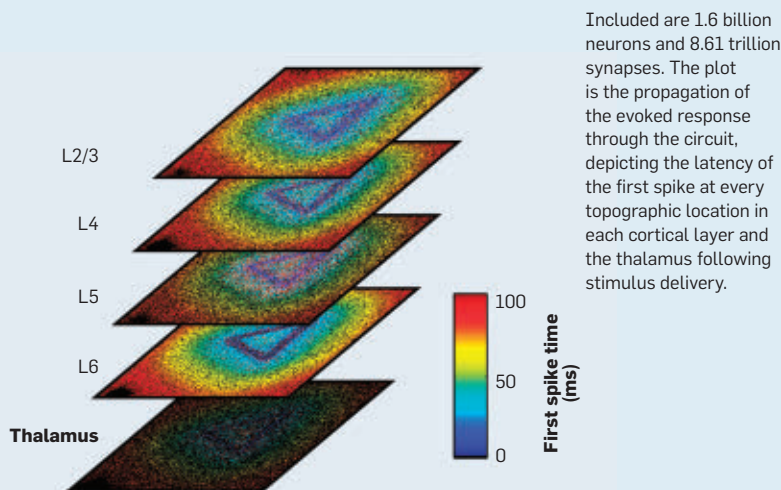
mately requires new breakthroughs in philosophy, neuroanatomy, neurophysiology, computational neuroscience, supercomputing, and computer architecture orchestrated in a coherent, unified assault on a challenge of unprecedented magnitude. The state of today's effort in cognitive computing was best captured by Winston Churchill: "Now this is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning."

On the heels of the unprecedented simulation scale and the trends in development of supercomputer technology, the good news is that human-scale cortical simulations are not only within reach but appear inevitable within a decade.

The bad news is that the power and space requirements of such simulations may be many orders of magnitude greater than those of the biological brain. This disparity owes its genesis to the salient differences between the von Neumann architecture and the brain itself.³⁹ Modern computing posits a stored program model, traditionally implemented in digital, synchronous, serial, centralized, fast, hardwired, general-purpose, brittle circuits, with explicit memory-addressing imposing a dichotomy between computation and data. In stark contrast, the brain uses replicated computational units, neurons and synapses, implemented in mixed-mode analog-digital, asynchronous, parallel, distributed, slow, reconfigurable, specialized, fault-tolerant biological substrates, with implicit memory addressing blurring the boundary between computation and data.

The elegance and efficiency of biology entices us to explore entirely new computing architectures, system designs, and programming paradigms. Under the umbrella of the U.S. Defense Advanced Research Projects Agency (DARPA) Systems of Neuromorphic Adaptive Plastic Scalable Electronics initiative, beginning in 2008, we have embarked on an ambitious program to engender a revolutionarily compact, low-power neuromorphic chip comprising one million neurons and 10 billion synapses per square centimeter by exploiting breakthroughs in nanotechnology and neuromorphic very large-scale integration.²⁶

Figure 5. Simulated response of thalamocortical circuitry to a triangle-shape stimulus.



Included are 1.6 billion neurons and 8.61 trillion synapses. The plot is the propagation of the evoked response through the circuit, depicting the latency of the first spike at every topographic location in each cortical layer and the thalamus following stimulus delivery.

Finally, the ugly news is that the core set of algorithms implemented within the brain are as yet undiscovered, making our task as replete with uncertainty as it is rich with opportunity. Confronting this challenge requires the sustained, coherent effort of a dedicated interdisciplinary community of researchers endowed with substantial resources.¹ At the moment, this grand endeavor proceeds in parallel at multiple scales of investigation: abstract cognitive primitives and artificial neural networks; extremely detailed biological models; and fundamental language of spiking communication favored by us and others. We hope future discoveries will demonstrate these approaches to be complementary, each with its own virtues and each contributing to a unified solution to the challenge of cognitive computing. We share the inspired enthusiasm of U.S. Secretary of Energy and Nobel laureate Steven Chu: “I do not underestimate the difficulty of meeting these challenges, but I remain optimistic that we can meet them. I believe in the vitality of our country and our economy, and as a scientist, I am ever optimistic at our ability to extend the boundaries of what is possible.”

Acknowledgments

The research reported here was sponsored by DARPA's Defense Sciences Office's Program: Systems of Neuro-morphic Adaptive Plastic Scalable Electronics (SyNAPSE) under Contract HR0011-09-C-0002. The views and conclusions are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of DARPA or of the U.S. Government.

We would like to thank Horst Simon for his collaboration and the U.S. Department of Energy National Nuclear Security Administration Advanced Simulation and Computing Program for time on the Dawn Blue Gene/P supercomputer at Lawrence Livermore National Laboratory operated by Lawrence Livermore National Security, LLC, for the U.S. Department of Energy, National Nuclear Security Administration, under Contract DE-AC52-07NA27344. We have also benefited tremendously from supercomputing

resources at IBM Research-Almaden, IBM Research-Thomas J. Watson Research Center, and KAUST/IBM Center for Deep Computing Research.

We are grateful to T. Binzegger, R.J. Douglas, and K.A.C. Martin for sharing their thalamocortical connectivity data and indebted to Gregory Corrado for numerous insightful discussions and significantly clarifying our exposition here. C

References

1. Albus, J.S., Bekey, G.A., Holland, J.H., Kanwisher, N.G., Krichmar, J.L., Mishkin, M., Modha, D.S., Raichle, M.E., Shepherd, G.M., and Tononi, G. A proposal for a decade of the Mind Initiative [Letter]. *Science* 317, 5843 (Sept. 7, 2007), 1321.
2. Ananthanarayanan, R., Esser S.K., Simon H.D., and Modha, D.S. The cat is out of the bag: Cortical simulations with 10^9 neurons and 10^{13} synapses. Gordon Bell Prize Winner. In *Proceedings of the ACM/IEEE Conference on Supercomputing* (Portland, OR, Nov. 14–20). ACM, New York, NY, 2009, 1–12.
3. Ananthanarayanan, R. and Modha, D.S. Anatomy of a cortical simulator. In *Proceedings of the ACM/IEEE Conference on Supercomputing* (Reno, NV, Nov. 10–16). ACM, New York, NY, 2007, 3–14.
4. Arbib, M., Ed. *The Handbook of Brain Theory and Neural Networks*. MIT Press, Cambridge, MA, 2002.
5. Baumgartner, P. and Payr, S. Eds. *Speaking Minds: Interviews with 20 Eminent Cognitive Scientists*. Princeton University Press, Princeton, NJ, 1995.
6. Binzegger, T., Douglas, R.J., and Martin, K.A.A. quantitative map of the circuit of cat primary visual cortex. *Journal of Neuroscience* 24, 39 (Sept. 2004), 8441–8453.
7. Brette, R. et al. Simulation of networks of spiking neurons: A review of tools and strategies. *Journal of Computational Neuroscience* 23, 3 (July, 2007), 349–398.
8. Gazzaniga, M.S. *The Cognitive Neurosciences, Fourth Edition*. MIT Press, Cambridge, MA, 2009.
9. George, D. and Hawkins, J. Towards a mathematical theory of cortical microcircuits. *PLoS Computational Biology* 5, 10 (Oct. 2009), e1000532.
10. Goertzel, B. and Pennachin, C. *Artificial General Intelligence*. Springer, Berlin, Heidelberg, 2009.
11. Herz, A.V.M., Golisch, T., Machens, C.K., and Jaeger, D. Modeling single-neuron dynamics and computations: A balance of detail and abstraction. *Science* 314, 5796 (Oct. 2006), 80.
12. Hill, S. and Tononi, G. Modeling sleep and wakefulness in the thalamocortical system. *Journal of Neurophysiology* 93, 3 (Nov. 2005), 1671–1698.
13. Hodgkin, A.L. and Huxley, A.F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology* 117, 4 (Aug. 1952), 500–544.
14. IEEE Symposium on Information Theory. MIT, Cambridge, MA, 1956.
15. Izhikevich, E.M. and Edelman, G.M. Large-scale model of mammalian thalamocortical systems. *Proceedings of the National Academy of Sciences of the USA* 105, 9 (Mar. 2008), 3593–3598.
16. Jeffress, L.A. *Cerebral Mechanisms in Behavior: The Hixon Symposium*. John Wiley & Sons, Inc., New York, 1955.
17. Johansson, C. and Lansner, A. Towards cortex-sized artificial neural systems. *Neural Networks* 20, 1 (Jan. 2007), 48–61.
18. Kandel, E.R., Schwartz, J.H., and Jessell, T.M. *Principles of Neural Science, Fourth Edition*. McGraw-Hill Medical, New York, 2000.
19. Kötter, R. Online retrieval, processing, and visualization of primate connectivity data from the CoCoMac database. *Neuroinformatics* 2, 2 (June 2004), 127–144.
20. Markram, H. The Blue Brain Project. *National Review of Neuroscience* 7, 2 (Feb. 2006), 153–160.
21. Marr, D. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman, Cambridge, MA, 1983.
22. McCarthy, J. Generality in artificial intelligence. In *Formalizing Common Sense*, V. Lifschitz, Ed. Ablex, Bristol, U.K., 1990, 226–236.
23. McClelland, J.L. and Rumelhart, D.E. Parallel Distributed Processing. In *Psychological and Biological Models, Volume 2*. MIT Press, Cambridge, MA, 1987.
24. McCulloch, W.S. and Pitts, W. How we know universals: The perception of auditory and visual forms. *Bulletin of Mathematical Biology* 9, 3 (Sept. 1947), 127–147.
25. McCulloch, W.S. and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* 5 (1943), 115–133.
26. Mead, C. *Analog VLSI and Neural Systems*. Addison Wesley Publishing Co., Boston, 1989.
27. Minsky, M. *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*. Simon & Schuster, New York, 2006.
28. Modha, D.S. and Singh, R. Network architecture of the long-distance pathways in the macaque brain. *Proceedings of the National Academy of Sciences of the USA* 107, 30 (June 2010), 13485–13490.
29. Mountcastle, V.B. *Perceptual Neuroscience: The Cerebral Cortex*. Harvard University Press, Cambridge, MA, 1998.
30. Rosenblatt, F. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review* 65, 6 (Nov. 1958), 386–408.
31. Schultz, W., Dayan, P., and Montague, P.R. A neural substrate of prediction and reward. *Science* 275, 5306 (Mar. 1997), 1593–1599.
32. Sherbondy, A.J., Dougherty, R.F., Ananthanarayanan, R., Modha, D.S., and Wandell, B.A. Think global, act local: Projectome estimation with BlueMatter. In *Proceedings of the Medical Image Computing and Computer Assisted Intervention Society, Lecture Notes in Computer Science* (London, Sept. 20–24). Springer, Berlin, 2009, 861–868.
33. Song, S., Miller, K.D., and Abbott, L.F. Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience* 3, 9 (Sept. 2000), 919–926.
34. Sutton, R.S. and Barto, A.G. Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review* 88, 2 (Mar. 1981), 135–140.
35. Teddington Symposium on the Mechanization of Thought Processes (National Physical Laboratory, Nov. 24–27). Her Majesty's Stationary Office, London, 1958.
36. Traub, R.D., Contreras, D., Cunningham, M., Murray, H., LeBeau, F.E.N., Roopun, A., Bibbig, A., Wilent, W.B., Higley, M., and Whittington, M.A. Single-column thalamocortical network model exhibiting gamma oscillations, spindles, and epileptogenic bursts. *Journal of Neurophysiology* 93, 4 (Nov. 2005), 2194–2232.
37. Valiant, L.G. *Circuits of the Mind*. Oxford University Press, New York, 2000.
38. von Hayek, F.A. *The Sensory Order: An Inquiry Into the Foundations of Theoretical Psychology*. University of Chicago Press, Chicago, 1952.
39. von Neumann, J. *The Computer and the Brain, Second Edition*. Yale University Press, New Haven, CT, 2000.

For recommended additional reading and sources, see the online appendix in the ACM Digital Library.

Dharmendra S. Modha (dmodha@almaden.ibm.com) is the manager of Cognitive Computing at IBM Research-Almaden, San Jose, CA.

Rajagopal Ananthanarayanan (ananthr@google.com) is a member of the technical staff of Google, Inc., Mountain View, CA, and was at IBM Research-Almaden, San Jose, CA, when he participated in this work.

Steven K. Esser (sesser@us.ibm.com) is a research staff member of IBM Research-Almaden, San Jose, CA.

Anthony Ndirango (andirang@gmail.com) was at IBM Research-Almaden, San Jose, CA, when he participated in this work.

Anthony J. Sherbondy (tony@addepar.com) is an engineer in Addepar, Mountain View, CA, and was at IBM Research-Almaden, San Jose, CA, when he participated in this work.

Raghavendra Singh (raghavs@in.ibm.com) is a research staff member of IBM Research-India, New Delhi.

DOI:10.1145/1978542.1978561

Women increasingly publish in ACM conference proceedings and are notably productive authors.

BY J. MCGRATH COHOON, SERGEY NIGAI,
AND JOSEPH “JOFISH” KAYE

Gender and Computing Conference Papers

SCHOLARS KNOW THAT diversity of thought produces better and faster solutions to complex problems.²⁰ For this reason, as well as other practical and ethical reasons, the computing disciplines strive to improve women's representation in the field.⁵ These efforts often concentrate on piquing the interest of young girls and college students but less often on women's engagement as leaders and problem solvers. Here, we address an aspect of women's leadership by measuring trends and influences on women's authorship of computing-conference papers. Our findings contribute to knowledge about the conditions that promote gender diversity in this important aspect of the intellectual life of the field. Analyzing data from ACM-affiliated conferences reveals several trends:

- ▶ Women's authorship increased from 7% in 1967 to 27% in 2009;
- ▶ Relative to their representation in the likely pool of ACM conference-paper authors, women were especially productive, with each potential woman author writing on average one more paper per year than the potential men authors;
- ▶ The increase in women's share of papers was due in part to their increased numbers in the community of potential authors, as well as general trends in academic publishing¹³; and
- ▶ Women's share of papers at various conferences in any given year varied by conference topic, and to a much lesser extent by paper-acceptance rates; conference size had no notable effect on women's authorship.

We collected and mined data from more than 3,000 ACM-affiliated conferences, workshops, symposia, and forums, 1966–2009, providing evidence of women's increased contribution to this form of professional engagement and contribution to computing. Using custom software called Genderyzer^{a,15} we identified gender for 90% of the 356,703 authors who published papers for ACM events 1966–2009.^b

Summary results outlined in Figure 1 show that over the past 43 years, women comprised 22% of all authors whose gender was ascertainable. From 1966 to 2009, women went from

a Created by author Kaye while a student at Cornell University and an employee of Nokia; <http://genderyzer.com>

b Details on software accuracy, which we believe to be the first tests of such accuracy, are discussed in the section on methods.

» key insights

- **Women's authorship increased from 7% in 1967 to 27% in 2009.**
- **Relative to representation among likely ACM conference-paper authors, women were more productive than men.**
- **The increase in women's share of papers was due to their increased numbers among potential authors and to general trends in academic publishing.**



7% to 27% of ACM conference paper authors; this growth averaged about 0.44 percentage points per year, in established and new conferences alike.

Prior Research on Diversity Influencers

The persistent gender imbalance in computing¹⁹ challenges researchers and practitioners in terms of both explanations and effective intervention. Insight from social psychology, sociology, and feminist studies of technology suggest deep cultural beliefs about gender and technology affect all of us—men and women—when it comes to women’s participation in stereotypically male activities. These stereotypes steer men into and women away from fields like computing, where the alignment of “feminine” and “technical” is less than obvious.

People assess women’s (and men’s) characteristics and accomplishments in light of gender stereotypes that classify technical activities like computing as fitting masculine interests and abilities.^{7,9,22} As a result, women and those who assess their performance tend to downgrade women’s technical abilities relative to those of men with the same computing skills and accomplishments. At the same time, men and the managers who assess their performance could inflate their perceptions of men’s technical abilities. Both genders unconsciously shift their appraisal of others and themselves to better align with the gender stereotypes.⁷ For women, this process often results in low confidence and avoidance or departure from computing.^{6,18}

Mounting evidence supports the stereotype explanation for occupational gender segregation^{4,22}; for example, cross-national comparisons have found that even in societies where men and women are believed to be equally competent and able to achieve at the highest levels, they segregate into different occupations when given the opportunity.⁴ Deeply held cultural beliefs that men and women are fundamentally different come into play when social structures fail to inhibit the influence of these stereotypes, according to Charles’s and Bradley’s empirical study of 21 countries.⁴ They argued that in coun-



The results indicated that for every additional woman with a computing Ph.D., women’s author credits grew by 3.6 papers.



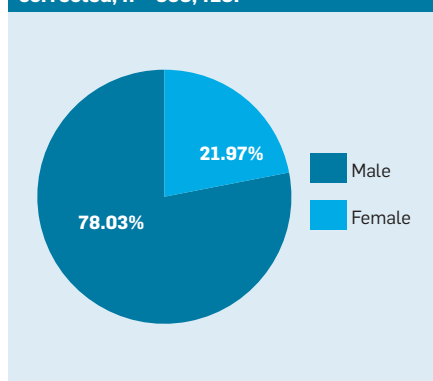
tries where the educational system keeps able girls on the path toward math-based careers through their teens, after which the particularly strong influence of gender stereotypes lessened, women were better represented in computing. This observation implies that social structures (such as educational requirements) can override stereotypes about gender and technology and offer a path toward gender balance in computing.

Theories about the influence of gender and technology stereotypes suggest at least three possible hypotheses about the general characteristics of women’s contributions as authors of computing-conference papers: One is it might be more difficult for women than for men in computing to contribute to the intellectual life of the field; for example, this difficulty could result if reviewers unconsciously downgrade ratings of women’s submissions³ or if low confidence inhibits women’s willingness to publicly exhibit their technical thoughts and advancements in practice.

A second hypothesis is that women in computing might find publishing conference papers among their more easily accomplished professional activities. This ease might relate to alignment of verbal skills with stereotypically feminine abilities. Such alignment is so strong that “textbooks routinely cite sex differences in language competence... as established fact,”²³ with women believed to be more verbally proficient than men. Stereotypes about women’s verbal abilities might offer a comfortable “gender-authentic” role for women in computing. If this is true, it suggests that women might be relatively well represented among authors of computing-conference papers and also cluster in the subfields best aligned with feminine stereotypes.

A third hypothesis is that although stereotypes could hinder entry for most women, those who do go into computing perceive little influence and proceed on equal footing with their male colleagues. Potential support for this hypothesis comes from studies showing that women scientists have more egalitarian unconscious associations about gender and science than do women who are not scientists.²¹

Figure 1. Total gender composition of all ACM conference publications, 1966–2009, corrected; n = 358,428.



Finally, other contemporary social structures might work through stereotypes or, independently, influence women's ACM conference authorship; for example, women's opportunities to write could be unduly affected if teaching and service expectations are heavier for women than for men. These expectation differences could stem from men and women being employed in different types of academic institutions, industry labs, and corporations that are more or less hierarchical in structure.²⁴ Likewise, access to professional networks that afford collaboration might differ for men and for women. Care-related responsibilities, falling disproportionately on women, also appear to have complex influence on paper productivity.¹⁰ All these potential influences warrant investigation but are beyond our scope here.

Here, we report on our initial investigations into the extent women contribute to advancing thought through computing-conference papers and how these contributions are distributed across computing subfields.

Data Set for the Study

We obtained the data for the study on April 26–30, 2009 by screenscraping the proceedings for every conference in the ACM Digital Library. We began with the ACM proceedings page, followed by each list of proceedings and each year's proceedings page^c for each conference. We then processed proceedings pages, using a custom script

Jofish Kaye developed to extract author names and paper titles. The data comprised 432 ACM-affiliated conferences, workshops, and symposia^d held from 1966 to 2009. The resulting data set represents approximately 86,000 papers in approximately 3,100 proceedings. The average conference had 117 authors, though there was considerable variation; the standard deviation in number of authors was 134. There were also many small conference-like events; of the 432 conferences, the average number of authors was fewer than 100 at 316 conferences and fewer than 50 at 164 conferences.

Names Indicate Author Gender

Large data sets of names offer both opportunities and challenges for quantifying women's participation. Because most non-Chinese names are gender-specific, they can be used to identify the male and female representation among people listed in the data. We automated this process with a program called the Genderyzer developed by Kaye and freely available for public use at <http://genderyzer.com>. It compares first names to a data set drawn from various sources, including census data, official national name lists, baby-naming Web sites, and crowd-

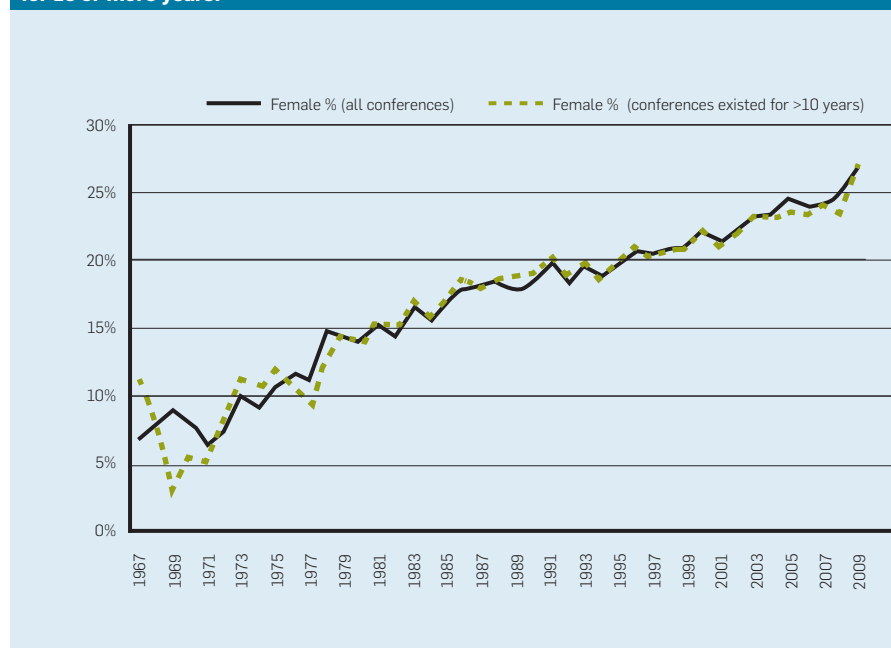
sourced data. Genderyzer labels each name as one of the following categories: male, female, unknown, initials, and ambiguous¹⁵; unknown names are displayed to both the current and the following user for crowdsourcing.

To determine Genderyzer's accuracy before employing it for the study, we conducted tests involving creating independent large data sets of names with a priori known gender and geographic origin. We created six separate data sets with a total of 4,700 observations, including U.S., international, and Asian names drawn from the records of international sports competitions. Applying Genderyzer to this data produced measures of accuracy for the software's performance. We then used it to combine the data sets into two files that mimicked the gender and country-of-origin characteristics of Ph.D. degree recipients in computer and information science in the U.S. over the previous five years.^e Using this data set, we employed a random-sampling algorithm to gauge how well the software performs on a sample with an arbitrary number of observations from a population we believed was very close to the popula-

^e Source: National Center for Education Statistics; <http://webcaspar.nsf.gov/>. The data set provides information on number of Ph.D. graduates in computer science in U.S. academic institutions.

^d We call all these events "conferences" for simplicity.

Figure 2. Percentage of women authors in all ACM conferences vs. conferences that existed for 10 or more years.



^c http://portal.acm.org/browse_dl.cfm?linked=1&part=series&coll=ACM&dl=ACM&CFID=57555177&CFTOKEN=41432595

tion of ACM conferences participants in terms of ethnic origin and sex as determined by Genderyzer. This technique allowed us to formulate 95% confidence intervals for the sex composition of the names the software identified as “ambiguous” and “unknown.”^f

The following example clarifies

^f To be conservative, we used the lower bound of the confidence interval for our analysis.

the method we used for determining the gender composition in a sample of names. Assume we have a population of names with a known 2:2:1 American-Asian-international ratio and 3:1 male-female ratio out of a random sample of 100 names. Based on our testing, we know that Genderyzer on average correctly identifies sex for 70% of the names and places the rest in one of three categories: “unknown,” “ambiguous,” and “initials”

(rather than full names). We repeatedly drew 50 random samples of size 100 from the combined data sets to calculate the average number and 95% confidence intervals for the number of female names in the “ambiguous” and “unknown” categories. Since we had a priori knowledge of sex and country-of-origin variables, we could identify that on average in such a sample 40% and 45% of the names in “unknown” and “ambiguous” categories, respectively, belonged to females. We used the results from our random trials to make well-supported assumptions about the distribution in the “ambiguous” and “unknown” categories to augment the percentage of identifiable names.

We had no knowledge of the actual distribution of “initials” so assumed that names in that category had the same gender composition as the weighted average of our four other categories in the same year. It is possible that this assumption could lead to underestimation of women’s representation if women use initials more than men to avoid possible gender bias. We chose to be conservative in our estimation.

Papers and Authors Increased Exponentially

The annual number of conference papers published by ACM as represented in our data set grew from 149 in 1966 to 12,222 in 2008. This increase is hardly surprising, given the phenomenal growth and differentiation of computing, as well as the general growth in academic publishing.¹³ The number of authors grew even more dramatically—from 389 to 37,944 during the same period. This difference in growth rates of papers and authors is explained by the increasing prevalence of collaborative authorship. In 1966, papers had on average 2.6 authors, but, by 2008, papers had on average 3.1 authors. Over those 43 years, most authors were men, though women authors were increasingly prevalent in recent years. In 2008, there were approximately 2.3 male authors and 0.8 woman authors per published ACM conference paper.

Women’s authorship increased as they garnered Ph.D. degrees. Figure 2 reflects the substantial increases in

Figure 3. Women percentage of all authors, women percentage of Ph.D.s, and women percentage of cumulative Ph.D.s.

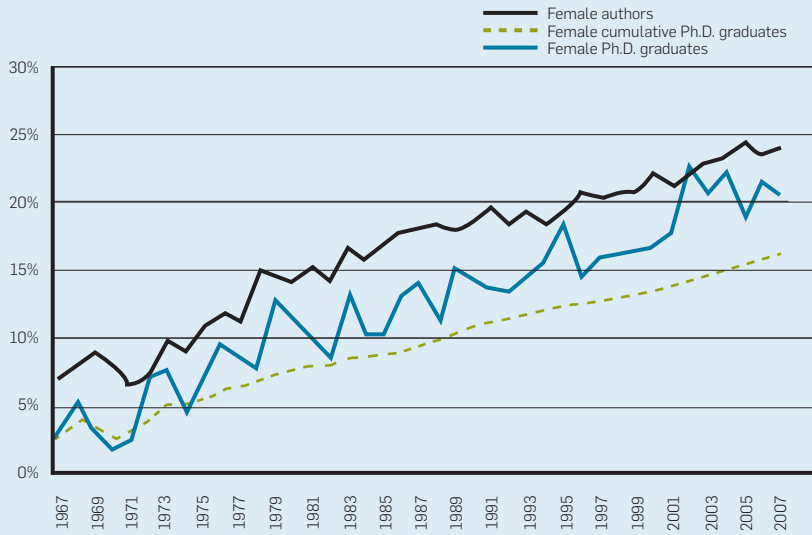


Figure 4. Number of papers published by individual men, individual women, and co-authored by multiple authors.

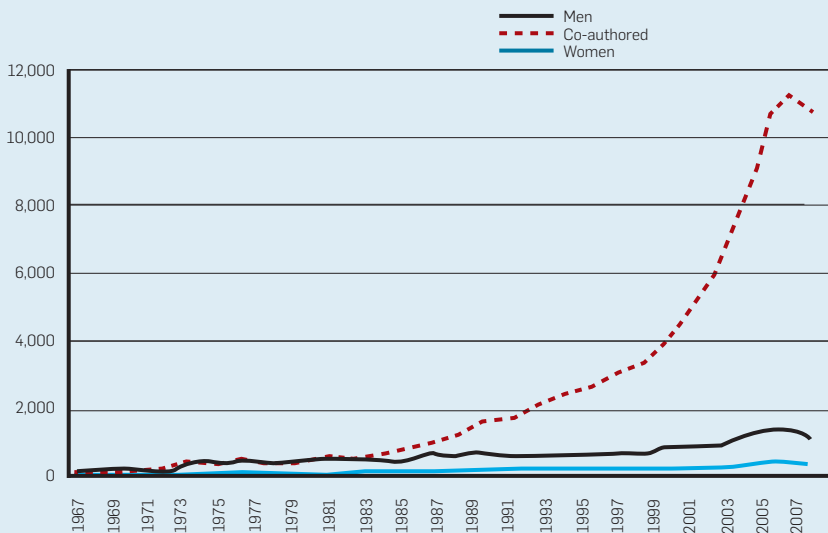


Figure 5. Women's average share of authors, 1998–2008.

% Women Authors	ACM Conferences running for 10 or more years					
10%	APL	ICFF				
12%	GH					
13%	SIGADA					
15%	UIST					
16%	SIGGRAPH	FPGA	POPL	OOPSLA	ISAAC	PLDI
17%	MICRO	SBCCI				
18%	CCS	SC	ISCA			
19%	ACSC	CODES+ISSS	ASPLOS	PODS	SODA	PACT
20%	WSC	ICS	STOC	LCTES	PADS	
21%	KDD	COMM	DATE			
22%	SIGSOFT	SPAA	HPDC	VRST	DAC	PODC
23%	ICSE	IUI	AGENTS			
24%	SIGMETRICS	DOLAP	ISPD	DLS	SPM	
25%	SIGMOD	MSWIM	ISLPED	SAC		
26%	HT	SIGIR	GIS			
27%	ICCAD					
28%	DIAL_M	CIKM	ADC	CHI	MM	
29%	SIGCSE					
30%	JCDL-DL					
32%	ITICSE	SIGDOC				
40%	CPR					
44%	SIGUCCS					

authorship ($B = 0.76$ significant at 1%).

Visually comparing the trends in women's Ph.D.s and authorship makes possible two observations, as reflected in Figure 3:

Parallel rates. Growth in women's publishing rates paralleled growth in women's Ph.D. degree rates; the average annual growth in women's share of ACM conference authorship was 0.44 percentage points, compared with 0.45 points for computing Ph.D. degrees; and

Publishing rate. Women publish at higher rates than one might expect from their representation among Ph.D. holders. In 1967, women's representation among authors was about four points greater than among Ph.D. degree recipients. This overrepresentation persisted in most years, holding for both annual and cumulative percentage of women Ph.D. holders.

We also analyzed the relationship between growth in the cumulative number of Ph.D.s and number of author credits, while accounting for autocorrelation, by running regression in first differences. The results indicated that for every additional woman with a computing Ph.D., women's author credits^j grew by 3.6 papers. Growth was less for men; additional Ph.D.s corresponded to only 2.6 more author credits. We found no correlation between being first or subsequent author and gender. Women and men were equally likely to be first authors on their papers.

These results appear to contradict well-established findings that academic men publish more than academic women, so it is important to recognize that several potential unknown factors might affect men and women authors differently, some of which we discuss later in the section on implications.

Women's sole authorship and collaborative authorship both increased. We investigated the possibility that more collaboration by women could contribute to women's apparent productivity and help explain the upward trend in women's representation

women's share of authorship of ACM conference papers. The increase averaged 0.44 percentage points annually, 1966–2009, with 10-year intervals finding women's share of authorship at 8% in 1968, 15% in 1978, 18% in 1988, 21% in 1998, and 25% in 2008.

Figure 2 also shows this rise in women's participation was not an artifact of newly created conferences catering to women. Tracking a set of 64 longstanding ACM conferences^g resulted in the same trend in the full data set, confirming that women's authorship grew about 18–20 percentage points from 1966 to 2009. We explore two possible explanations for this trend. One may be the increase in women's representation among potential authors, as women earned more computing Ph.D. degrees. As women's representation in the community increased, one would expect a concomitant increase in their contribution to the intellectual life of the field. The second is that women may have benefited disproportionately

from collaboration. Other explanations are also possible, some we hope to explore in future articles.

Research and publication are important activities for most professionals with Ph.D. degrees. Therefore, it should come as little surprise to learn that the proportion of women Ph.D. recipients in computer science^h strongly correlates with women's conference authorship.ⁱ A substantial portion of the upward trend is accounted for by increased women's share of Ph.D. degrees in computing. There is a moderately strong positive association between absolute growth in women Ph.D. graduates and paper

^h Source: National Center for Education Statistics, accessed through <http://webcaspar.nsf.gov/>. The data set (from the National Science Foundation) provides information on number of Ph.D. graduates in computer science in U.S. academic institutions. "Computer science" includes computer and information science, general; information sciences and systems; computer science; management information systems; and management science.

ⁱ Graduate students earn many author credits for papers in the ACM Digital Library, but their number and gender representation should be similar to that of Ph.D. recipients.

^j "Author credits" count authors each time they publish, so we are not comparing degree recipients to papers published or to author names; we instead compare degree recipients with instances of authorship.

^g All ACM conferences that existed for 10 or more years.

among authors. Our analysis suggests that collaboration alone explains little about the increase in women's share of authorship.

Figure 4 outlines both the exponential increase in the number of multiple-authored papers presented at ACM-sponsored conferences and the small increase in number of papers published by lone authors. By 2008, collaboration was most common, with 97% of all ACM conference papers written collaboratively.

From 1966 to 2009, individual men published more ACM papers than did individual women but at about the same gender representation seen for authors overall. The number of papers published by individual men peaked in 2006 at a little over 1,300. Individual women also contributed the most papers that year—about 415, or 24% of all individually authored papers that year.

The trend toward increased co-authorship appears to have begun in the mid-1980s and accelerated in the late 1990s. By contrast, the trend in women's representation among ACM conference authors increased at about the same pace since the early 1970s. This observation suggests that while women may have benefited from the increasingly common practice of co-authorship, collaboration probably does not explain much of the trend toward gender parity among computing-conference-paper authors.

Variations Across Conferences

Women's share of paper authorship varies across ACM conferences; Figure 5 plots women's percentage of authors in 64 longstanding conferences, each averaged over the 10-year interval 1998–2008. The range is from a mean of 10%–44% women authors, with most conferences having 17%–29% women authors. The average percentage of women authors among the 64 conferences was 23%, with a standard deviation of 6%. Tables showing all the large^k ACM conferences with especially high or low average participation of women authors 1998–2008 are available online. We focused on relatively large conferences, dropping

those with fewer than 100 authors.^l Looking at the extremes might hint at alignment with gender stereotypes as a factor in the distribution of women authors across conferences; at the high end, the conference topics are children, education, and human-computer interaction. Any potential misalignment with feminine stereotypes is less obvious at the low end.

The trend over time (not shown in Figure 5) for most (40) of the 64 conferences was a clear upward slope in women's authorship. A number of conferences (18) had neutral slopes for the trend in women's authorship, while (six) had negative slopes indicating declines in women's share of authorship over time, though included conferences with high female participation to begin with.^m There was no obvious pattern reflecting which of the conferences had positive, neutral, or negative slopes.

Conference topic relates to authorship overall, and to women's authorship. Thus far, the data appears to support the hypothesis that female authors might be more prevalent in conferences focused on specific topics. Investigating further, we coded each conference according to its ACM-designated general-topic classifications: Algorithms, Design, Documentation, Economics, Experimentation, Human Factors, Languages, Legal Aspects, Management, Measurement, Performance, Reliability, Security, Standardization, Theory, and Verification. At face value alone, it seems reasonable to expect the topic classifications most closely aligned with

feminine stereotypes would be Human Factors, Design, and Documentation, and those most closely aligned with masculine stereotypes would be Algorithms, Theory, and Security.

We analyzed a subset of the full data set ($n = 391$ conferences) for which we were able to obtain additional information. The data contained all the cases with publicly available information on paper-acceptance rate (used as a proxy for conference prestige), along with conference location and ACM general classifications terms for each conference. Most conferences in this subset were held between 1998 and 2008, but we included 91 conferences dating earlier than 1998 to maximize the number of observations. The earliest conference we included in the set was held in 1981.

Coding by conference topic shows variance in the prevalence of authors publishing on certain topics. The descriptive results show that, like men, women were most likely to publish papers in ACM conferences on Design and on Theory. Human Factors and Algorithms were the next most popular conference topics, with women much more likely than men to publish in conferences on Human Factors and men more likely than women to publish in conferences on Algorithms. The greatest gender differences were evident in conferences focusing on Human Factors, Languages, Algorithms, and Performance, in decreasing order.

Following up on the descriptive evidence, our final statistical analysis used ordinary-least-squares regression to measure factors contributing to variation in women's percentage of authors published in a conference. The results show that, controlling for year, conference topic substantially predicts author gender composition for a conference. Conference acceptance rate is also weakly associated with women's authorship ($B = 0.07$, $Beta = 0.14$, significant at the 0.001 level). Conferences with more papers accepted were slightly more likely for a greater share of those papers to have women authors. The "Human Factors" topic had the strongest relationship with women's share of authorship ($B = 0.049$, $Beta = 0.314$, significant at 0.001 level). Other top-

^l Averaging across all years for which we have data produced little difference compared with averaging across only the 10 most recent years; 83% of the listed conferences stayed in the same categories.

^m Women's share of authorship increased in ACSC, ADAC, ASPLOS, CCS, CHI, CODES+ISSS, COMM, CPR, DAC, FPCA, GH, HDPC, HT, ICAAD, ICSE, ISLPED, ICSA, ISPD, KDD, LCTES, MICRO, OOPSLA, PACT, PADS, PODS, JCDL-DL, SAC, SCG, SIGCSE, SIGGRAPH, SIGIR, SIGMETRICS, SIGMOD, SIGSOFT, SIGUCCS, SPAA, SPM, STOC, UIST, and WSC; was neutral in VRST, SIGDOC, PLDI, PODC, POPL, SBCCI, SC, MSWIM, ISSAC, ITCSE, IUI, MM, ICS, DOLAP, APL, DATE, DIAL_M, and CIKM; and fell in GIS, DIS, AGENTS, ICFP, SIGAda, SODA. Full conference names for each acronym are available from the authors or online from <http://www.acm.org>.

^k "Large" is in terms of number of authors, not necessarily attendance.

ics significantly and positively correlated with the percentage of women authors were Documentation, Management, and Measurement (with respective values of $B = 0.040, 0.038, 0.030$ and $Beta = 0.122, 0.256, 0.192$, all significant at the 0.001 level). Algorithms was the topic with the strongest negative association with women's share of authorship ($B = -0.033$, $Beta = -0.209$, significant at the .001 level). Other topics with significant negative correlations with percentage of women authors were Performance and Reliability (with respective values of $B = -0.021, -0.036$ and $Beta = -0.144, -0.180$).

These findings lend mild support to the hypothesis that alignment with gender stereotypes predicts the extent of women's authorship. As expected, conferences focused on Human Factors and on Documentation were associated with a greater proportion of women authors, while conferences on Algorithms were associated with a greater proportion of men authors. We found no evidence supporting our hypothesis that Design, Theory, and Security would also be gender skewed.

Implications

The descriptive evidence presented here documents two facts:

Substantial growth. Women's contribution to computing's intellectual life, measured by their share of ACM conference-paper authorship, grew substantially between the late 1960s and 2009, though they remain little more than a quarter of all conference authors; and

Variation across conferences. The extent of women's authorship varies across conferences.

In addition to describing trends, we've taken the first steps toward identifying factors that affect trends in women as authors. We tested whether women's Ph.D.s are related to their authorship and found them to be strongly related, and that the relationship between Ph.D.s and authorship is even stronger for women than for men. This result suggests that women in the pool of eligible authors generally overcome challenges they may face in publishing conference papers.

More interesting, our findings in-



Women were especially evident in conferences devoted to human factors and management and scarce among conferences devoted to algorithms and reliability.



dicating that for each computing Ph.D. holder, there are more women author credits than men author credits in ACM conferences. This finding appears to contradict well-established research results about academic productivity. Many studies have shown that academic men publish more than their women colleagues.^{1,10,11}

The apparent contradiction in productivity results could have several causes. The current study measured the relationship between author credits and likely authors. In contrast, other studies averaged papers published per male or female faculty member.¹ Our per-Ph.D. measure would inflate women's productivity if women authors were more likely than men authors to be without Ph.D.s in a computing discipline or to hold degrees from non-U.S. institutions. Alternatively, men and women might tend to publish in different venues, with women overrepresented at ACM conferences compared to journals, IEEE, and other non-ACM computing conferences. It might also be the case that men more than women hold positions in industry, where publishing is less career-critical than in academe. Finally, computing may be a special case, with conference-paper authorship patterns different from those in other academic disciplines. Each potential explanation calls for more in-depth study and postponing any celebration of women's success in this arena.

Whether or not our finding about disproportionate productivity holds up to further scrutiny, the results still show a clear benefit to the discipline from increasing women's number and representation among Ph.D. degree holders. The benefit is evident in the strong positive association between women Ph.D.s and women's contributions as authors. Every additional woman Ph.D. yields an additional contribution to the intellectual life, as well as to the diversity, of computing. Gender balance among the thought leaders in computing remains a distant goal, but women's educational achievement is moving us all toward that end.

We also considered whether conference topic and prestige predict the variation in women's author-

ship across conferences and found weak supporting evidence. The data showed that, as the stereotypes-hypothesis predicted, some gender differences persist in topics and prestige; for example, women were especially evident in conferences devoted to human factors and management and scarce among conferences devoted to algorithms and reliability. They were also slightly less likely to publish in the more prestigious conferences (measured by paper-acceptance rates).

The dissimilarity in topics on which men and women publish probably reflects gender differences in their thesis topics. Another possible cause is that reviewers are influenced by gender stereotypes if blind review processes are not used. Additional research is needed to investigate potential bias when not using a blind review process. However, regardless of the timing and mechanism, the observed pattern of gender difference is somewhat consistent with gender stereotypes, offering mild support for the hypothesis that gender stereotypes contribute to the segregation of men and women into different computing subfields.

The evidence may have offered only a limited explanation for cross-conference variation due to measurement problems and missing explanatory factors; for example, ACM conference-topic classifications might be a poor measure because it is unclear they accurately represent the dominant themes in a conference. Likewise, our method of categorizing topics as more or less stereotypically masculine or feminine was ad hoc. A more rigorous approach could better test whether an association exists between the conference topic and gender stereotypes. Further investigation is needed to find accurate topic descriptions, stereotypes related to those topics, and empirical links between the two.

Conclusion

Author data documents that the number of ACM conference papers grew from 1966 to 2009, and while women remain severely underrepresented in computing, we found a substantial increase in their share of papers

published, a trend that came about as women earned more Ph.D. degrees in computing. Furthermore, the ratio of Ph.D. holders to papers published indicates that women were relatively more productive authors of ACM conference papers than were men over the same years. Women's contributions, while increasing in most ACM conferences, were greater in some conferences than in others. Gender stereotypes may contribute to this clustering.

This research into women's contributions to computing continues with the support of the National Center for Women & Information Technology (<http://www.ncwit.org/>) and ACM. With ACM journal data and more detailed conference data, investigations are exploring the gender composition of journal authorship and influence. Comparisons of computing with other disciplines where women are well- or underrepresented are also under way. Like this first attempt to track and explain trends and variation, these planned investigations will shed light on the conditions that promote women's participation in computing, and the common benefits derived from their contributions.

Acknowledgements

Thanks to the ACM and the National Center for Women & Information Technology for their support of this project. C

References

1. Abramo, G., D'Angelo, C.A., and Caprasecca, A. Gender differences in research productivity: A bibliometric analysis of the Italian academic system. *Scientometrics* 79, 3 (2009), 517–539.
2. American Federation of Information Processing Societies. *AFIPS Records 1960–1990*; <http://www.cbi.umn.edu/collections/inv/cbi00044.html>
3. Budden, A., Tregenza, T., Aarssen, L., Koricheva, J., Leimu, R., and Lortie, C. Double-blind review favours increased representation of female authors. *Trends in Ecology and Evolution* 23, 1 (2007), 4–6.
4. Charles, M. and Bradley, K. A matter of degrees: Female underrepresentation in computer science programs cross-nationally. In *Women and Information Technology: Research on Underrepresentation*, J.M. Cohoon and W. Aspray, Eds. MIT Press, Cambridge, MA, 2006.
5. Cohoon, J.M. and Aspray, W. *Women and Information Technology: Research on Underrepresentation*. MIT Press, Cambridge, MA, 2006.
6. Cohoon, J.M., Wu, Z., and Luo, L. Will they stay or will they go? *SIGCSE Bulletin* 40, 1 (Feb. 2008), 397–401.
7. Correll, S.J. Gender and the career-choice process: The role of biased self-assessments. *The American Journal of Sociology* 106, 6 (2001), 1691–1730.
8. DiTomaso, N., Post, C., and Parks-Yancy, R. Workforce diversity and inequality: Power, status, and numbers. *Annual Review of Sociology* 33 (2007), 473–501.
9. Faulkner, W. The power and the pleasure? A research agenda for 'making gender stick' to engineers. *Science,*

- Technology & Human Values* 25, 1 (2000), 87.
10. Fox, M.F. Gender, family characteristics, and publication productivity among scientists. *Social Studies of Science* 35, 1 (2005), 131–150.
 11. Fox, M.F. and Mohapatra S. Social-organizational characteristics of work and publication productivity among academic scientists in doctoral-granting departments. *Journal of Higher Education* 78, 5 (2007), 543–571.
 12. Goldberg, P. Are women prejudiced against women? *Society* 5, 5 (1968), 28–30.
 13. Greco, R., Wharton, R., Estelami, H., and Jones, R. The state of scholarly journal publishing 1981–2000. *Journal of Scholarly Publishing* 37, 3 (2006), 155–214.
 14. Greenwald, G. and Krieger, L. Implicit bias: Scientific foundations. *California Law Review* 94, 4 (2006), 945–967.
 15. Kaye, J. Some statistical analyses of CHI. In *Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems* (Boston, Apr. 4–9). ACM Press, New York, 2009, 2585–2594.
 16. Kochan, T., Bezrukova, K., Ely, R., Jackson, S., Joshi, A., Jehn, K., Leonard, J., Levine, D., and Thomas, D. The effects of diversity on business performance: Report of the Diversity Research Network. *Human Resource Management* 42 (2003), 3–21.
 17. Mannix, E. and Neale, M. What differences make a difference? The promise and reality of diverse teams in organizations. *American Psychological Association* 6, 2 (2005), 31–55.
 18. Margolis, J. and Fisher, A. *Unlocking the Clubhouse: Women in Computing*. MIT Press, Cambridge, MA, 2002.
 19. National Center for Women & Information Technology. *Women in IT: The Facts*. Boulder, CO; http://www.ncwit.org/pdf/NCWIT_WomenInITFacts_FINAL.pdf
 20. Page, S.E. *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies*. Princeton University Press, Princeton, NJ, 2007.
 21. Smyth, F.L., Greenwald, A.G., and Nosek, B.A. *Gender Gap in Implicit Gender-Science Associations Is Greatest Among Science Majors* (unpublished manuscript, 2010).
 22. Wajcman, J. *TechnoFeminism*. Polity Press, Cambridge, U.K., 2004.
 23. Wallentin, M. Putative sex differences in verbal abilities and language cortex: A critical review. *Brain and Language* 108 (2009), 175–183.
 24. Whittington, K. and Smith-Doerr, L. Women inventors in context: Disparities in patenting across academia and industry. *Gender and Society* 22, 194 (2008), 194–218.
 25. Williams, K. and O'Reilly, C.A. Demography and diversity in organizations: A review of 40 years of research. In *Research in Organizational Behavior*, B.M. Staw and L.L. Cummings, Eds. JAI, Greenwich, CT, 1998, 77–140.

J. McGrath Cohoon (jmcohoon@virginia.edu) is an associate professor in the Department of Science, Technology, and Society in the School of Engineering and Applied Science at the University of Virginia, Charlottesville, VA.

Sergey Nigai (nigai@kof.ethz.ch) is a Ph.D. candidate in the Department of Management, Technology and Economics at ETH Zurich, Switzerland.

Joseph "Jofish" Kaye (jofish.kaye@nokia.com) is a senior research scientist and ethnographer at Nokia Research Center, Palo Alto, CA.

Algorithmic-based user incentives ensure the trustworthiness of evaluations of Wikipedia entries and Google Maps business information.

BY LUCA DE ALFARO, ASHUTOSH KULSHRESHTHA,
IAN PYE, AND B. THOMAS ADLER

Reputation Systems for Open Collaboration

CONTENT CREATION USED TO be an activity pursued individually or in closed circles of collaborators. Books, encyclopedias, and map collections had either a lone author or a group of authors who knew one another and worked together; it was simply too difficult to coordinate the work of geographically

dispersed groups of people when the main form of communication was letters or telephone. The Internet changed all that, so millions of people around the world are now able to collaborate. The first open-collaboration systems—wikis—focused on text content; the range of content that can be created collaboratively has since expanded to include video editing (such as MetaVid⁵), documents (such as Google Docs,^a Zoho^b), architectural sketching (such as Sketchup^c), and geographical maps (such as OpenStreetMaps¹⁰ and Map-Maker^d).

a <http://docs.google.com>

b <http://www.zoho.com>

c <http://sketchup.google.com>

d <http://www.google.com/mapmaker>

Open collaboration promises immense benefit, as shown by the history of Wikipedia, but is also a challenge to content creators and content

» key insights

- Reputation systems have both a descriptive role, providing information on content and user quality, and a prescriptive role, providing incentives for constructive behavior.
- Content-driven reputation systems derive feedback from analyzing content and interactions; they can scale to very large systems and made resistant to many types of bias.
- Finding “signal in the noise” of crowdsourced content is extremely challenging; a good reputation system allows users to zero in on the best content, spotting vandalism and attacks.

consumers. For content creation, the ability and knowledge of contributors is likely to vary. Collaborative systems open to all will inevitably be subjected to spam, vandalism, and attempts to influence information. How can systems be built to encourage constructive interaction and minimize the consequences of vandalism and spam? How can construction of high-quality information be facilitated? For content consumption, visitors are presented with the outcome of a complex collaboration process. The content may result from weaving together many contributions, with authors usually not known to the visitor and possibly even anonymous. The corollary of “anybody can contribute” is “anybody could have contributed.” How can users judge how much trust to put in the information they see?

Reputation systems can help address these challenges, facilitating both content creation and content consumption. To support this claim, we describe the reputation systems we built at the University of California, Santa Cruz, and at Google for two major collaborative applications: writing articles for Wikipedia and editing business locations on Google Maps.

We describe these systems because they are designed for two well-known cooperative systems and because they represent opposite ends of the reputation-system-design spectrum. The Wikipedia reputation system WikiTrust relies on a chronological analysis of user contributions to articles, metering positive or negative increments of reputation whenever new contributions are made. Users obtain new identities at will, and there is no “ground truth” against which their contributions are compared. The reputation mechanism can be explained in simple terms to users while helping provide an incentive to provide good-quality contributions. The Google Maps system Crowdsensus compares the information provided by users on map business listings and computes both a likely reconstruction of the correct listing and a reputation value for each user. Unlike WikiTrust, users have a stable identity in the system, and their contributions can be compared with the ground truth of the real world, if desired. The reputation system oper-



Even for successful sites, establishing a community of dedicated users and accumulating sufficient high-quality feedback can take years.



ates largely in the background, working not chronologically but by iteratively refining joint estimates of user reputations and listing values.

Content-driven vs. user-driven reputation. Wikipedia and Maps are both content-driven, relying on automated content analysis to derive user and content reputations. Reputation systems (such as the eBay system for sellers and buyers and the Amazon and NewEgg systems of product reviews and ratings) are user-driven, based on explicit user feedback and ratings.

Content-driven systems derive their feedback from an analysis of all interactions and, consequently, get feedback from all users uniformly. User-driven systems often suffer from selection bias, as users who are particularly happy or unhappy are more likely to provide feedback or ratings. Moreover, in user-driven systems, users might do one thing and say another. Sellers and buyers might give each other high ratings simply to obtain high ratings in return, regardless of how they feel about the transaction.⁷ Content-driven reputation systems derive user feedback from user actions and can be more resistant to manipulation.⁴

Deploying user-driven and content-driven reputation systems presents different challenges. The success of a user-driven system depends on user feedback. Even for successful sites, establishing a community of dedicated users and accumulating sufficient high-quality feedback can take years. However, when useful feedback is extracted automatically from user interactions and data, content-driven reputation systems can deliver results immediately.

On the other hand, the algorithmic nature of content-driven reputation systems can play against their success, preventing users from understanding and trusting the reputation values they generate. Reading “Product A received 25 positive, 12 neutral, and two negative votes,” users understand the meaning and often trust (to some extent) the result, in spite of possible selection bias of voting users and manipulation schemes by malicious users. When an algorithm produces the answer for a Wikipedia page “This sentence has reputation 4 out of a maximum of 10,” users typically wonder how the repu-

tation is computed and question the appropriateness of the algorithms. In reputation systems that make reputation values available to users, simpler is often better, even when the performance, in numerical terms, is worse; users need to understand the origin of reputation to be able to trust it.^{6,7}

WikiTrust and Crowdsensus are examples of content-driven reputation systems; others analyze the wording of consumer reviews to extract reviewer and product reputation^{12,15} and compute Wikipedia content reputation.¹⁴ The algorithms PageRank¹³ and HITS¹¹ constitute content-driven reputation systems for ranking Web pages. Beyond the Web, consumer-credit-rating agencies are an example of a content-driven reputation system in the financial world.

WikiTrust

We developed WikiTrust,^e a reputation system for wiki authors and content, to provide an incentive to give quality contributions to Wikipedia and indicate to Wikipedia visitors the quality of content. WikiTrust employs two reputation systems, one for users and one for content. Users gain reputation when they make edits that are preserved by subsequent authors and lose reputation when their work is partly or wholly undone. Text starts with no reputation and gains reputation when revised by high-reputation authors; text can lose reputation when disturbed by edits. While WikiTrust was designed for wikis, its principles are applicable to any content-management system in which the content evolves through a sequence of revisions, provided the difference between revisions is measurable.

WikiTrust is available via a Firefox browser extension. When a user visits a page of one of several Wikipedias, the browser extension displays an additional WikiTrust tab alongside the standard wiki tabs (such as edit and history). When users click on the WikiTrust tab, the extension contacts the back-end servers to obtain text-reputation information, which is visualized via the text background color: perfect-reputation text appears on a white background, and the back-

ground turns a darker shade of orange as the reputation of the text gets lower. The text color alerts viewers to content that might have been tampered with (see Figure 1). WikiTrust does not currently display user reputations, due to our desire not to alter the social experience of contributing to Wikipedia.

User reputation. User reputations are computed according to the quality and quantity of their contributions. A contribution is considered of good quality if the change it introduced is preserved in subsequent revisions.^{2,3,8} To evaluate the quality of a contribution that produced a revision b , WikiTrust compares b with two reference points: a past revision a and a future revision c . From the point of view of c , if b is closer than a , then the author of b did good work, since that author made changes that made the page more like how it will be in the future revision c (see Figure 2a). On the

other hand, if b is farther away from c than a was, the change from a to b was not preserved in c (see Figure 2b). To capture this intuition, we define the quality $q(b \mid a, c)$ of b with respect to a and c as the amount of improvement $d(a, c) - d(b, c)$ divided by the amount of work $d(a, b)$ involved in creating b . If the distance d satisfies the triangular inequality, then $q(b \mid a, c)$ falls somewhere between -1 and $+1$; it is equal to -1 if $a = c$ (so the change $a \rightarrow b$ was entirely reverted) and equal to $+1$ if the change $a \rightarrow b$ was entirely preserved.

Authors start with a small amount of reputation. When a new revision c is produced, it is used to judge the quality of several preceding revisions b , using as reference point revisions a that are either not too far in time from b and c or are by authors with solid reputations.⁴ For each such triple considered, the reputation of the author of b is increased by the amount $q(b \mid a, c)$.

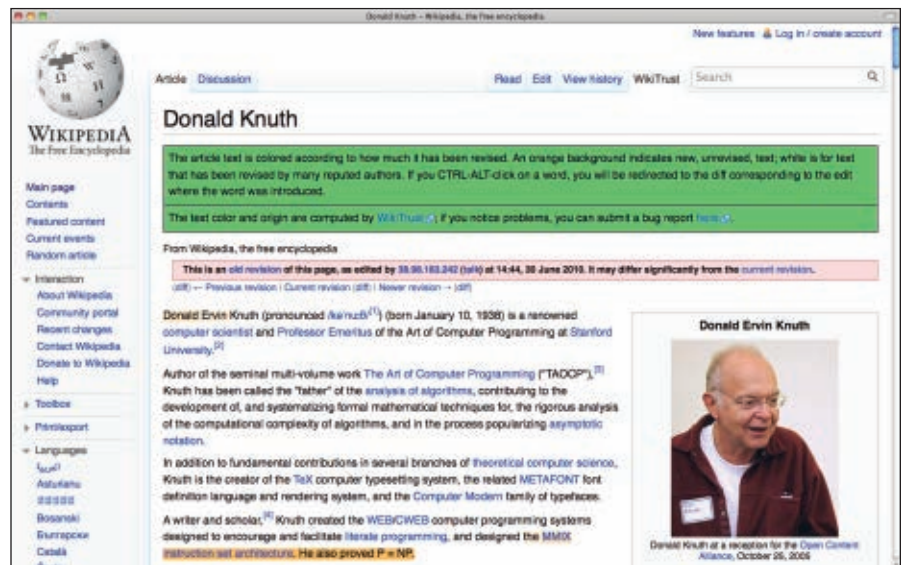
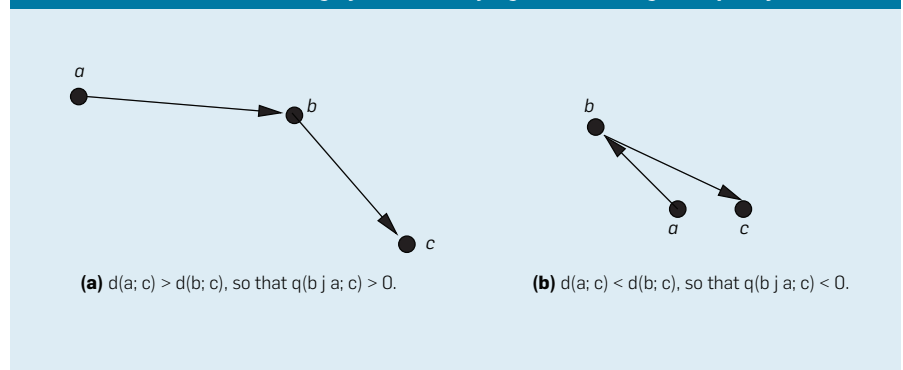


Figure 1. The Wikipedia page for Donald Knuth, as rendered by WikiTrust; the text background is a darker shade of orange the lower the reputation of the text.

Figure 2. A revision, as in 2a, which brings b closer to c , is judged to be of positive quality; a revision, as in 2b, which is largely reverted, is judged to be of negative quality.



^e <http://www.wikitrust.net>

$c) \cdot \log(1 + r_c)$, where r_c is the reputation of the author of c . The dependence of the increment on the reputation of c 's author ensures the judgment of higher-reputation authors carries more weight. A linear dependence would lead to an oligarchy in which longtime good-reputation users have overwhelming influence over new users, while new users can give no significant feedback in return. Assigning all users the same influence would lead to a completely democratic system, which would not be ideal in wikis, as the reputations of good users who participated in reversion wars with vandals would be too much at risk. The logarithmic factor balances oligarchy and democracy.

Users judge other users via their actions (their edits) and are thus liable to be judged in turn, making the system resistant to manipulation. For instance, the only way user A can damage the reputation of user B is by reverting user B 's edits. However, if subsequent users reinstate B 's edits, A 's reputation suffers most, as B 's contribution is longer-lived than A 's.

When developing a reputation system, it is essential for its designers to be able to evaluate its performance quantitatively; otherwise, it is impossible to tune the system or compare different algorithms. A powerful evaluation criterion is whether reputation can help predict the quality of future user contributions.² This is a tough test; reputation is not only a badge earned through past work but an indicator of future behavior. If low-reputation users were as likely as high-reputation users to do good work, why pay attention to user reputation in the first place?

To evaluate the WikiTrust user-

reputation system, we measured the precision and recall with which low-reputation can predict reversions.² For each revision b , we say the author of b has low reputation if his reputation is in the bottom 20% and that b has been reverted if its average quality is below -0.8 . This is a proper evaluation, since the reputation of b 's author depends only on the author's behavior before b , whereas b 's quality depends on how b will be judged by future revisions (see Table 1). The recall is high, indicating high-reputation authors are unlikely to be reverted; the precision is lower because many novice authors make good-quality contributions. In measuring precision and recall, the system weighs each contribution according to the number of words added and deleted. The data is based on Wikipedia dumps ending in late 2009, except for the English Wikipedia, where the dump is from January 2008, augmented with updates through January 2010 for the 30,000 pages of the Wikipedia 0.7 project.^f

Content reputation. WikiTrust aims to signal the quality of Wikipedia content, alerting visitors to possible vandalism and content tampering. To this end, WikiTrust computes and displays the reputation of Wikipedia content at the granularity of individual words; see Figure 1 for an example in which the assertion on NP-completeness, having earned low reputation, is highlighted in orange.

WikiTrust computes content reputation according to the extent the content has been revised and the reputations of the users who revised it.^{1,14}

^f http://en.wikipedia.org/wiki/Wikipedia:Version_1.0_Editorial_Team

When a new revision is created, the text affected directly by the edit is assigned a small fraction of the revision author's reputation. The text left unchanged gains reputation, with the idea that the author, by leaving it unchanged, has implicitly expressed approval for it. The same idea can be applied to many types of content; all the system must do is identify when an edit occurs, what content is new or directly affected by the edit (it receives a fraction of the author's reputation), and what content is unaffected and thus implicitly validated (it may gain reputation).

WikiTrust also makes the content-reputation system difficult to subvert. Since it is possible to alter the content of sentences by inserting new text, as well as by rearranging or deleting text, WikiTrust ensures that each such action leaves a mark on the content. Furthermore, the algorithm allows users to raise text reputation up to only their own reputation, so low-reputation users cannot erase the low-reputation marks they leave behind with more activity. To ensure that a single high-reputation user gone rogue cannot arbitrarily raise the reputation of text via repeated edits, the system associates with each individual word the identities of the last few users who raised the word's reputation and prevent users whose identity is associated with a word from again raising the word's reputation. The resulting content reputation system includes two properties:

Revisions. Content reputation is an indication of the extent to which content has been revised and of the reputation of the users who revised it; and

Consensus. High content reputation requires consensus, achieved only as a result of the approval of multiple distinct high-reputation users.

Evaluation. The system's predictive ability is a measure of its performance. Higher-quality content should be less likely to be deleted in future revisions. This evaluation is imperfect, as it disregards the fact that our content-reputation system aims not only for predictive value but warning value with respect to unrevised, possibly malicious edits. Our 2008 analysis of 1,000 articles selected at random from English Wikipedia articles with at least 200 revisions produced the following results¹:

Recall of deletions. Only 3.4% of the

Table 1. Predictive ability of the WikiTrust user-reputation system on various language editions of Wikipedia.

Wikipedia	Precision	Recall
Dutch	58.1	95.6
English	58.0	77.1
French	43.7	89.1
German	50.4	93.4
Polish	43.1	91.7
Portuguese	48.3	94.1

content was in the lower half of the reputation range, yet this percentage corresponds to 66% of the text deleted from one revision to the next;

Precision of deletions. Text in the lower half of the reputation range had a probability of 33% of being deleted in the very next revision, unlike the 1.9% probability for general text. The deletion probability increases to 62% for text in the bottom 20% of the reputation range; and

Reputation as predictor of content longevity. Top reputation words had an expected lifespan 4.5 times longer than words with bottom reputation.

Lessons learned. The original reputation system described in Adler and de Alfaro² was open to many attacks that allowed users to gain reputation while doing no useful work or worse, damaging the system. For instance, users could gain reputation by first vandalizing a revision using an alternate “sacrificial” identity, then undoing the vandalism using their main identity. As we believed these attacks could have crippled the system, we took pains to prevent them before making it available to the public.⁴ However, neither users nor researchers providing us with feedback expressed concern over the robustness of the original design. We suspect the system would have been more successful by making WikiTrust available earlier and dealing with the security issues later, adopting the common (if hardly principled) approach of “security as an afterthought.”

There was much interest in how the system measured contribution quality. Early in its development, we realized that if we relied on a standard edit distance between revisions, users whose contributions were later reworded would sometimes lose reputation in spite of their good work. We solved this by adopting an edit distance that accounts for block moves and differentiates between word insertions and deletions, which are both given a weight of 1, and word replacements, which are given a weight of 12; under this edit distance, authors of reworded contributions still receive partial credit for their work. We were sure our choice of edit distance would remain an obscure detail buried in the codebase. However, we found ourselves explaining it many times to Wikipedia

contributors; users care deeply how their reputation is computed, even when reputation is not displayed to anyone. Perceived fairness is a very important quality of a reputation system.

Design Space

Table 2 outlines the design space for reputation systems for collaborative content. The first distinction focuses on the signals used for computing reputation; are they derived from explicit user feedback or inferred algorithmically from system events? The two types of systems can work side by side; for instance, sale and product-return information can be used to compute NewEgg product ratings, and WikiTrust users were given the option of voting explicitly for the correctness of Wikipedia revisions.

The second distinction concerns the visibility of the reputation system to the users. Many systems are useful even if they work behind the scenes, used to rank content, prevent abuse, fight spam, and more. Examples are Web-content ranking algorithms (such as PageRank¹³ and HITS¹¹). Reputation systems working behind the scenes can make use of any signals available on users and content and use advanced algorithms and techniques (such as machine learning). On the other hand, if the goal of the system is to influence user behavior, its existence and the reputation values it computes must be revealed to the users. It is thus important that users are able to form some idea of how the reputation values are computed; they want to know the metrics used to judge them, and systems that cannot be understood are typically considered arbitrary, capricious, unfair, or downright evil.

The strength of the identity system is also a relevant factor in the design of reputation systems. In systems with weak identity, new users must be assigned the same amount of reputation as bad users. There is no benefit of the doubt; if new users could enjoy a reputation above the minimum, bad users could simply use a new identity whenever their reputation fell below that of new users.

The next distinction concerns the existence of a ground truth to which content should correspond in order to have perfect quality. No such ground

truth exists for Wikipedia articles, which do not converge to a canonical form as they are edited but rather continually evolve as content is added and refined. For any map-business listings, such ground truth exists for many information fields; for example, there is one (or a few) correct values for the telephone number of each business, and in the eBay seller-rating system, it can be usefully assumed that each seller has intrinsic “honesty,” with buyer feedback processed to estimate honesty. The latter highlights how the existence of a ground truth matters not so much because data can be compared to the ground truth (often expensive or impossible) but because the assumption that a ground truth exists affects the type of algorithms that can be used.

Finally, reputation algorithms span a spectrum from chronological to global. At one extreme, purely chronological algorithms consider the stream of actions on the systems, including contributions and comments, and for each action update the reputations of the participating users. The eBay reputation system is chronological, as is WikiTrust. At the other end of the spectrum are reputation systems based on global algorithms that operate at once on the whole network of recommendations, generally in batch mode.

Table 2. Design space for reputation systems for collaborative content.

User-driven vs. content-driven.

User-driven reputation systems rely on ratings provided by users; content-driven systems rely on the algorithmic analysis of content and user interactions.

Visible to users?

Are users aware of the existence of the reputation system?

Weak vs. strong identity.

How readily can users acquire a new identity in the system?

Existence of ground truth.

Is there a ground truth to which users expect the content to converge, assuming users are truthful?

Chronological vs. global reputation updates.

Chronological algorithms consider system activity in the order it occurs; global algorithms consider the whole system and typically operate in batch mode.

Each type of algorithm has advantages. Global algorithms can make use of the information in the graph topology; an example is the way PageRank and HITS propagate reputation along edges.^{11,13} However, global algorithms may require more computational resources, as they need to consider the whole system at once. Chronological algorithms can leverage the asymmetry between past and future to prevent attacks. In a chronological reputation system, new identities, including fake ones used for attacks, are assigned an initial reputation lower than that of established users. By making it difficult for users to gain reputation from users who are themselves of low reputation, WikiTrust is able to prevent many types of Sybil attacks.⁴

Crowdsensus

To illustrate how the characteristics of the design space could influence the structure of a reputation system, consider Crowdsensus, a reputation system we built in 2009 at Google to analyze user edits to Google Maps. Users can edit business listings on Google Maps, providing values for the title, phone, Web site, physical address, location, and business categories. The goal is to measure the accuracy of users contributing information and reconstruct insofar as is possible correct listing information for the businesses.

The design space of a reputation system for editing Google Maps business listings differs in several respects from the design space of a Wikipedia reputation system. First, for each business listing, there is, at least, in first approximation a ground truth; ideally, each business has exactly an appropriate phone number, Web site, and so forth. Reality is more complex, as there are businesses with multiple equivalent phone numbers, alternative Web sites, and more. Nevertheless, for our purpose here, we consider the simpler setting in which every listing attribute has exactly one correct value. We note, too, that it might be expensive to check the ground truth for each business listing, and, in the worst case, it might require sending someone on site. Crowdsensus does not require actually checking the ground truth; it simply relies on the existence of a ground truth, and user



Reputation systems are the online equivalent of the body of laws regulating the real-world interaction of people.



reputation is not visible to the users. Consequently, users need not understand the details of how reputation is computed, making it possible to use advanced algorithms and techniques to determine the correctness of the information. The identity notion is stronger in Google Maps than in Wikipedia; in particular, it is a practical nuisance for established users of Google products to open and use separate accounts for Maps editing. And the ample computational resources available at Google enable us to consider global reputation systems, in addition to chronological ones.

These considerations led to our design for Crowdsensus that is very different from WikiTrust. The input to Crowdsensus is a sequence of statements that are triples of the form (u, a, v) , meaning user u asserts that attribute a of some business has value v . Thus, Crowdsensus is set to solve what is called a “collective revelation problem,”⁹ even though some of the instruments by which such problems are solved (such as monetary payoffs and elaborate ways of revealing a user’s information) are not available in Crowdsensus. Crowdsensus is structured as a fixpoint graph algorithm; the vertices of the graph are the users and business attributes. For each statement (u, a, v) , we insert an edge, from u to a labeled by v and an edge from a back to u . Crowdsensus associates to each user vertex u a truthfulness value q_u , representing the probability that u is telling the truth about the values of attributes; this value is initially set to an a priori default, then estimated iteratively.

Crowdsensus computation is structured in a series of iterations. At the beginning of each iteration, user vertices send their truthfulness value to the attributes. Each attribute vertex thus receives the list $(q_1, v_1), \dots, (q_n, v_n)$ consisting of the values $v_1, \dots, v_n$ proposed for the attribute, along with the (estimated) truthfulness $q_1, \dots, q_n$ of the user who proposed them. An attribute inference algorithm is then used to derive a probability distribution over the proposed values $v_1, \dots, v_n$. Crowdsensus then sends to each user vertex u_i the estimated probability that v_i is correct; on this basis, a truthfulness-inference algorithm estimates the user’s truthfulness, concluding the iteration.

tion. The algorithm employs multiple iterations, so the information about a user's truthfulness gained from some statements is able to propagate to other statements.

The attribute inference algorithm is the heart of Crowdsensus. We first used standard algorithms (such as Bayesian inference) but quickly realized they were suboptimal for the real case of Google Maps. First, users lack independent information on the correct value of attributes. Users typically have only a few ways to learn, say, the phone number of a restaurant; they can go there and ask or read it on a coupon, but 100 users providing us a phone number does not correspond to 100 independent ways of learning a phone number. We had to develop algorithms that could account for this lack of independence. Business attributes have different characteristics, and we found it very important to develop attribute-inference algorithms tailored to every type of attribute; for example, geographical positions (expressed as latitude-longitude pairs) have a natural notion of proximity (a distance), and it is essential to make use of it in the inference algorithms. Web sites also involve some notion of distance, at least insofar as two sites might belong to the same domain. Our implementation of Crowdsensus employs different inference algorithms for different types of attributes. The complete system is more complex in several respects, containing algorithms for attributes with multiple correct values, dealing with spam, and protecting the system from abuse. Furthermore, the Google Maps data pipeline comprises several interdependent algorithms and subsystems, and we designed Crowdsensus as one of the many components of the overall pipeline.

To illustrate the Crowdsensus algorithm, consider the case of N users and M attributes; the true value of each attribute is chosen uniformly at random from a set of K possible values. For each user u , we choose a probability p_u uniformly at random in the $[0, 1]$ interval; user u provides, with probability p_u , the correct attribute value and probability $1 - p_u$, a value selected uniformly at random from the K possible values. Note that Crowdsensus is not informed of

the probability p_u of a user u ; rather, Crowdsensus computes the truthfulness q_u for u from the statements by u . For simplicity, we assume that for each attribute, J estimates are provided by J users selected at random.

We experimented using a standard Bayesian inference for attribute values. For $M = 1,000$, $N = 100$, $K = 10$, and $J = 10$, Crowdsensus has an error rate in the reconstruction of the correct value of each feature of 2.8%. In contrast, a (noniterative) algorithm that performs Bayesian inference without using information on user reputation has an error rate of 7.9%. The roughly threefold reduction in error rate, from 7.9% to 2.8%, is due to the power of user reputation in steering the inference process. The statistical correlation between the true truthfulness p_u and the reconstructed truthfulness q_u over all users was 0.988, indicating Crowdsensus was able to precisely reconstruct the user's truthfulness. If we take $J = 5$, the error rate of Crowdsensus is 12.6%, compared with an error rate of 22% for standard Bayesian inference; the correlation between true and inferred truthfulness is 0.972.

Conclusion

We conclude on a note of optimism concerning the role of reputation systems in mediating online collaboration. Such systems are the online equivalent of the body of laws regulating the real-world interaction of people. As a larger fraction of people's productive lives are carried out through online, computer-mediated interaction, we expect development of such an online body of algorithmic legislation to be a rich field of work and research, with important implications for society overall.

Acknowledgements

This work was supported in part by the Center for Information Technology Research in the Interest of Society and the Institute for Scalable Scientific Data Management (<http://institute.lanl.gov/isti/issdm/>). We thank Shelly Spearing and Scott Brandt for their enthusiastic support. 

References

1. Adler, B., Chatterjee, K., de Alfaro, L., Faella, M., Pye, I., and Raman, V. Assigning trust to Wikipedia

content. In *Proceedings of the WikiSym International Symposium on Wikis* (Porto, Portugal, Sept. 8–10). ACM Press, New York, 2008.

2. Adler, B. and de Alfaro, L. A content-driven reputation system for the Wikipedia. In *Proceedings of the 16th International World Wide Web Conference*. ACM Press, New York, 2007.
3. Adler, B., de Alfaro, L., Pye, I., and Raman, V. Measuring author contributions to the Wikipedia. In *Proceedings of the WikiSym International Symposium on Wikis* (Banff, Alberta, Canada, May 8–12). ACM Press, New York, 2008.
4. Chatterjee, K., de Alfaro, L., and Pye, I. Robust content-driven reputation. In *Proceedings of the First ACM Workshop on Artificial Intelligence and Security* (Alexandria, VA, Oct. 27). ACM Press, New York, 2008.
5. Dale, M., Stern, A., Deckert, M., and Sack, W. System demonstration: Metavid.org: A social Website and open archive of congressional video. In *Proceedings of the 10th International Conference on Digital Government Research: Social Networks: Making Connections Between Citizens, Data, and Government* (Puebla, Mexico, May 17–21). Digital Government Society of North America, 2009, 309–310.
6. Dellarocas, C. The digitization of word-of-mouth: Promises and challenges of online reputation systems. *Management Science* 49, 10 (Oct. 2003), 1407–1424.
7. Dellarocas, C., Fan, M., and Wood, C.A. *Self-Interest, Reciprocity, and Participation in Online Reputation Systems*. Technical Report Paper 205, Center for eBusiness, Sloan School of Management, MIT, Cambridge, MA, 2005.
8. Druck, G., Miklau, G., and McCallum, A. Learning to predict the quality of contributions to Wikipedia. In *Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence* (Chicago, July 13–14). AAAI Press, 2008.
9. Goel, S., Reeves, D.M., and Pennock, D.M. Collective revelation: A mechanism for self-verified, weighted, and truthful predictions. In *Proceedings of the 10th ACM Conference on Electronic Commerce* (Stanford, CA, July 6–8). ACM Press, New York, 2009, 265–274.
10. Haklay, M. and Weber, P. OpenStreetMap: User-generated streetmaps. *Pervasive Computing* 7, 4 (Oct.–Dec. 2008), 12–18.
11. Kleinberg, J.M. Authoritative sources in a hyperlinked environment. *Journal of the ACM* 46, 5 (Sept. 1999), 604–632.
12. Liu, Y., Huang, X., An, A., and Yu, X. Helpmeter: A nonlinear model for predicting the helpfulness of online reviews. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (Sydney, Australia, Dec. 9–12). IEEE Computer Society Press, Washington, D.C., 2008, 793–796.
13. Page, L., Brin, S., Motwani, R., and Winograd, T. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report. Stanford Digital Library Technologies Project, Stanford, CA, 1998.
14. Zeng, H., Alhoussaini, M.A., Ding, L., Fikes, R., and McGuinness, D.L. Computing trust from revision history. In *Proceedings of the International Conference on Privacy, Security and Trust* (Markham, Ontario, Canada, Oct. 30–Nov. 1). ACM Press, New York, 2006.
15. Zhang, Z. Weighing stars: Aggregating online product reviews for intelligence e-commerce applications. *IEEE Intelligent Systems* 23, 5 (Sept. 2008), 42–49.

Luca de Alfaro (luca@dealfaro.org) is a professor of computer science at the University of California, Santa Cruz, and was an engineer and research scientist at Google 2008–2011.

Ashutosh Kulshreshtha (ashu@google.com) is technical lead for Crowdsensus at Google Inc., Mountain View, CA.

Ian Pye (ian@cloudflare.com) is a software engineer at CloudFlare, Inc., San Francisco, and Ph.D. candidate at the University of California, Santa Cruz, working on the WikiTrust project.

B. Thomas Adler (thumper@alumni.caltech.edu) is a software engineer at Facebook, Inc., Palo Alto, CA, and Ph.D. candidate at the University of California, Santa Cruz, working on the WikiTrust project.

BI technologies are essential to running today's businesses and this technology is going through sea changes.

BY SURAJIT CHAUDHURI, UMESHWAR DAYAL,
AND VIVEK NARASAYYA

An Overview of Business Intelligence Technology

BUSINESS INTELLIGENCE (BI) SOFTWARE is a collection of decision support technologies for the enterprise aimed at enabling knowledge workers such as executives, managers, and analysts to make better and faster decisions. The past two decades have seen explosive growth, both in the number of products and services offered and in the adoption of these technologies by industry. This growth has been fueled by the declining cost of acquiring and storing very large amounts of data arising from sources such as customer transactions in banking, retail as well as in e-businesses, RFID tags for inventory tracking, email, query logs for Web sites, blogs, and product reviews. Enterprises today collect data at a finer granularity, which is therefore of much larger volume. Businesses are leveraging their data asset

aggressively by deploying and experimenting with more sophisticated data analysis techniques to drive business decisions and deliver new functionality such as personalized offers and services to customers. Today, it is difficult to find a successful enterprise that has not leveraged BI technology for its business. For example, BI technology is used in manufacturing for order shipment and customer support, in retail for user profiling to target grocery coupons during checkout, in financial services for claims analysis and fraud detection, in transportation





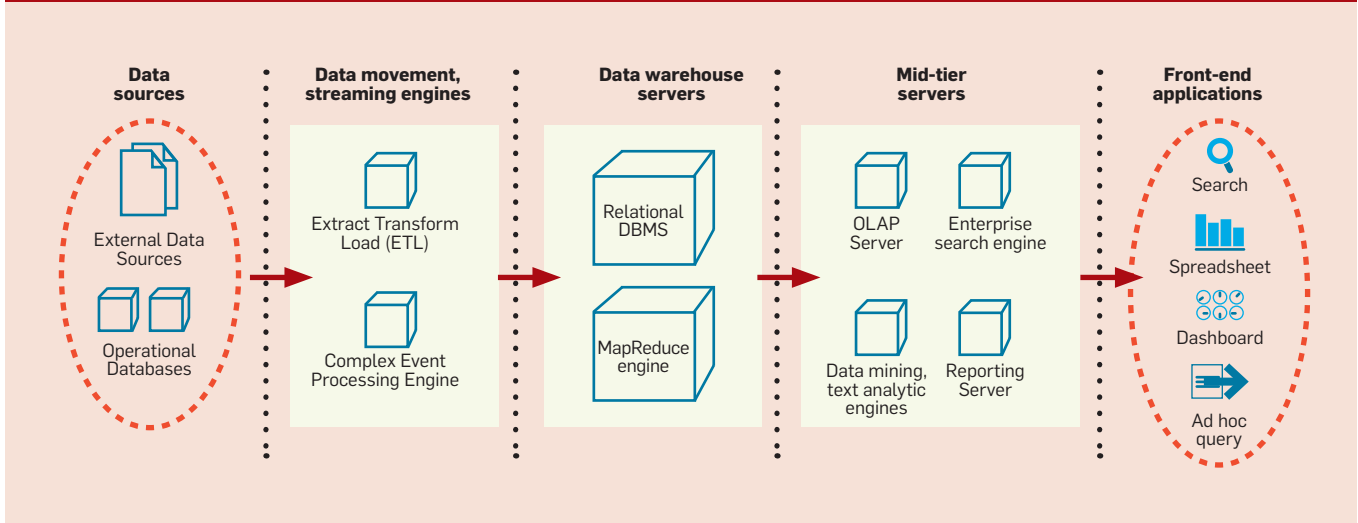
for fleet management, in telecommunications for identifying reasons for customer churn, in utilities for power usage analysis, and health care for outcomes analysis.

A typical architecture for supporting BI within an enterprise is shown in Figure 1 (the shaded boxes are technology that we focus on in this article). The data over which BI tasks are performed often comes from different sources—typically from multiple operational databases across departments within the organization, as well as external vendors. Different sources contain data

of varying quality, use inconsistent representations, codes, and formats, which have to be reconciled. Thus the problems of integrating, cleansing, and standardizing data in preparation for BI tasks can be rather challenging. Efficient data loading is imperative for BI. Moreover, BI tasks usually need to be performed incrementally as new data arrives, for example, last month's sales data. This makes efficient and scalable data loading and refresh capabilities imperative for enterprise BI. These back-end technologies for preparing the data for BI are collectively

» key insights

- **The cost of data acquisition and data storage has declined significantly. This has increased the appetite of businesses to acquire very large volumes in order to extract as much competitive advantage from it as possible.**
- **New massively parallel data architectures and analytic tools go beyond traditional parallel SQL data warehouses and OLAP engines.**
- **The need to shorten the time lag between data acquisition and decision making is spurring innovations in business intelligence technologies.**

Figure 1. Typical business intelligence architecture.

referred to as Extract-Transform-Load (ETL) tools. Increasingly there is a need to support BI tasks in near real time, that is, make business decisions based on the operational data itself. Specialized engines referred to as Complex Event Processing (CEP) engines have emerged to support such scenarios.

The data over which BI tasks are performed is typically loaded into a repository called the *data warehouse* that is managed by one or more data warehouse servers. A popular choice of engines for storing and querying warehouse data is relational database management systems (RDBMS). Over the past two decades, several data structures, optimizations, and query processing techniques have been developed primarily for executing complex SQL queries over large volumes of data—a key requirement for BI. An example of such an ad hoc SQL query is: find customers who have placed an order during the past quarter whose amount exceeds the average order amount by at least 50%. Large data warehouses typically deploy parallel RDBMS engines so that SQL queries can be executed over large volumes of data with low latency.

As more data is born digital, there is increasing desire to architect low-cost data platforms that can support much larger data volume than that traditionally handled by RDBMSs. This is often described as the “Big Data” challenge. Driven by this goal, engines based on the *MapReduce*⁹ paradigm—originally built for analyzing Web documents and Web search query logs—are now being targeted for enterprise analyt-

ics. Such engines are currently being extended to support complex SQL-like queries essential for traditional enterprise data warehousing scenarios.

Data warehouse servers are complemented by a set of *mid-tier servers* that provide specialized functionality for different BI scenarios. Online analytic processing (OLAP) servers efficiently expose the *multidimensional* view of data to applications or users and enable the common BI operations such as filtering, aggregation, drill-down and pivoting. In addition to traditional OLAP servers, newer “in-memory BI” engines are appearing that exploit today’s large main memory sizes to dramatically improve performance of multidimensional queries. *Reporting servers* enable definition, efficient execution and rendering of reports—for example, report total sales by region for this year and compare with sales from last year. The increasing availability and importance of text data such as product reviews, email, and call center transcripts for BI brings new challenges. *Enterprise search engines* support the keyword search paradigm over text and structured data in the warehouse (for example, find email messages, documents, history of purchases and support calls related to a particular customer), and have become a valuable tool for BI over the past decade. *Data mining engines* enable in-depth analysis of data that goes well beyond what is offered by OLAP or reporting servers, and provides the ability to build predictive models to help answer questions such as: which existing customers are

likely to respond to my upcoming catalog mailing campaign? *Text analytic engines* can analyze large amounts of text data (for example, survey responses or comments from customers) and extract valuable information that would otherwise require significant manual effort, for example, which products are mentioned in the survey responses and the topics that are frequently discussed in connection with those products.

There are several popular front-end applications through which users perform BI tasks: spreadsheets, enterprise portals for searching, performance management applications that enable decision makers to track key performance indicators of the business using visual dashboards, tools that allow users to pose ad hoc queries, viewers for data mining models, and so on. Rapid, ad hoc *visualization* of data can enable dynamic exploration of patterns, outliers and help uncover relevant facts for BI.

In addition, there are other BI technologies (not shown in Figure 1) such as *Web analytics*, which enables understanding of how visitors to a company’s Web site interact with the pages; for example which landing pages are likely to encourage the visitor to make a purchase. Likewise, vertical packaged applications such as customer relationship management (CRM) are widely used. These applications often support built-in analytics, for example, a CRM application might provide functionality to segment customers into those most likely and least likely to repurchase a particular product.

Another nascent but important area is *mobile BI* that presents opportunities for enabling novel and rich BI applications for knowledge workers on mobile devices.

In this short article, we are not able to provide comprehensive coverage of all technologies used in BI (see Chaudhuri et al.⁵ for additional details on some of these technologies). We therefore chose to focus on technology where research can play, or has historically played, an important role. In some instances, these technologies are mature but challenging research problems still remain—for example, data storage, OLAP servers, RDBMSs, and ETL tools. In other instances, the technology is relatively new with several open research challenges, for example, MapReduce engines, near real-time BI, enterprise search, data mining and text analytics, cloud data services.

Data Storage

Access structures. Decision support queries require operations such as filtering, join, and aggregation. To efficiently support these operations, special data structures (not typically required for OLTP queries) have been developed in RDBMSs, described here. Access structures used in specialized OLAP engines that do not use RDBMSs are discussed later.

Index structures. An index enables associative access based on values of a particular column. When a query has one or more filter conditions, the selectivities of these conditions can be exploited through *index scans* (for example, an index on the StoreId column can help retrieve all sales for StoreId = 23) and *index intersection* (when multiple conditions exist). These operations can significantly reduce, and in some cases eliminate, the need to access the base tables, for example, when the index itself contains all columns required to answer the query. *Bitmap* indexes support efficient index operations such as union and intersection. A bitmap index on a column uses one bit per record for each value in the domain of that column. To process a query of the form *column1 = val1 AND column2 = val2* using bitmap indexes, we identify the qualifying records by taking the bit-wise AND of the respective bit vectors. While such representations are very



Today, it is difficult to find a successful enterprise that has not leveraged BI technology for their business.

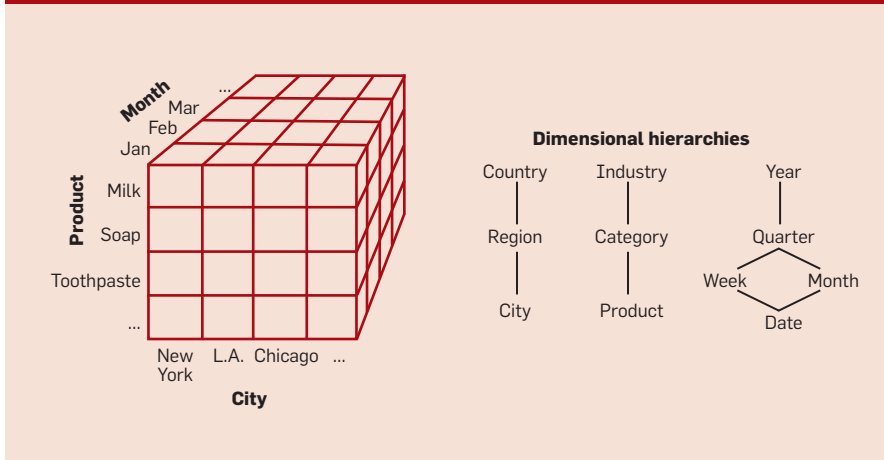


effective for low cardinality domains (for example, gender), they can also be used for higher cardinality domains using bitmap compression.

Materialized views. Reporting queries often require summary data, for example, aggregate sales of the most recent quarter and the current fiscal year. Hence, precomputing and materializing summary data (also referred to as materialized views) can help dramatically accelerate many decision support queries. The greatest strength of a materialized view is its ability to specifically target certain queries by effectively caching their results. However this very strength also can limit its applicability, that is, for a slightly different query it may not be possible to use the materialized view to answer that query. This is in contrast to an index, which is a much more general structure, but whose impact on query performance may not be as dramatic as a materialized view. Typically, a good physical design contains a judicious mix of indexes and materialized views.

Partitioning. Data partitioning can be used to improve both performance (discussed later) and manageability. Partitioning allows tables and indexes to be divided into smaller, more manageable units. Database maintenance operations such as loading and backup can be performed on partitions rather than an entire table or index. The common types of partitioning supported today are hash and range. Hybrid schemes that first partition by range followed by hash partitioning within each range partition are also common.

Column-oriented storage. Traditional relational commercial database engines store data in a row-oriented manner, that is, the values of all columns for a given row in a table are stored contiguously. The Sybase IQ product³⁰ pioneered the use of column-oriented storage, where all values of a particular *column* are stored contiguously. This approach optimizes for “read-mostly” workloads of ad hoc queries. The column-oriented representation has two advantages. First, significantly greater data compression is possible than in a row-oriented store since data values within a column are typically much more repetitive than across columns. Second, only the columns accessed in the query need to be scanned. In con-

Figure 2. Multidimensional data.

trast, in a row-oriented store, it is not easy to skip columns that are not accessed in the query. Together, this can result in reduced time for scanning large tables.

Finally, we note that in the past decade, major commercial database systems have added automated physical design tools that can assist database administrators (DBAs) in choosing appropriate access structures (see Chaudhuri and Narasayya⁷ for an overview) based on *workload* information, such as queries and updates executed on the system, and constraints, for example, total storage allotted to access structures.

Data Compression can have significant benefits for large data warehouses. Compression can reduce the amount of data that needs to be scanned, and hence the I/O cost of the query. Second, since compression reduces the amount of storage required for a database, it can also lower storage and backup costs. A third benefit is that compression effectively increases the amount of data that can be cached in memory since the pages can be kept in compressed form, and decompressed only on demand. Fourth, certain common query operations (for example, equality conditions, duplicate elimination) can often be performed on the compressed data itself without having to decompress the data. Finally, compressing data that is transferred over the network effectively increases the available network bandwidth. This is important for parallel DBMSs where data must be moved across nodes. Data compression plays a key role not just in relational DBMSs, but also in other specialized engines, for example, in OLAP.

There are different compression techniques used in relational DBMSs. *Null suppression* leverages the fact that several commonly used data types in DBMSs are *fixed length* (for example, *int*, *bigint*, *datetime*, *money*), and significant compression is possible if they are treated as variable length for storage purposes. Only the non-null part of the value is stored along with the actual length of the value. *Dictionary compression* identifies repetitive values in the data and constructs a dictionary that maps such values to more compact representations. For example, a column that stores the shipping mode for an order may contain string values such as ‘AIR’, ‘SHIP’, ‘TRUCK’. Each value can be represented using two bits by mapping them to values 0,1,2 respectively. Finally, unlike compression schemes in row-oriented stores where each instance of a value requires an entry (potentially with fewer bits), in column-oriented stores other compression techniques such as *run-length encoding* (RLE) can become more effective. In RLE compression, a sequence of k instances of value v is encoded by the pair (v,k) . RLE is particularly attractive when long runs of the same value occur; this can happen for columns with relatively few distinct values, or when the column values are sorted.

There are several interesting technical challenges in data compression. First, new compression techniques suitable for large data warehouses and incurring an acceptable trade-off with decompression and update costs are important. Second, even for known compression techniques important open problems remain—for example,

for RLE—the choice of sort order of the table can significantly affect the amount of compression possible. Determining the best sort order to use is a non-trivial optimization problem. Finally, the decision of whether to compress access structures is workload dependent. Thus, there is a need for automated physical design tools to also recommend which access structures should be compressed and how based on workload information.

Query Processing

A popular conceptual model used for BI tasks is the *multidimensional* view of data, as shown in Figure 2. In a multidimensional data model, there is a set of *numeric measures* that are the objects of analysis. Examples of such measures are sales, budget, revenue, and inventory. Each of the numeric measures is associated with a set of *dimensions*, which provide the context for the measure. For example, the dimensions associated with a sale amount can be the Product, City, and the Date when the sale was made. Thus, a measure can be viewed as a value in the multidimensional space of dimensions. Each dimension is described by a set of attributes, for example, the Product dimension may consist of the following attributes: the category, industry, model number, year of its introduction. The attributes of a dimension may be related via a *hierarchy* of relationships. For example, a product is related to its category and the industry attributes through a hierarchical relationship (Figure 2). Another distinctive feature of the conceptual model is its stress on *aggregation* of measures by one or more dimensions; for example, computing and ranking the *total* sales by each county for each year.

OLAP Servers. Online Analytic processing (OLAP) supports operations such as filtering, aggregation, pivoting, rollup and drill-down on the multidimensional view of the data. OLAP servers are implemented using either a multidimensional storage engine (MOLAP); a relational DBMS engine (ROLAP) as the backend; or a hybrid combination called HOLAP.

MOLAP servers. MOLAP servers directly support the multidimensional view of data through a storage engine that uses the multidimensional array

abstraction. They typically precompute large *data cubes* to speed up query processing. Such an approach has the advantage of excellent indexing properties and fast query response times, but provides relatively poor storage utilization, especially when the data set is sparse. To better adapt to sparse data sets, MOLAP servers identify dense and sparse regions of the data, and store/index these regions differently. For example dense sub-arrays of the cube are identified and stored in array format, whereas the sparse regions are compressed and stored separately.

ROLAP servers. In ROLAP, the multidimensional model and its operations have to be mapped into relations and SQL queries. They rely on the data storage techniques described earlier to speed up relational query processing. ROLAP servers may also need to implement functionality not supported in SQL, for example, extended aggregate functions such as *median*, *mode*, and time window based *moving average*. The database designs used in ROLAP are optimized for efficiency in querying and in loading data. Most ROLAP systems use a *star schema* to represent the multidimensional data model. The database consists of a single fact table and a single table for each dimension. Each row in the fact table consists of a pointer (a.k.a. foreign key) to each of the dimensions that provide its multidimensional coordinates, and stores the numeric measures for those coordinates. Each dimension table consists of columns that correspond to attributes of the dimension. Star schemas do not explicitly provide support for attribute hierarchies. *Snowflake schemas* (shown in Figure 3) provide a refinement of star schemas where the dimensional hierarchy is explicitly represented by normalizing the dimension tables. This leads to advantages in maintaining the dimension tables.

HOLAP servers. The HOLAP architecture combines ROLAP and MOLAP by splitting storage of data in a MOLAP and a relational store. Splitting the data can be done in different ways. One method is to store the detailed data in a RDBMS as ROLAP servers do, and precomputing aggregated data in MOLAP. Another method is to store more recent data in MOLAP to pro-

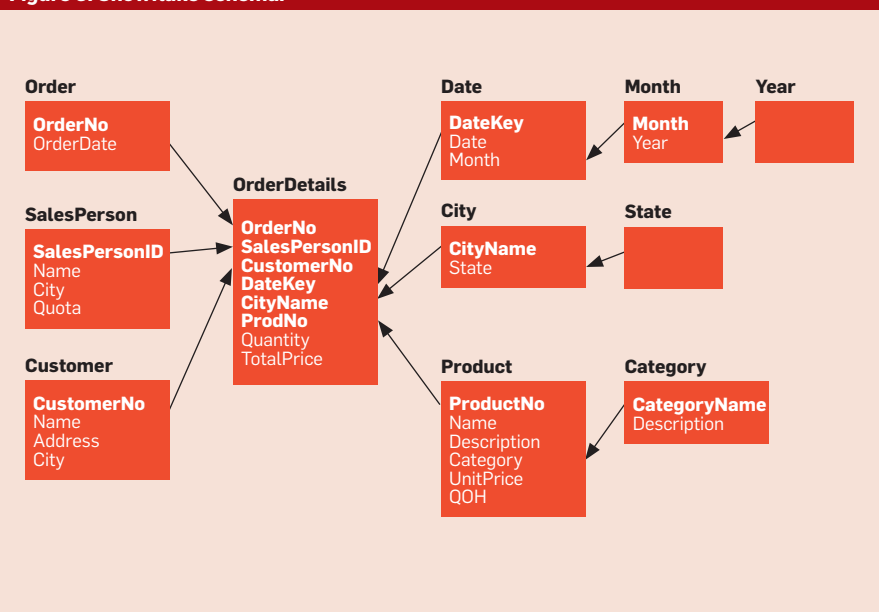
vide faster access, and older data in ROLAP. Since MOLAP performs better when the data is reasonably dense and ROLAP servers perform for sparse data, Like MOLAP servers, HOLAP servers also perform density analysis to identify sparse and dense sub-regions of the multidimensional space. All major data warehouse vendors today offer OLAP servers (for example, IBM Cognos,¹⁵ Microsoft SQL,¹⁷ and Oracle Hyperion²³).

In-memory BI engines. Technology trends are providing an opportunity for a new class of OLAP engines focused on exploiting large main memory to make response times for *ad-hoc queries* interactive. First, the ratio of time to access data on disk vs. data in memory is increasing. Second, with 64-bit operating systems becoming common, very large addressable memory sizes (for example, 1TB) are possible. Third, the cost of memory has dropped significantly, which makes servers with large amounts of main memory affordable. Unlike traditional OLAP servers, in-memory BI engines (for example, QlikView²⁴) rely on a different set of techniques for achieving good performance. First, since the detailed data is memory resident they avoid expensive I/Os required to access data cubes, indexes, or materialized views. Second, they use data structures that would not be suitable for disk-based access, but are very effective for in-memory access.

For example, consider a query that computes the total sales for each customer in a particular state. When the data is initially loaded into the system, the engine can associate *pointers* from each state to customers in that state, and similarly pointers from a customer to all the order detail records for that customer. This allows fast associative access required to answer the query quickly, and is reminiscent of approaches used by object-oriented databases as well as optimizations in traditional DBMSs such as *join indices*. Third, in-memory BI engines can significantly increase the effective data sizes over which they can operate in memory by using data organization techniques such as column-oriented storage and data compression. In-memory BI engines are best suited for read-mostly data without in-place data updates where new data arrives primarily in the form of incremental batch inserts due to data decompression cost.

Relational Servers. Relational database servers (RDBMSs) have traditionally served as the backend of large data warehouses. Such data warehouses need to be able to execute *complex* SQL queries as efficiently as possible against *very large databases*. The first key technology needed to achieve this is query optimization, which takes a complex query and compiles that query into an execution plan. To ensure that the execution plan can scale well to

Figure 3. Snowflake schema.



large databases, data partitioning and parallel query processing are leveraged extensively (see Graefe¹³ for an overview of query processing techniques). We therefore discuss two pieces of key technology—query optimization and parallel query processing.

Query optimization technology has been a key enabler for BI. The query optimizer is responsible for selecting an *execution plan* for answering a query. The execution plan is a composition of physical operators (such as Index Scan, Hash Join, Sort) that when evaluated generates the results of the query. The performance of a query crucially depends on the ability of the optimizer to choose a good plan from a very large space of alternatives. The difference in execution time between a good and bad plan for such complex queries can be several orders of magnitudes (for example, days instead of minutes). This topic has been of keen interest in database research and industry (an overview of the field appears in Chaudhuri⁴). Following the pioneering work done in the System R optimizer from IBM Research in the late 1970s, the next major architectural innovation came about a decade later: extensible optimizers. This allowed system designers to “plug-in” new *rules* and extend the capabilities of the optimizer. For example, a rule could represent equivalence in relational algebra (for example, pushing down an aggregation below join). Application of such rules can potentially transform the execution plan into one that executes much faster. Extensible optimizers allowed many important optimizations developed by industry and research over the years to be incorporated relatively easily without having to repeatedly modify the search strategy of the optimizer.

Despite the success of query optimization and the crucial role it plays in BI, many fundamental challenges still remain. The optimizer needs to address the inherently difficult problem of estimating the *cost* of a plan, that is, the total work (CPU, I/O, among others) done by the plan. However, constrained by the requirement to impose only a small overhead, the optimizer typically uses limited statistical information such as histograms describing a column’s data distribution. Such approximations

sometimes result in brittleness since large inaccuracies can lead to generation of very poor plans. There has been research in leveraging feedback from query execution to overcome errors made by the query optimizer by observing *actual* query execution behavior (for example, the actual result size of a query expression), and adjusting the execution plan if needed. However, collecting and exploiting feedback at low overhead is also challenging, and much more work is needed to realize the benefits of this approach.

Parallel processing and appliances. Parallelism plays a significant role in processing queries over massive databases. Relational operators such as selection, projection, join, and aggregation present many opportunities for parallelism. The basic paradigm is *data parallelism*, that is, to apply relational operators in parallel on disjoint subsets of data (partitions), and then combine the results. The article by Dewitt and Gray¹⁰ provides an overview of work in this area. For several years now, all major vendors of database management systems have offered data partitioning and parallel query processing technology. There are two basic architectures for parallelism: *Shared disk*, where each processor has a private memory but shares disks with all other processors. *Shared nothing*, where each processor has private memory and disk and is typically a low-cost commodity machine. Interestingly, while these architectures date back about two decades, neither has yet emerged as a clear winner in the industry and successful implementations of both exist today.

In shared disk systems all nodes have access to the data via shared storage, so there is no need to a priori partition the data across nodes as in the shared nothing approach. During query processing, there is no need to move data across nodes. Moreover, load balancing is relatively simple since any node can service any request. However, there are a couple of issues that can affect scalability of shared disk systems. First, the nodes need to communicate in order to ensure data consistency. Typically this is implemented via a *distributed lock manager*, which can incur non-trivial overhead. Second, the network must support the combined I/O bandwidth

of all processors, and can become a bottleneck. Shared disk systems are relatively cost effective for small- to medium-sized data warehouses.

In shared nothing systems (for example, Teradata³¹) data needs to be distributed across nodes a priori. They have the potential to scale to much larger data sizes than shared disk systems. However, the decision of how to effectively distribute the data across nodes is crucial for performance and scalability. This is important both from the standpoint of leveraging parallelism, but also to reduce the amount of data that needs to be transferred over the network during query processing. Two key techniques for data distribution are partitioning and cloning. For example consider a large database with the schema shown in Figure 3. Each of the two large fact tables, *Orders* and *OrderDetails* can be hash partitioned across all nodes on the *OrderId* attribute respectively, that is, on the attribute on which the two tables are joined. All other dimension tables, which are relatively small, could be cloned (replicated) on each node. Now consider a query that joins *Customers*, *Orders* and *OrderDetails*. This query can be processed by issuing one query per node, each operating on a subset of the fact data and joining with the entire dimension table. As a final step, the results of each of these queries are sent over the network to a single node that combines them to produce the final answer to the query.

Data warehouse appliances. Recently a new generation of parallel DBMSs referred to as data warehouse appliances (for example, Netezza¹⁹) have appeared. An appliance is an integrated set of server and storage hardware, operating system and DBMS software specifically *pre-installed* and *pre-optimized* for data warehousing. These appliances have gained impetus from the following trends. First, since DW appliance vendors control the full hardware/software stack, they can offer the more attractive one service call model. Second, some appliances push part of the query processing into specialized hardware thereby speeding up queries. For example, Netezza uses FPGAs (field-programmable gate arrays) to evaluate selection and projection operators on a table in the storage layer itself. For

typical decision support queries this can significantly reduce the amount of data that needs to be processed in the DBMS layer.

Distributed Systems using MapReduce Paradigm. Large-scale data processing engines based on the *MapReduce* paradigm⁹ were originally developed to analyze Web documents, query logs, and click-through information for index generation and for improving Web search quality. Platforms based on a distributed file system and using the *MapReduce* runtime (or its variants such as Dryad¹⁶) have been successfully deployed on clusters with an order of magnitude more nodes than traditional parallel DBMSs. Also, unlike parallel DBMSs where the data must first be loaded into a table with a predefined schema before it can be queried, a *MapReduce* job can directly be executed on schema-less input files. Furthermore, these data platforms are able to automatically handle important issues such as data partitioning, node failures, managing the flow of data across nodes, and heterogeneity of nodes.

Data platforms based on the *MapReduce* paradigm and its variants have attracted strong interest in the context of the “Big Data” challenge in enterprise analytics, as described in the introduction. Another factor that makes such platforms attractive is the ability to support analytics on unstructured data such as text documents (including Web crawls), image and sensor data by enabling execution of custom *Map* and *Reduce* functions in a scalable manner. Recently, these engines have been extended to support features necessary for enterprise adoption (for example, Cloudera⁸). While serious enterprise adoption is still in early stages compared to mature parallel RDBMS systems, exploration using such platforms is growing rapidly, aided by the availability of the open source Hadoop¹⁴ ecosystem. Driven by the goal of improving programmer productivity while still exploiting the advantages noted here, there have been recent efforts to develop engines that can take a SQL-like query, and automatically compile it to a sequence of jobs on a *MapReduce* engine (for example, Thusoo et al.³²). The emergence of analytic engines based on *MapReduce*

is having an impact on parallel DBMS products and research. For example, some parallel DBMS vendors (for example, Aster Data²) allow invocation of *MapReduce* functions over data stored in the database as part of a SQL query. The *MapReduce* function appears in the query as a table that allows its results to be composed with other SQL operators in the query. Many other DBMS vendors provide utilities to move data between *MapReduce*-based engines and their relational data engines. A primary use of such a bridge is to ease the movement of structured data distilled from the data analysis on the *MapReduce* platform into the SQL system.

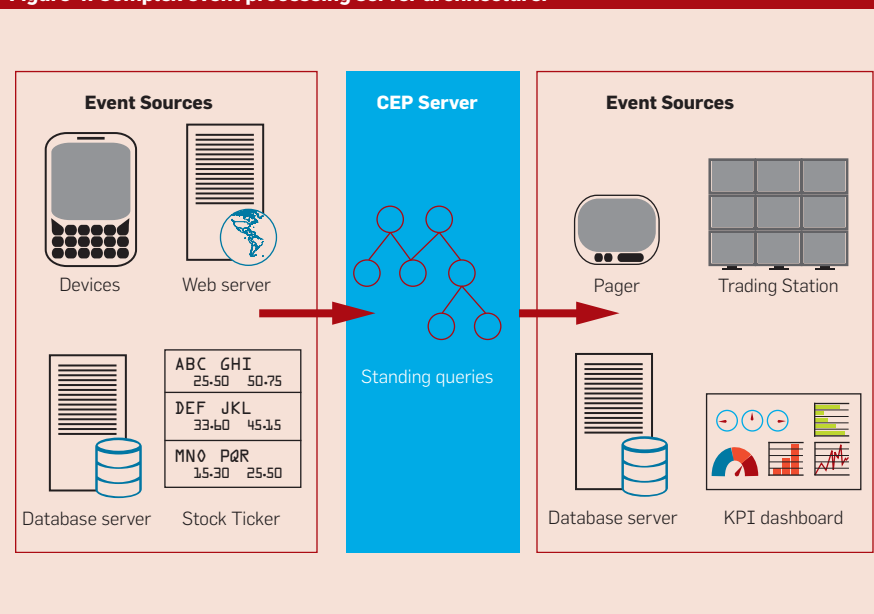
Near Real-Time BI. The competitive pressure of today’s businesses has led to the increased need for near real-time BI. The goal of near real-time BI (also called operational BI or just-in-time BI) is to reduce the latency between when operational data is acquired and when analysis over that data is possible. Consider an airline that tracks its most profitable customers. If a high-value customer has a lengthy delay for a flight, alerting the ground staff proactively can help the airline ensure that the customer is potentially rerouted. Such near real-time decisions can increase customer loyalty and revenue.

A class of systems that enables such real-time BI is Complex Event Processing (CEP) engines (for example,

Streambase²⁹). Businesses can specify the patterns or temporal trends that they wish to detect over streaming operational data (referred to as events), and take appropriate actions when those patterns occur. The genesis of CEP engines was in the financial domain where they were used for applications such as algorithmic stock trading, which requires detecting patterns over stock ticker data. However, they are now being used in other domains as well to make decisions in real time, for example, clickstream analysis or manufacturing process monitoring (for example, over RFID sensor data).

CEP is different from traditional BI since operational data does not need to be first loaded into a warehouse before it can be analyzed (see Figure 4). Applications define declarative queries that can contain operations over streaming data such as filtering, windowing, aggregations, unions, and joins. The arrival of events in the input stream(s) triggers processing of the query. These are referred to as “standing” or “continuous” queries since computation may be continuously performed as long as events continue to arrive in the input stream or the query is explicitly stopped. In general, there could be multiple queries defined on the same stream; thus one of the challenges for the CEP engine is to effectively share computation across queries when possible. These engines also need to handle situations where the streaming

Figure 4. Complex event processing server architecture.



data is delayed, missing, or out-of-order, which raise both semantic as well as efficiency challenges.

There are several open technical problems in CEP; we touch upon a few of them here. One important challenge is to handle continuous queries that reference data in the database (for example, the query references a table of customers stored in the database) without affecting near real-time requirements. The problem of optimizing query plans over streaming data has several open challenges. In principle, the benefit of an improved execution plan for the query is unlimited since the query executes “forever.” This opens up the possibility of more thorough optimization than is feasible in a traditional DBMS. Moreover, the ability to observe execution of operators in the execution plan over an extended period of time can be potentially valuable in identifying sub-optimal plans. Finally, the increasing importance of real-time analytics implies that many traditional data mining techniques may need to be revisited in the context of streaming data. For example, algorithms that require multiple passes over the data are no longer feasible for streaming data.

Enterprise Search

BI tasks often require searching over different types of data within the enterprise. For example, a salesperson

who is preparing for a meeting with a customer would like to know relevant customer information before the meeting. This information is today siloed into different sources: CRM databases, email, documents, and spreadsheets, both in enterprise servers as well as on the user’s desktop. Increasingly, a large amount of valuable data is present in the form of text, for example, product catalogs, customer emails, annotations by sales representatives in databases, survey responses, blogs and reviews. In such scenarios, the ability to retrieve and rank the required information using the keyword search paradigm is valuable for BI. Enterprise search focuses on supporting the familiar keyword search paradigm over text repositories and structured enterprise data. These engines typically exploit structured data to enable *faceted* search. For example, they might enable filtering and sorting over structured attributes of documents in the search results such as authors, last modification date, document type, companies (or other entities of interest) referenced in documents. Today, a number of vendors (for example, FAST Engine Search¹¹ and Google Search Appliance¹²) provide enterprise search capability.

A popular architecture for enterprise search engines is the *integrated* model, shown in Figure 5. The search engine crawls each data source and

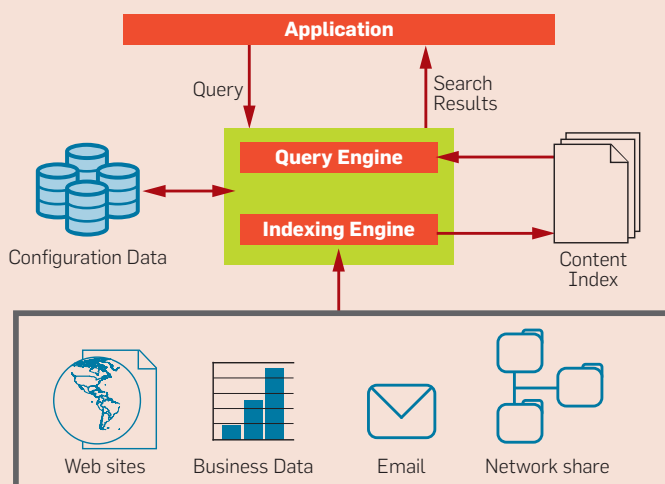
stores the data into a central content index using an internal representation that is suitable for fast querying. The configuration data controls what objects to index (for example, a *crawl query* that returns objects from a database) as well as what objects to return in response to a user query (for example, a *serve query* to run against the database when the query keywords match a crawled object). Several technical challenges need to be addressed by enterprise search engines. First, crawling relies on the availability of appropriate *adapters* for each source. Achieving a high degree of data freshness requires specialized adapters that can efficiently identify and extract data changes at the source. Second, ranking results across data sources is non-trivial since there may be no easy way to compare relevance across sources. Unlike ranking in Web search, links across documents in an enterprise are much sparser and thus not as reliable a signal. Similarly, query logs and click-through information are typically not available at sufficient scale to be useful for ranking. Finally, deploying enterprise search can involve manually tuning the relevance, for example, by adjusting the weight of each source.

Extract-Transform-Load Tools

The accuracy and timeliness of reporting, ad hoc queries, and predictive analysis depends on being able to efficiently get high-quality data into the data warehouse from operational databases and external data sources. Extract-Transform-Load (ETL) refers to a collection of tools that play a crucial role in helping discover and correct data quality issues and efficiently load large volumes of data into the warehouse.

Data quality. When data from one or more sources is loaded into the warehouse, there may be errors (for example, a data entry error may lead to a record with State = ‘California’ and Country = ‘Canada’), inconsistent representations for the same value (for example, ‘CA’, ‘California’), and missing information in the data. Therefore, tools that help detect data quality issues and restore data integrity in the warehouse can have a high payoff for BI. *Data profiling* tools enable identification of data quality issues by detect-

Figure 5. Enterprise search architecture (integrated model).



ing violations of properties that are expected to hold in the data. For example, consider a database of customer names and addresses. In a clean database, we might expect that (Name, Address) combinations are unique. Data profiling tools *verify* whether this uniqueness property holds, and can quantify the degree to which it is violated, for example, this might happen if Name or Address information is missing. Data profiling tools can also *discover* rules or properties that hold in a given database. For example, consider an external data source that needs to be imported into a data warehouse. It is important to know which columns (or sets of columns) are *keys* (unique) for the source. This can help in matching the incoming data against existing data in the warehouse. For efficiency, these tools often use techniques such as sampling when profiling large databases.

Accurately *extracting structure* from a string can play an important role in improving data quality in the warehouse. For example, consider a shopping Web site that stores MP3 player product data with attributes such as Manufacturer, Brand, Model, Color, Storage Capacity and receives a data feed for a product as text, for example, “Coby MP3 512MB MP-C756 – Blue.” Being able to robustly parse the structured information present in the text into the appropriate attributes in the data warehouse is important, for example, for answering queries on the Web site. Vendors have developed extensive sets of parsing rules for important verticals such as products and addresses. The survey article by Sarawagi²⁸ discusses techniques to the broader area of information extraction.

Another important technology that can help improve data quality is *de-duplication*: identifying groups of approximately duplicate entities (for example, customers). This can be viewed as a graph clustering problem where each node is an entity and an edge exists between two nodes if the degree of similarity between two entities is sufficiently high. The function that defines the degree of similarity between two entities is typically based on string similarity functions such as edit distance (for example, ‘Robert’ and ‘Robert’ have an edit distance of as well as

domain-specific rules (for example, ‘Bob’ and ‘Robert’ are synonymous). Thus, the ability to efficiently perform such approximate string matching across many pairs of entities (also known as *fuzzy matching*) is important for de-duplication. Most major vendors support fuzzy matching and de-duplication as part of their ETL suite of tools. An overview of tools for merging data from different sources can be found in Bernstein.³

Data load and refresh. Data load and refresh utilities are responsible for moving data from operational databases and external sources into the data warehouse quickly and with as little performance impact as possible at both ends. There are two major challenges. First, there is a need to efficiently *capture data* at the sources, that is, identify and collect data to be moved to the data warehouse. *Triggers* are general-purpose constructs supported by SQL that allow rows modified by an insert/update SQL statement to be identified. However, triggers are a relatively heavyweight mechanism and can impose non-trivial overheads on the operational database running OLTP queries. A more efficient way of capturing changed data is to sniff the transaction log of the database. The *transaction log* is used by the database system to record all changes so that the system can recover in case of a crash. Some utilities allow pushing *filters* when processing transaction log records, so that only relevant changed data is captured; for example, only changed data pertaining to a particular department within the organization.

The second aspect relates to techniques for efficiently *moving* captured data into the warehouse. Over the years, database engines have developed specialized, performance optimized APIs for bulk-loading data rather than using standard SQL. Partitioning the data at the warehouse helps minimize disruption of queries at the data warehouse server. The data is loaded into a partition, which is then switched in using a metadata operation only. This way, queries referencing that table are blocked only for a very short duration required for the metadata operation rather than during the entire load time. Finally, load utilities

also typically checkpoint the operation so that in case of a failure the entire work does not need to be redone. Using the techniques discussed above for capturing changed data and efficient loading, these days utilities are able to approach refresh rates in a few seconds (for example, Oracle GoldenGate²²). Thus, it is potentially possible to even serve some near real-time BI scenarios, as discussed earlier.

Other BI Technology

Here, we discuss two areas we think are becoming increasingly important and where research plays a key role.

Data Mining and Text Analytics.

Data mining enables in-depth analysis of data including the ability to build predictive models. The set of algorithms offered by data mining go well beyond what is offered as aggregate functions in relational DBMSs and in OLAP servers. Such analysis includes decision trees, market basket analysis, linear and logistic regression, neural networks and more (see survey⁶). Traditionally, data mining technology has been packaged separately by statistical software companies, for example, SAS,²⁶ and SPSS.²⁷ The approach is to select a subset of data from the data warehouse, perform sophisticated data analysis on the selected subset of data to identify key statistical characteristics, and to then build predictive models. Finally, these predictive models are deployed in the operational database. For example, once a robust model to offer a room upgrade to a customer has been identified, the model (such as a decision tree) must be integrated back in the operational database to be actionable. This approach leads to several challenges: data movement from warehouse to the data mining engine, and potential performance and scalability issues at the mining engine (or implied limitations on the amount of data used to build a model). To be practical, such models need to be efficient to apply when new data arrives. Increasingly, the trend is toward “in-database analytics,” that is, integrating the data mining functionality in the backend data-warehouse architecture so that these limitations may be overcome (for example, Netz et al.²⁰ and Oracle Data Mining²¹).

Text analytics. Consider a company making portable music players that conducts a survey of its products. While many survey questions are structured (for example, demographic information), other open-ended survey questions (for example, “Enter other comments here”) are often free text. Based on such survey responses, the company would like to answer questions such as: Which products are referenced in the survey responses? What topics about the product are people mentioning? In these scenarios, the challenge is to reduce the human cost of having to read through large amounts of text data such as surveys, Web documents, blogs, and social media sites in order to extract structured information necessary to answer these queries. This is the key value of text analytic engines. Today’s text analysis engines (for example, FAST¹¹ and SAS²⁶) primarily extract structured data that can be broadly categorized as: *Named entities* are references to known objects such as locations, people, products, and organizations. *Concepts/topics* are terms in the documents that are frequently referenced in a collection of documents. For example, in the above scenario of portable music players, terms such as “battery life,” “appearance,” and “accessories” may be important concepts/topics that appear in the survey. Such information can potentially be used as a basis for categorizing the results of the survey. *Sentiment analysis* produces labels such as “positive,” “neutral,” or “negative” with each text document (or part of a document such as a sentence). This analysis can help answer questions such as which product received the most negative feedback.

Cloud Data Services. Managing enterprise BI today requires handling tasks such as hardware provisioning, availability, and security patching. Cloud virtualization technology (for example, Amazon EC2¹) allows a server to be hosted in the cloud in a virtual machine, and enables server consolidation through better utilization of hardware resources. Hosted servers also offer the promise of reduced cost by offloading manageability tasks, and leveraging the pay-as-you-go pricing model to only pay for services that are actually used. The success of hardware

virtualization in the cloud has prompted database vendors to virtualize data services so as to further improve resource utilization and reduce cost. These data services initially started as simple key-value stores but have now begun to support the functionality of a single node relational database as a hosted service (for example, Microsoft SQL Azure¹⁸). While the primary initial users of such cloud database services are relatively simple departmental applications (OLTP), the paradigm is being extended to BI as well (for example, Pentaho²⁵).

The need for the full range of BI services over the data collected by these applications raises new challenges for cloud database services. First, the performance and scale requirements of large reporting or ad hoc queries will require database service providers to support a massively parallel processing system (parallel DBMS and/or MapReduce-based engine) in the cloud. Second, these services are multi-tenant, and complex SQL queries can be resource intensive. Thus, the ability to provide performance Service Level Agreements (SLAs) to tenants and judiciously allocate system resources across tenant queries becomes important. Third, many of the technical challenges of traditional “in-house” BI such as security and fine grained access control become amplified in the context of cloud data services. For example, techniques for processing queries on encrypted data become more important in public clouds. For these reasons, an intermediate step in adoption of BI technologies may be in private clouds, which hold promise similar to public clouds but with more control over aspects such as security.

Conclusion

The landscape of BI in research and industry is vibrant today. Data acquisition is becoming easier and large data warehouses with 10s to 100s of terabytes or more of relational data are becoming common. Text data is also being exploited as a valuable source for BI. Changes in the hardware technology such as decreasing cost of main memory are impacting how the backend of large data-warehouses are architected. Moreover, as cloud data services take root, more changes in

the BI backend architecture are expected. Finally, there is increasing demand to deliver interactive BI experiences on mobile devices for today’s knowledge workers. There are ample opportunities to enable novel, rich, and interactive BI applications on the next generation of mobile devices. Thus, business intelligence software has many exciting technical challenges and opportunities still ahead that will continue to reshape its landscape. ■

References

1. Amazon EC2; <http://aws.amazon.com>
2. Aster Data; <http://www.asterdata.com>.
3. Bernstein, P. and Haas, L. Information integration in the enterprise. *Commun. ACM* 51, 9 (Sept. 2008).
4. Chaudhuri, S. An overview of query optimization in relational systems. *ACM PODS* 1998.
5. Chaudhuri, S. and Dayal, U. An overview of data warehousing and OLAP technology. *SIGMOD Record* 26, 1 (1997).
6. Chaudhuri, S., Dayal, U. and Ganti, V. Database technology for decision support systems. *IEEE Computer* 34, 12 (2001).
7. Chaudhuri, S. and Narasayya, V. Self-tuning database systems: A decade of progress. In *Proceedings of VLDB 2007*.
8. Cloudera Enterprises; <http://www.cloudera.com>
9. Dean, J. and Ghemawat, S. MapReduce: Simplified data processing on large clusters. In *Proceedings of OSDI 2004*.
10. DeWitt D.J. and Gray J. Parallel database systems: The future of high -performance database systems. *Commun. ACM* 35, 6 (June 1992).
11. FAST Enterprise Search; <http://www.fastsearch.com>
12. Google Search Appliance; <http://www.google.com/enterprise/gsa>
13. Graefe, G. Query evaluation techniques for large databases. *ACM Computing Surveys* 25, 2 (June 1993).
14. Hadoop; <http://hadoop.apache.org>
15. IBM Cognos; <http://www.ibm.com>
16. Isard et al. Dryad: Distributed data-parallel programs from sequential building blocks. In *Proceedings of Eurosys 2001*.
17. Microsoft SQL Server Analysis Services; <http://www.microsoft.com>
18. Microsoft SQL Azure; <http://www.microsoft.com>
19. Netezza; <http://www.netezza.com>
20. Netz, A., Chaudhuri, S., Fayyad, U. and Bernhardt, J. Integrating data mining with SQL databases. *OLE DB for Data Mining*, 2001.
21. Oracle Data Mining; <http://www.oracle.com>
22. Oracle GoldenGate; <http://www.oracle.com>
23. Oracle Hyperion; <http://www.oracle.com>
24. QlikView; <http://www.qlikview.com>
25. Pentaho; <http://www.pentaho.com>
26. SAS: Business Analytics and Business Intelligence Software; <http://www.sas.com>
27. SPSS: Data Mining, Statistical Analysis, Predictive Analytics, Decision Support Systems; <http://www.spss.com>
28. Sarawagi, S. Information extraction. *Foundations and Trends in Databases* 1, 3 (2008), 261-377.
29. Streambase; <http://www.streambase.com>
30. Sybase IQ; <http://www.sybase.com>
31. Teradata; <http://www.teradata.com>
32. Thusoo, A. et al. Hive—A warehousing solution over a MapReduce framework. *VLDB* 2009

Surajit Chaudhuri (surajitc@microsoft.com) is a principal researcher at Microsoft Research, Redmond, WA.

Umeshwar Dayal (umeshwar.dayal@hp.com) is an HP Fellow in the Intelligent Enterprise Technology Labs at Hewlett-Packard Lab, Palo Alto, CA.

Vivek Narasayya (viveknar@microsoft.com) is a principal researcher at Microsoft Research, Redmond, WA.

research highlights

P. 100

Technical Perspective Sketches Get Sketchier

By Peter J. Haas

P. 101

Theory and Applications of b -Bit Minwise Hashing

By Ping Li and Arnd Christian König

P. 110

Technical Perspective Skin Introducing the Future

By Scott Klemmer

P. 111

Skinput: Appropriating the Skin as an Interactive Canvas

By Chris Harrison, Desney Tan, and Dan Morris

Technical Perspective

Sketches Get Sketchier

By Peter J. Haas

ARE DATA SYNOPSES—such as the hash-based sketches discussed by Li and König in the following paper—still needed for querying massive datasets? After all, disk and memory are cheap, and both modern multicore processors and data-processing platforms such as Hadoop are rendering Web scale datasets ever more amenable to parallel and distributed processing techniques. The answer to this question is a firm “yes!” Many important data-processing tasks, such as detecting duplicate Web pages, are not embarrassingly parallel and involve a huge number of operations. Even massively parallelizable tasks face formidable performance challenges: it has long been observed that data volumes are growing at faster rates than computing power, and this trend continues. Moreover, methods that rely solely on parallelism can be expensive. Indeed, under evolving “platform as a service” models for cloud computing, users will pay costs that directly reflect the computing resources they use. In this setting, use of synopsis techniques can lead to significant cost savings, as well as to energy savings and greener computing. The need to process streaming data exacerbates the pressures on data management systems and makes synopsis techniques even more appealing. So synopses such as samples, histograms, wavelets, and sketches will continue to play a key role in information management, data mining, and knowledge discovery. Reducing synopses’ time and space requirements thus remains an important challenge.

The paper highlights the fascinating interplay between two synopsis techniques: hashing and sampling. Random samples have long served as useful synopses because of their flexibility: a given sample can be used to estimate many different characteristics of the original dataset. This

flexibility often comes at the price of limited accuracy, however. In particular, sampling does poorly at finding “needles in haystacks” and so fails at, for example, estimating the number of distinct values in a “skewed” dataset containing a few values that each occur millions of times along with a few hundred values that each occur once. Hash-based sketches, on the other hand, are typically applicable to only one particular task, but perform that task extremely well; a famous example is the Flajolet-Martin sketch for the number of distinct values, based on a single pass through the dataset. In general, hash-based sketches have proven to be extremely well suited for streaming applications, and for this reason have received much attention in recent years.

Although sampling and hashing might seem complementary, they are closely related. Indeed, suppose we apply a given hash function to each of the elements of a set and then select the element having the smallest hashed value. We can reasonably view this process as being equivalent to selecting an element of the set at random. Repeating this process with several different hash functions yields a random sample of set elements. (Strictly speaking, we need to view the hash functions as being randomly and uniformly selected from an appropriate family of hash functions, in order to put hashing into a probabilistic setting comparable to that of sampling.) The crucial property of this approach is that the same hash function can be used to sample from multiple sets, and this coordinated sampling leads to powerful methods for rapidly estimating the similarity between pairs of sets. Indeed, it has long been recognized in the statistics community that inducing correlations between sam-

random numbers—can greatly reduce variability when estimating similarities between different populations or systems; hash-based sampling is another manifestation of this idea.

The minwise-hashing (minHash) algorithm originally developed by Andrei Broder and colleagues exploits hash-based sampling to estimate the “Jaccard” similarity between two sets in a simple and elegant way, by computing the fraction of hash functions that result in the same element being selected from both sets, or equivalently, the fraction of hash functions for which the minimum hashed values over each of the two sets coincide. Thus, the sketch for a given dataset is the collection of minimum hashed values for the different hash functions. This idea has been applied to problems ranging from the identification of duplicate Web pages or documents to applications in clustering, caching, online advertising, file-system management, and association-rule mining. The striking result in this paper is that, if the goal is to identify sets of high similarity, it suffices to retain only a few low-order bits of each hashed value in the original minHash sketch. The resulting modification to the original minHash algorithm is minor, but can reduce the space and estimation-time requirements by well over an order of magnitude. The challenge is to find the right estimator for the Jaccard similarity and then to theoretically verify both the correctness and superior properties of the resulting algorithm. An elegant use of probabilistic approximations accomplishes this goal. This work is an ideal blend of theory and application. 

Peter J. Haas (peterh@almaden.ibm.com) is a research staff member at IBM Almaden Research Center, San Jose, CA.

© 2011 ACM 0001-0782/11/08 \$10.00

Theory and Applications of b -Bit Minwise Hashing

By Ping Li and Arnd Christian König

Abstract

Efficient (approximate) computation of set similarity in very large datasets is a common task with many applications in information retrieval and data management. One common approach for this task is *minwise hashing*. This paper describes *b-bit minwise hashing*, which can provide an order of magnitude improvements in storage requirements and computational overhead over the original scheme in practice.

We give both theoretical characterizations of the performance of the new algorithm as well as a practical evaluation on large real-life datasets and show that these match very closely. Moreover, we provide a detailed comparison with other important alternative techniques proposed for estimating set similarities. Our technique yields a very simple algorithm and can be realized with only minor modifications to the original minwise hashing scheme.

1. INTRODUCTION

With the advent of the Internet, many applications are faced with very large and inherently high-dimensional datasets. A common task on these is *similarity search*, that is, given a high-dimensional data point, the retrieval of data points that are close under a given distance function. In many scenarios, the storage and computational requirements for computing exact distances between all data points are prohibitive, making data representations that allow compact storage and efficient approximate distance computation necessary.

In this paper, we describe *b-bit minwise hashing*, which leverages properties common to many application scenarios to obtain order-of-magnitude improvements in the storage space and computational overhead required for a given level of accuracy over existing techniques. Moreover, while the theoretical analysis of these gains is technically challenging, the resulting algorithm is simple and easy to implement.

To describe our approach, we first consider the concrete task of Web page duplicate detection, which is of critical importance in the context of Web search and was one of the motivations for the development of the original *minwise hashing* algorithm by Broder et al.^{2,4} Here, the task is to identify pairs of pages that are textually very similar. For this purpose, Web pages are modeled as “a set of shingles,” where a shingle corresponds to a string of w contiguous words occurring on the page. Now, given two such sets $S_1, S_2 \subseteq \Omega$, $|\Omega| = D$, the normalized similarity known as *resemblance* or *Jaccard similarity*, denoted by R , is

$$R = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = \frac{a}{f_1 + f_2 - a}, \quad \text{where } f_1 = |S_1|, f_2 = |S_2|.$$

Duplicate detection now becomes the task of detecting pairs of pages for which R exceeds a threshold value. Here, w is a tuning parameter and was set to be $w=5$ in several studies.^{2,4,7} Clearly, the total number of possible shingles is huge. Considering 10^5 unique English words, the total number of possible 5-shingles should be $D=(10^5)^5=O(10^{25})$. A prior study⁷ used $D=2^{64}$ and even earlier studies^{2,4} used $D=2^{40}$. Due to the size of D and the number of pages crawled as part of Web search, computing the exact similarities for all pairs of pages may require prohibitive storage and computational overhead, leading to approximate techniques based on more compact data structures.

1.1. Minwise hashing

To address this issue, Broder and his colleagues developed *minwise hashing* in their seminal work.^{2,4} Here, we give a brief introduction to this algorithm. Suppose a random permutation π is performed on Ω , that is,

$$\pi : \Omega \rightarrow \Omega, \quad \text{where } \Omega = \{0, 1, \dots, D-1\}.$$

An elementary probability argument shows that

$$\Pr(\min(\pi(S_1)) = \min(\pi(S_2))) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = R. \quad (1)$$

After k minwise independent permutations, $\pi_1, \pi_2, \dots, \pi_k$, one can estimate R without bias, as a binomial probability:

$$\hat{R}_M = \frac{1}{k} \sum_{j=1}^k 1\{\min(\pi_j(S_1)) = \min(\pi_j(S_2))\}, \quad (2)$$

$$\text{Var}(\hat{R}_M) = \frac{1}{k} R(1-R). \quad (3)$$

We will frequently use the terms “sample” and “sample size” (i.e., k). For minwise hashing, a sample is a hashed value, $\min(\pi_j(S_i))$, which may require, for example, 64 bits.⁷

Since the original minwise hashing work,^{2,4} there have been considerable theoretical and methodological developments.^{3,5,12,14,16,17,22}

Applications: As a general technique for estimating set similarity, minwise hashing has been applied to a wide range of applications, for example, content matching for online advertising,²³ detection of redundancy in enterprise

The previous version of this paper, entitled “b-Bit Minwise Hashing for Estimating Three-way Similarities,” was published in *Proceedings of the Neural Information Processing Systems: NIPS 2010* (Vancouver, British Columbia, Canada).

file systems,⁸ syntactic similarity algorithms for enterprise information management,²¹ Web spam,²⁵ etc.

Many of the applications of minwise hashing are targeted at detecting duplicates or pairs of somewhat high similarity. By proposing an estimator that is particularly accurate for these scenarios, we can reduce the required storage and computational overhead dramatically. Here, the computational savings are a function of how the minwise hashes are used. For any technique that does compute the pairwise similarity for (a large subset of) all pairs, the computation is typically bound by the speed at which the samples can be brought into memory (as the computation itself is simple); hence, the space reduction our technique offers directly translates into order-of-magnitude speedup as well.

However, even with the data-size reduction, computing all pairwise similarities is prohibitively expensive in many scenarios. This has led to a number of approaches that avoid this computation by grouping (subsets of) the samples into buckets and only computing the pairwise similarities for items within the same (set of) buckets. This approach avoids the quadratic number of comparisons, at the cost of some loss in accuracy. Examples of such approaches are the *supershingles*⁴ or techniques based on *locally sensitive hashing* (LSH)^{1, 5, 13} (also see Chapter 3 of Rajaraman and Ullman²⁴ for an excellent detailed explanation of LSH and see Cohen et al.⁶ for nice applications of LSH ideas in mining associations).

1.2. *b*-Bit minwise hashing

In this paper, we establish a unified theoretical framework for *b*-bit minwise hashing. In our scheme, a *sample* consists of *b* bits only, as opposed to, for example, *b* = 64 bits⁷ in the original minwise hashing. Intuitively, using fewer bits per sample will increase the estimation variance, compared to (3), at the same *sample size* *k*. Thus, we will have to increase *k* to maintain the same accuracy. Interestingly, our theoretical results will demonstrate that, when resemblance is not too small (which is the case in many applications, e.g., consider $R \geq 0.5$, the threshold used in Broder et al.^{2, 4}), we do not have to increase *k* much. This means our proposed *b*-bit minwise hashing can be used to improve estimation accuracy and significantly reduce storage requirements at the same time.

For example, when *b* = 1 and $R = 0.5$, the estimation variance will increase at most by a factor of 3. In order not to lose accuracy, we have to increase the sample size by a factor of 3. If we originally stored each hashed value using 64 bits, the improvement by using *b* = 1 will be $64/3 = 21.3$.

Algorithm 1 illustrates the procedure of *b*-bit minwise hashing, based on the theoretical results in Section 2.

1.3. Related work

Locality sensitive hashing (LSH)^{1, 5, 13} is a set of techniques for performing approximate search in high dimensions. In the context of estimating set intersections, there exist LSH families for estimating the *resemblance*, the *arccosine*, and the *hamming distance*. Our *b*-bit minwise hashing proposes a new construction of an LSH family (Section 7.4).

Algorithm 1 The *b*-bit minwise hashing algorithm, applied to estimating pairwise resemblances in a collection of *N* sets.

Input: Sets $S_n \subseteq \Omega = \{0, 1, \dots, D-1\}$, $n = 1$ to N .

Preprocessing

- (1): Generate *k* random permutations $\pi_j: \Omega \rightarrow \Omega$, $j = 1$ to *k*.
- (2): For each set S_n and each permutation π_j , store the lowest *b* bits of $\min(\pi_j(S_n))$, denoted by e_{n,i,π_j} , $i = 1$ to *b*.

Estimation: (Use two sets S_1 and S_2 as an example)

- (1): Compute $\hat{P}_b = \frac{1}{k} \sum_{j=1}^k \{ \prod_{i=1}^b 1\{e_{1,i,\pi_j} = e_{2,i,\pi_j}\} \}$.
- (2): Estimate the resemblance by $\hat{R}_b = \frac{\hat{P}_b - C_{1,b}}{1 - C_{2,b}}$, where $C_{1,b}$ and $C_{2,b}$ are from Theorem 1 in Section 2.

In Charikar⁵ and Gionis et al.,¹⁰ the authors describe hashing schemes that map objects to $\{0, 1\}$. The algorithms for the construction, however, are problem specific. Three discovered 1-bit schemes are (i) the *simhash*⁵ based on *sign random projection*,^{11, 18} (ii) the hamming distance algorithm based on simple random sampling,¹³ and (iii) the hamming distance algorithm based on a variant of random projection.¹⁵

Section 4 will compare our method with two *hamming distance* algorithms.^{13, 15} We also wrote a report (http://www.stat.cornell.edu/~li/b-bit-hashing/RP_minwise.pdf), which demonstrated that, unless the similarity is very low, *b*-bit minwise hashing outperforms sign random projections.

A related approach is *conditional random sampling* (CRS)^{16, 17} which uses only a single permutation and instead of a single minimum retains as set of the smallest hashed values. CRS provides more accurate (in some scenarios substantially so) estimators for binary data and naturally extends to real-value data and dynamic streaming data; moreover, the same set of hashed values can be used to estimate a variety of summary statistics including histograms, l_p distances (for any *p*), number of distinct values, χ^2 distances, entropies, etc. However, we have not developed a *b*-bit scheme for CRS, which appears to be a challenging task.

2. THE FUNDAMENTAL RESULTS

Consider two sets $S_1, S_2 \subseteq \Omega = \{0, 1, 2, \dots, D-1\}$, $f_1 = |S_1|$, $f_2 = |S_2|$, $a = |S_1 \cap S_2|$. Apply a random permutation π on S_1 and S_2 , where $\pi: \Omega \rightarrow \Omega$. Define the minimum values under π to be z_1 and z_2 :

$$z_1 = \min(\pi(S_1)), \quad z_2 = \min(\pi(S_2)). \quad (4)$$

Define $e_{1,i}$ = *i*th lowest bit of z_1 , and $e_{2,i}$ = *i*th lowest bit of z_2 . Theorem 1 derives the main probability formula. Its proof assumes that *D* is large, which is virtually always satisfied in practice. This result is a good example of approaching a difficult problem by reasonable approximations.

THEOREM 1. Assume *D* is large.

$$P_b = \Pr\left(\prod_{i=1}^b 1\{e_{1,i} = e_{2,i}\}\right) = C_{1,b} + (1 - C_{2,b})R \quad (5)$$

$$r_1 = \frac{f_1}{D}, \quad r_2 = \frac{f_2}{D},$$

$$C_{1,b} = A_{1,b} \frac{r_2}{r_1 + r_2} + A_{2,b} \frac{r_1}{r_1 + r_2}, \quad (6)$$

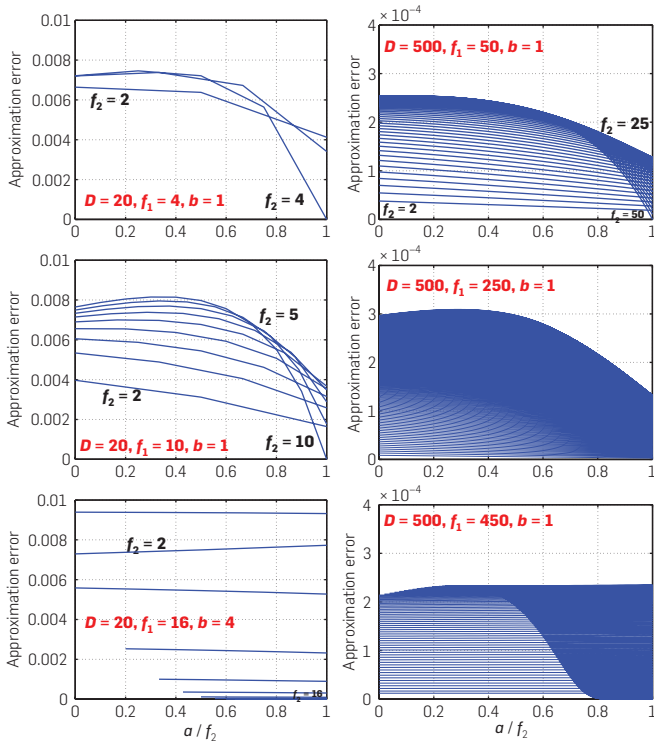
$$C_{2,b} = A_{1,b} \frac{r_1}{r_1 + r_2} + A_{2,b} \frac{r_2}{r_1 + r_2}, \quad (7)$$

$$A_{1,b} = \frac{r_1 [1-r_1]^{2^b-1}}{1-[1-r_1]^{2^b}}, \quad A_{2,b} = \frac{r_2 [1-r_2]^{2^b-1}}{1-[1-r_2]^{2^b}}. \quad (8)$$

The intuition for the difference between (5) and the equivalent equation for minwise hashing (1) is that even when $R=0$, the collision probability P_b (i.e., the probability that two minima agree on their last b bits) is not zero, but rather $C_{1,b}$. Having to account for this type of “false positives” makes the derivation more difficult, resulting in the additional terms in (5). Of course, as expected, if $R=1$, then $P_b=1$ (because in this case $r_1=r_2$ and $C_{1,b}=C_{2,b}$).

Note that the only assumption needed in the proof of Theorem 1 is that D is large, which is virtually always satisfied in practice. Interestingly, (5) is remarkably accurate even for very small D . Figure 1 shows that when $D=20$ ($D=500$), the absolute error caused by using (5) is <0.01 (<0.0004).

Figure 1. The absolute errors (approximate-exact) by using (5) are very small even for $D=20$ (left panels) or $D=500$ (right panels). The exact probability can be numerically computed for small D (from a probability matrix of size $D \times D$). For each D , we selected three f_1 values. We always let $f_2 = 2, 3, \dots, f_1$ and $a = 0, 1, 2, \dots, f_2$.



2.1. The unbiased estimator

Theorem 1 suggests an unbiased estimator $\hat{R}_b$ for R :

$$\hat{R}_b = \frac{\hat{P}_b - C_{1,b}}{1 - C_{2,b}}, \quad (9)$$

$$\hat{P}_b = \frac{1}{k} \sum_{j=1}^k \left\{ \prod_{i=1}^b 1\{e_{1,i,\pi_j} = e_{2,i,\pi_j}\} \right\}, \quad (10)$$

where e_{1,i,π_j} (e_{2,i,π_j}) denotes the i th lowest bit of z_1 (z_2), under the permutation π_j . The variance is

$$\begin{aligned} \text{Var}(\hat{R}_b) &= \frac{\text{Var}(\hat{P}_b)}{[1 - C_{2,b}]^2} = \frac{1}{k} \frac{P_b(1-P_b)}{[1 - C_{2,b}]^2} \\ &= \frac{1}{k} \frac{[C_{1,b} + (1 - C_{2,b})R][1 - C_{1,b} - (1 - C_{2,b})R]}{[1 - C_{2,b}]^2} \end{aligned} \quad (11)$$

For large b , $\text{Var}(\hat{R}_b)$ converges to the variance of $\hat{R}_M$, the estimator for the original minwise hashing:

$$\lim_{b \rightarrow \infty} \text{Var}(\hat{R}_b) = \frac{R(1-R)}{k} = \text{Var}(\hat{R}_M).$$

In fact, when $b=64$, $\text{Var}(\hat{R}_b)$ and $\text{Var}(\hat{R}_M)$ are numerically indistinguishable for practical purposes.

2.2. The variance-space trade-off

As we decrease b , the space needed for storing each “sample” will be smaller; the estimation variance (11) at the same sample size k , however, will increase. This variance-space trade-off can be precisely quantified by $B(b; R, r_1, r_2)$:

$$\begin{aligned} B(b; R, r_1, r_2) &= b \times \text{Var}(\hat{R}_b) \times k \\ &= \frac{b[C_{1,b} + (1 - C_{2,b})R][1 - C_{1,b} - (1 - C_{2,b})R]}{[1 - C_{2,b}]^2}. \end{aligned} \quad (12)$$

Lower $B(b)$ is better. The ratio, $\frac{B(b_1; R, r_1, r_2)}{B(b_2; R, r_1, r_2)}$, measures the improvement of using $b=b_2$ (e.g., $b_2=1$) over using $b=b_1$ (e.g., $b_1=64$). Some algebra yields the following Lemma.

LEMMA 1. If $r_1=r_2$ and $b_1 > b_2$, then

$$\frac{B(b_1; R, r_1, r_2)}{B(b_2; R, r_1, r_2)} = \frac{b_1}{b_2} \frac{A_{1,b_1}(1-R) + R}{A_{1,b_2}(1-R) + R} \frac{1 - A_{1,b_2}}{1 - A_{1,b_1}}, \quad (13)$$

is a monotonically increasing function of $R \in [0, 1]$.

If $R \rightarrow 1$ (which implies $r_1, r_2 \rightarrow 1$), then

$$\frac{B(b_1; R, r_1, r_2)}{B(b_2; R, r_1, r_2)} \rightarrow \frac{b_1}{b_2}. \quad (14)$$

If $r_1=r_2$, $b_2=1$, $b_1=64$ (hence we treat $A_{1,b}=0$), then

$$\frac{B(b_1; R, r_1, r_2)}{B(1; R, r_1, r_2)} = 64 \frac{R}{1 + R - r_1}. \quad (15)$$

Suppose the original minwise hashing used $b=64$, then the maximum improvement of b -bit minwise hashing would be 64-fold, attained when $r_1=r_2=1$ and $R=1$. In the least favorable situation, that is, $r_1, r_2 \rightarrow 0$, the improvement will still be $64 \frac{0.5}{1+0.5} = \frac{64}{3} = 21.3$ -fold when $R=0.5$.

Figure 2. $\frac{B(64)}{B(b)}$, the relative storage improvement of using $b = 1, 2, 3, 4$ bits, compared to using 64 bits. $B(b)$ is defined in (12).

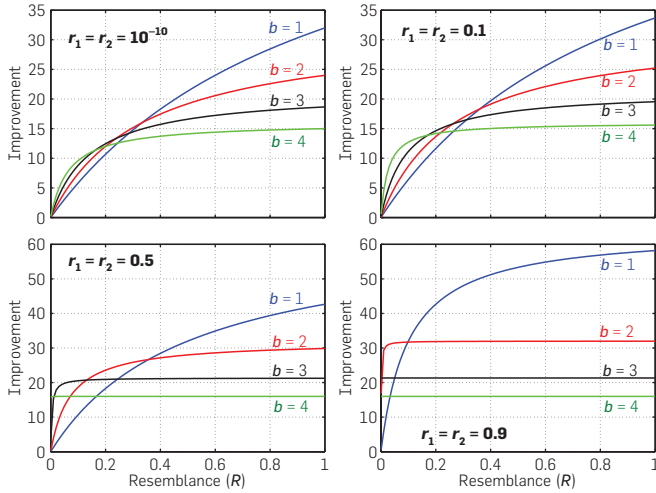


Figure 2 plots $\frac{B(64)}{B(b)}$ to directly visualize the relative improvements, which are consistent with what Lemma 1 predicts. The plots show that, when R is very large (which is the case in many practical applications), it is always good to use $b = 1$. However, when R is small, using larger b may be better. The cut-off point depends on r_1, r_2, R . For example, when $r_1 = r_2$ and both are small, it would be better to use $b = 2$ than $b = 1$ if $R < 0.4$, as shown in Figure 2.

3. EXPERIMENTS

In the following, we evaluate the accuracy of the theoretical derivation and the practical performance of our approach using two sets of experiments. *Experiment 1* is a sanity check, to verify: (i) our proposed estimator $\hat{R}_b$ in (9) is unbiased and (ii) its variance follows the prediction by our formula in (11). *Experiment 2* is a duplicate detection task using a Microsoft proprietary collection of 1,000,000 news articles.

3.1. Experiment 1

The data, extracted from Microsoft Web crawls, consists of six pairs of sets. Each set consists of the document IDs, which contain the word at least once. We now use b -bit minwise hashing to estimate the similarities of these sets (i.e., we estimate the strength of the word associations).

Table 1 summarizes the data and provides the theoretical improvements $\frac{B(64)}{B(1)}$. The words were selected to include highly frequent pairs (e.g., “OF-AND”), highly rare pairs (e.g., “GAMBIA-KIRIBATI”), highly unbalanced pairs (e.g., “A-TEST”), highly similar pairs (e.g., “KONG-HONG”), as well as pairs that are not quite similar (e.g., “LOW-PAY”).

We estimate the resemblance using the original minwise hashing estimator $\hat{R}_M$ and the b -bit estimator $\hat{R}_b$ ($b = 1, 2, 3$).

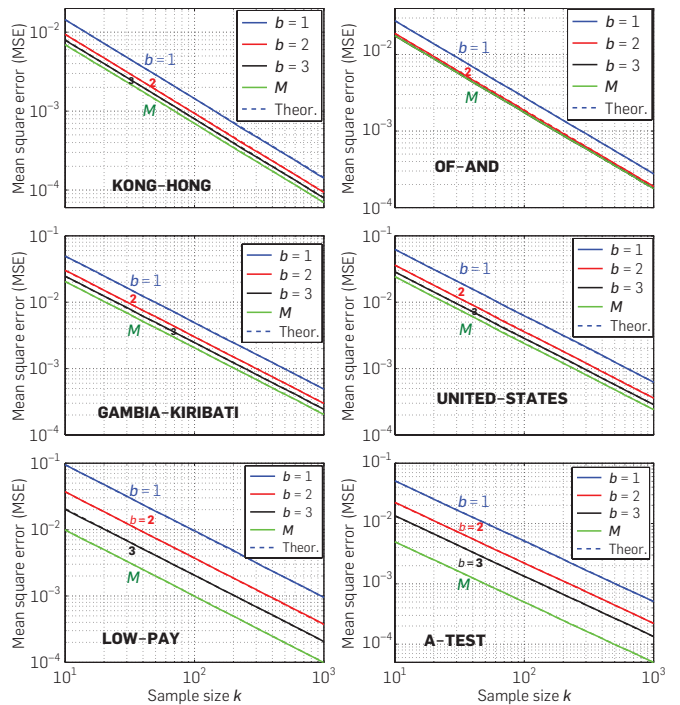
Figure 3 plots the empirical mean square errors (MSE = variance + bias²) in solid lines and the theoretical variances (11) in dashed lines for all word pairs. All dashed lines are invisible because they overlap with the corresponding solid curves. Thus, this experiment validates that the variance

Table 1. Six word pairs for Experiment 1.

Word 1	Word 2	r_1	r_2	R	$\frac{B(64)}{B}$
KONG	HONG	0.0145	0.0143	0.925	31.0
OF	AND	0.570	0.554	0.771	40.8
GAMBIA	KIRIBATI	0.0031	0.0028	0.712	26.6
UNITED	STATES	0.062	0.061	0.591	24.8
LOW	PAY	0.045	0.043	0.112	6.8
A	TEST	0.596	0.035	0.052	6.2

For example, “KONG” and “HONG” correspond to two sets of document IDs, which contained “KONG” and “HONG,” respectively.

Figure 3. Mean square errors (MSEs). “M” denotes the original minwise hashing. “Theor.” denotes the theoretical variances $\text{Var}(\hat{R}_b)$ (11) and $\text{Var}(\hat{R}_M)$ (3). The dashed curves, however, are invisible because the empirical MSEs overlap the theoretical variances. At the same k , $\text{Var}(\hat{R}_1) > \text{Var}(\hat{R}_2) > \text{Var}(\hat{R}_3) > \text{Var}(\hat{R}_M)$. However, $\hat{R}_1/\hat{R}_2/\hat{R}_3$ only requires 1/2/3 bits per sample, while $\hat{R}_M$ may require 64 bits.

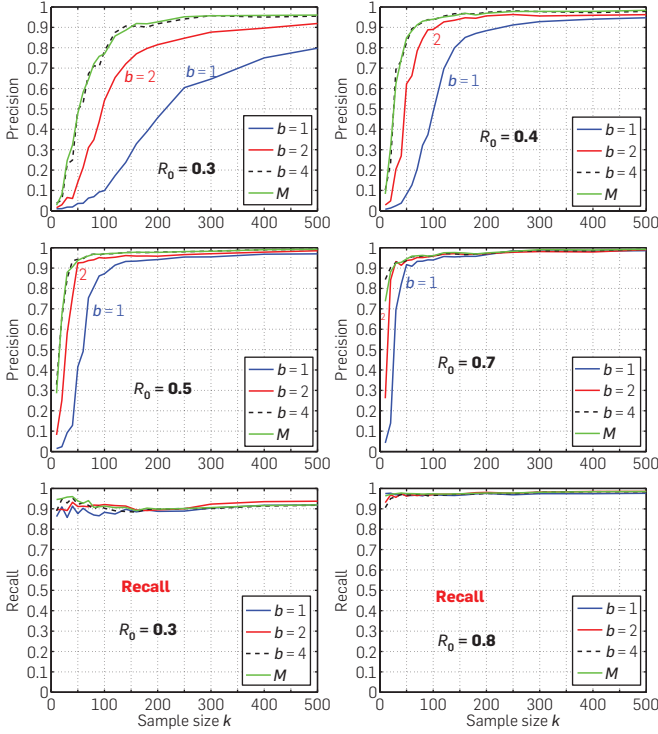


formula (11) is accurate and $\hat{R}_b$ is indeed unbiased (otherwise, the MSE will differ from the variance).

3.2. Experiment 2

To illustrate the improvements by the use of b -bit minwise hashing on a real-life application, we conducted a duplicate detection experiment using a corpus of 10^6 news documents. The dataset was crawled as part of the BLEWS project at Microsoft.⁹ We computed pairwise resemblances for all documents and retrieved document pairs with resemblance R larger than a threshold R_0 . We estimate the resemblances using $\hat{R}_b$ with $b = 1, 2, 4$ bits and the original minwise hashing. Figure 4 presents the precision and recall curves. The recall values (bottom two panels in Figure 4) are all very high and do not differentiate the estimators.

Figure 4. The task is to retrieve news article pairs with resemblance $R \geq R_0$. The recall curves cannot differentiate estimators. The precision curves are more interesting. When $R_0 = 0.4$, to achieve a precision = 0.80, the estimators $\hat{R}_M, \hat{R}_4, \hat{R}_2$, and $\hat{R}_1$ require $k = 50, 50, 75$, and 145, respectively, indicating $\hat{R}_4, \hat{R}_2$, and $\hat{R}_1$, respectively, improve $\hat{R}_M$ (assuming 64 bits per sample) by 16-, 21.4-, and 22-fold. The improvement becomes larger as R_0 increases.



The precision curves for $\hat{R}_4$ (using 4 bits per sample) and $\hat{R}_M$ (assuming 64 bits per sample) are almost indistinguishable, suggesting a 16-fold improvement in space using $b = 4$.

When using $b = 1$ or 2, the space improvements are normally around 20- to 40-fold, compared to $\hat{R}_M$ (assuming 64 bits per sample), especially for achieving high precision.

4. COMPARISONS WITH HAMMING DISTANCE ALGORITHMS

Closely related to the resemblance, the *hamming distance* H is another important similarity measure. In the context of hamming distance, a set $S_i \subseteq \Omega = \{0, 1, \dots, D-1\}$ is mapped to a D -dimensional binary vector Y_i : $Y_{it} = 1$, if $t \in S_i$ and 0 otherwise. The hamming distance between Y_1 and Y_2 is

$$H = \sum_{i=0}^{D-1} [Y_{1i} \neq Y_{2i}] = |S_1 \cup S_2| - |S_1 \cap S_2| = f_1 + f_2 - 2a,$$

i.e., $H/D = r_1 + r_2 - 2s$.

Thus, one can apply b -bit minwise hashing to estimate H , by converting the estimated resemblance $\hat{R}_b$ (9) to $\hat{H}_b$:

$$\hat{H}_b = f_1 + f_2 - 2 \frac{\hat{R}_b}{1 + \hat{R}_b} (f_1 + f_2) = \frac{1 - \hat{R}_b}{1 + \hat{R}_b} (f_1 + f_2). \quad (16)$$

The variance of $\hat{H}_b$ can be computed from $\text{Var}(\hat{R}_b)$ (11) by the “delta method” (i.e., $\text{Var}(g(x)) \approx \text{Var}(x) [g'(E(x))]^2$):

$$\text{Var}(\hat{H}_b) = \text{Var}(\hat{R}_b) \frac{4(r_1 + r_2)^2}{(1 + R)^4} D^2 + O\left(\frac{1}{k^2}\right). \quad (17)$$

We will first compare $\hat{H}_b$ with an algorithm based on simple random sampling¹³ and then with another algorithm based on a variant of random projection.¹⁵

4.1. Simple random sampling algorithm

To reduce the storage, we can randomly sample k coordinates from the original data Y_1 and Y_2 in D -dimensions. The samples, denoted by h_1 and h_2 , are k -dimensional bit vectors, from which we can estimate H :

$$\hat{H}_s = \frac{D}{k} \sum_{j=1}^k [h_{1j} \neq h_{2j}], \quad (18)$$

whose variance would be (assuming $k \ll D$)

$$\text{Var}(\hat{H}_s) = \frac{D^2}{k} \left[\frac{H}{D} - \frac{H^2}{D^2} \right]. \quad (19)$$

Comparing the two variances, (17) and (19), we find that the variance of using simple random sampling, that is, $\text{Var}(\hat{H}_s)$, is substantially larger than the variance of using b -bit minwise hashing, that is, $\text{Var}(\hat{H}_b)$, especially when the data is sparse. We consider in practice one will most likely implement the random sampling algorithm by storing only the original locations (coordinates) of the nonzeros in the samples. If we do so, the total bits on average will be $\frac{r_1 + r_2}{2} 64k$ (per set). This motivates us to define the following ratio:

$$G_{s,b} = \frac{\text{Var}(\hat{H}_s) \times \frac{r_1 + r_2}{2} 64k}{\text{Var}(\hat{H}_b) \times bk}. \quad (20)$$

to compare the storage costs. Recall each sample of b -bit minwise hashing requires b bits (i.e., bk bits per set). The following Lemma may help characterize the improvement:

LEMMA 2. If $r_1, r_2 \rightarrow 0$, then $G_{s,b}$ as defined in (20)

$$G_{s,b} \rightarrow \frac{8}{b} \frac{(2^b - 1)(1 + R)^3}{1 + (2^b - 1)R}. \quad (21)$$

In other words, for small r_1, r_2 , $G_{s,b} \approx \frac{8}{b} (2^b - 1)$; if $R \approx 0$; and $G_{s,b} \approx \frac{64}{b} \frac{2^b - 1}{2^b}$, if $R \approx 1$. Figure 5 plots $G_{s,b} = 1$, verifying the substantial improvement of b -bit minwise hashing over simple random sampling (often 10- to 30-fold).

4.2. Random projection + modular arithmetic

An interesting 1-bit scheme was developed in Kushilevitz et al.¹⁵ using random projection followed by modular arithmetic. A random matrix $U \in \mathbb{R}^{D \times k}$ is generated with entries being i.i.d. samples u_{ij} from a binomial distribution: $u_{ij} = 1$ with probability $\frac{\beta}{2}$ and $u_{ij} = 0$ with probability $1 - \frac{\beta}{2}$. Let $v_1 = Y_1 \times U \pmod{2}$ and $v_2 = Y_2 \times U \pmod{2}$. Kushilevitz et al.¹⁵ showed that

$$E_\beta = \Pr(v_{1,j} \neq v_{2,j}) = \frac{1}{2} (1 - [1 - \beta]^H) \quad (22)$$

which allows us to estimate the hamming distance H by

$$\hat{H}_{rp,\beta} = \frac{\log(1-2\hat{E}_\beta)}{\log(1-\beta)} \quad (23)$$

We calculate the variance of $\hat{H}_{rp,\beta}$ to be

$$\text{Var}(\hat{H}_{rp,\beta}) = \frac{1}{k} \frac{4E_\beta(1-E_\beta)}{(1-2E_\beta)^2 \log^2(1-\beta)} + O\left(\frac{1}{k^2}\right) \quad (24)$$

which suggests that the performance of this 1-bit scheme might be sensitive to β that must be predetermined for all sets at the processing time (i.e., it cannot be modified in the estimation phrase for a particular pair). Figure 6 provides the “optimal” β (denoted by β^*) values (as function of H) by numerically minimizing the variance (24).

It is interesting to compare this random projection-based 1-bit scheme with our b -bit minwise hashing using the following ratio of their variances:

$$G_{rp,b,\beta} = \frac{\text{Var}(\hat{H}_{rp,\beta}) \times k}{\text{Var}(\hat{H}_b) \times bk}. \quad (25)$$

Figure 5. $G_{s,b=1}$ as defined in (20) for illustrating the improvement of b -bit minwise hashing over simple random sampling. We consider $r_1 = 10^{-4}, 0.1, 0.5, 0.9$, r_2 ranging from $0.1 r_1$ to $0.9 r_1$ and s from 0 to r_2 . Note that $r_1 + r_2 - s \leq 1$ has to be satisfied.

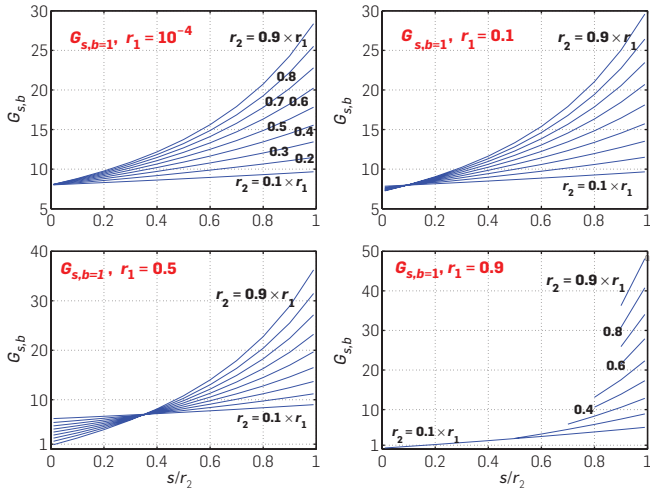


Figure 6. Left panel: the β^* values at which the smallest variances (24) are attained. Right panel: the corresponding optimal variances.

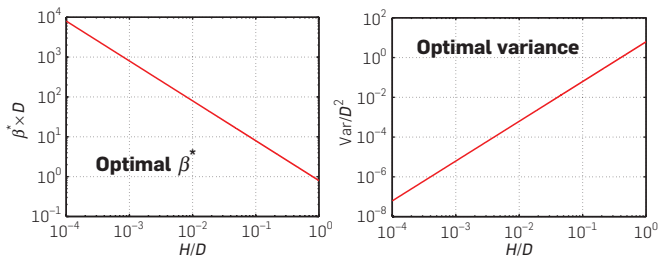


Figure 7 shows that if it is possible to choose the optimal β^* for random projection, one can achieve good performance, similar to (or even better than) b -bit minwise hashing.

The problem is that we must choose the same β for all sets. Figure 8 presents a typical example, which uses $H^*/D = 10^{-4}$ to compute the “optimal” β for a wide range of (r_1, r_2, s) values. The left bottom panel illustrates that when $r_1 = 10^{-4}$ using this particular choice of β results in fairly good performance compared to b -bit minwise hashing. (Recall $H/D = r_1 + r_2 - 2s$.) As soon as the true H substantially deviates from the guessed H^* , the performance of $\hat{H}_{rp,\beta}$ using random projection degrades dramatically.

There is one more issue. At the optimal $\beta^*(H)$, our calculations show that the probability (22) $E_\beta \approx 0.2746$. However, if the chosen $\beta > \beta^*(H)$, then E_β may approach $1/2$. As $\hat{E}_\beta$ is random, it is likely that the observed $\hat{E}_\beta > 1/2$, that is, $\log(1 - 2\hat{E}_\beta)$

Figure 7. $G_{rp,b=1,\beta^*}$ as defined in (25) for comparing $\hat{H}_b$ with $\hat{H}_{rp,\beta^*}$. For each combination of r_1, r_2, s , we computed and used the optimal β^* for the variance.

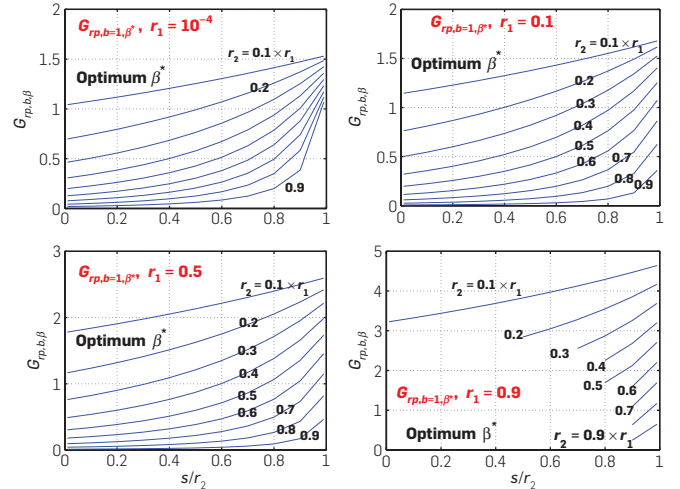


Figure 8. $G_{rp,b=1,\beta}$ (25) computed by using the fixed β , which is the optimal β when $H = H^* = 10^{-4} D$.

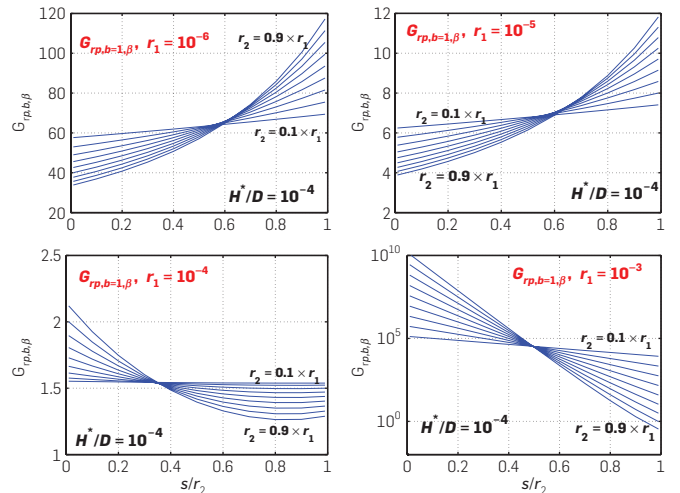
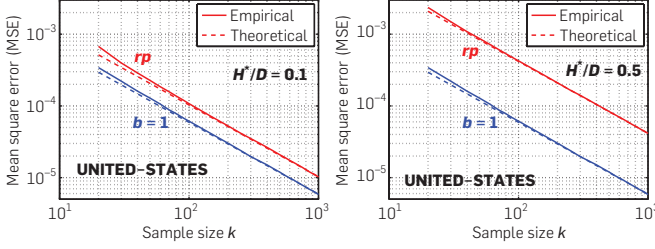


Figure 9. The exact H for this pair of sets is 0.0316. We use the optimal β at $H^*/D = 0.1$ (left panel) and $H^*/D = 0.5$ (right panel). Compared to $\hat{H}_{b=1}$, the MSEs of $\hat{H}_{rp,\beta}$ (labeled by “rp”) are substantially larger. The theoretical variances (dashed lines) (24) and (17), essentially overlap the empirical MSEs.



becomes undefined in (23). Thus, it is safer to “overestimate” H when choosing β . When we have a large collection of sets, this basically means the chosen β will be very different from its optimal value for most pairs.

Finally, Figure 9 provides an empirical study as a sanity check that the variance formula (24) is indeed accurate and that, if the guessed H for selecting β deviates from the true H , then the random projection estimator $\hat{H}_{rp,\beta}$ exhibits much larger errors than the b -bit hashing estimator $\hat{H}_b$.

5. IMPROVEMENT BY COMBINING BITS

Our theoretical and empirical results have confirmed that, when the resemblance R is reasonably high, even a single bit per sample may contain sufficient information for accurately estimating the similarity. This naturally leads to the conjecture that, when R is close to 1, one might further improve the performance by looking at a combination of multiple bits (i.e., “ $b < 1$ ”). One simple approach is to combine two bits from two permutations using XOR ($\oplus$) operations.

Recall $e_{1,1,\pi}$ denotes the lowest bit of the hashed value under π . Theorem 1 has proved that

$$P_1 = \Pr(e_{1,1,\pi} = e_{2,1,\pi}) = C_{1,1} + (1 - C_{2,1})R.$$

Consider two permutations π_1 and π_2 . We store

$$x_1 = e_{1,1,\pi_1} \oplus e_{1,1,\pi_2}, \quad x_2 = e_{2,1,\pi_1} \oplus e_{2,1,\pi_2}$$

Then $x_1 = x_2$ either when $e_{1,1,\pi_1} = e_{2,1,\pi_1}$ and $e_{1,1,\pi_2} = e_{2,1,\pi_2}$, or, when $e_{1,1,\pi_1} \neq e_{2,1,\pi_1}$ and $e_{1,1,\pi_2} \neq e_{2,1,\pi_2}$. Thus,

$$T = \Pr(x_1 = x_2) = P_1^2 + (1 - P_1)^2, \quad (26)$$

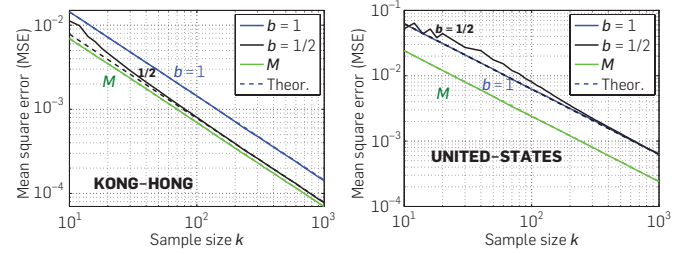
which is a quadratic equation with a solution:

$$\hat{R}_{1/2} = \frac{\sqrt{\max\{2\hat{T} - 1, 0\}} + 1 - 2C_{1,1}}{2 - 2C_{2,1}}. \quad (27)$$

This estimator is slightly biased at small sample size k . We use $\hat{R}_{1/2}$ to indicate that two bits are combined into one (but each sample is still stored using 1 bit). The asymptotic variance of $\hat{R}_{1/2}$ can be derived to be

$$\text{Var}(\hat{R}_{1/2}) = \frac{1}{k} \frac{T(1-T)}{4(1-C_{2,1})^2(2T-1)} + O\left(\frac{1}{k^2}\right). \quad (28)$$

Figure 10. MSEs for comparing $\hat{R}_{1/2}$ (27) with $\hat{R}_1$ and $\hat{R}_m$. Due to the bias of $\hat{R}_{1/2}$, the theoretical variances $\text{Var}(\hat{R}_{1/2})$, that is, (28), deviate from the empirical MSEs when k is small.



Interestingly, as $R \rightarrow 1$, $\hat{R}_{1/2}$ does twice as well as $\hat{R}_1$:

$$\lim_{R \rightarrow 1} \frac{\text{Var}(\hat{R}_1)}{\text{Var}(\hat{R}_{1/2})} = \lim_{R \rightarrow 1} \frac{2(1-2P_1)^2}{(1-P_1)^2 + P_1^2} = 2. \quad (29)$$

On the other hand, $\hat{R}_{1/2}$ may not be good when R is not too large. For example, one can numerically show that

$$\text{Var}(\hat{R}_1) < \text{Var}(\hat{R}_{1/2}), \quad \text{if } R < 0.5774, r_1, r_2 \rightarrow 0$$

Figure 10 plots the empirical MSEs for two-word pairs in Experiment 1, for $\hat{R}_{1/2}$, $\hat{R}_1$, and $\hat{R}_m$. For the highly similar pair, “KONG-HONG,” $\hat{R}_{1/2}$ exhibits superior performance compared to $\hat{R}_1$. For “UNITED-STATES,” whose $R = 0.591$, $\hat{R}_{1/2}$ performs similarly to $\hat{R}_1$.

In summary, for applications which care about very high similarities, combining bits can reduce storage even further.

6. COMPUTATIONAL IMPROVEMENTS

When computing set similarity for large sets of samples, the key operation is determining the number of identical b -bit samples. While samples for values of b that are multiples of 16 bits can easily be compared using a single machine instruction, efficiently computing the overlap between b -bit samples for small b is less straightforward. In the following, we will describe techniques for computing the number of identical b -bit samples when these are packed into arrays $A_l[1, \dots, \frac{k \cdot b}{w}]$, $l = 1, 2$ of w -bit words. To compute the number of identical b -bit samples, we iterate through the arrays; for each offset h , we first compute $v = A_1[h] \oplus A_2[h]$. Now, the number of b -bit blocks in u that contain only 0s corresponds to the number of identical b -bit samples.

The case of $b = 1$ corresponds to the problem of counting the number of 0-bits in a word. We tested a number of different methods and found the fastest approach to be precomputing an array `bits`[1, ..., 2¹⁶] such that `bits`[t] corresponds to the number of 0-bits in the binary representation of t and using lookups into this array. This approach extends to $b > 1$ as well.

To evaluate this approach we timed a tight loop computing the number of identical samples in two arrays of b -bit hashes covering a total of 1.8 billion 32-bit words (using a 64-bit Intel 6600 Processor). Here, the 1-bit hashing requires 1.67× the time that the 32-bit minwise hashing requires (1.73× when comparing to 64-bit minwise hashing). The

results were essentially identical for $b=2, 4, 8$. Given that, when $R > 0.5$, we can gain a storage reduction of 21.3-fold, we expect the resulting improvement in computational efficiency to be $21.3/1.67 = 12.8$ -fold in the above setup.

7. EXTENSIONS AND APPLICATIONS

7.1. Three-way resemblance

Many applications in data mining or data cleaning require not only estimates of two-way, but also of multi-way similarities. The original minwise hashing naturally extends to multi-way resemblance. In Li et al.,¹⁹ we extended b -bit minwise hashing to estimate three-way resemblance. We developed a highly accurate, but complicated estimator, as well as a much simplified estimator suitable for sparse data. Interestingly, at least $b \geq 2$ bits are needed in order to estimate three-way resemblance. Similar to the two-way case, b -bit minwise hashing can result in an order-of-magnitude reduction in the storage space required for a given estimation accuracy when testing for moderate to high similarity.

7.2. Large-scale machine learning

A different category of applications for b -bit minwise hashing is machine learning on very large datasets. For example, one of our projects²⁰ focuses on linear support vector machines (SVM). We were able to show that the resemblance matrix, the minwise hashing matrix, and the b -bit minwise hashing matrix are all positive definite matrices (kernels), and we integrated b -bit minwise hashing with linear SVM. This allows us to significantly speed up training and testing times with almost no loss in classification accuracy for many practical scenarios. In addition, this provides an elegant solution to the problem of SVM training in scenarios where the training data cannot fit in memory.

Interestingly, the technique we used for linear SVM essentially provides a universal strategy for integrating b -bit minwise hashing with many other learning algorithms, for example, logistic regression.

7.3. Improving estimates by maximum likelihood estimators

While b -bit minwise hashing is particularly effective in applications which mainly concern sets of high similarities (e.g., $R > 0.5$), there are other important applications in which not just pairs of high similarities matter. For example, many learning algorithms require all pairwise similarities and it is expected that only a small fraction of the pairs are similar. Furthermore, many applications care more about *containment* (e.g., which fraction of one set is contained in another set) than the *resemblance*. In a recent technical report (<http://www.stat.cornell.edu/~li/b-bit-hashing/AccurateHashing.pdf>), we showed that the estimators for minwise hashing and b -bit minwise hashing used in the current practice can be systematically improved and the improvements are most significant for set pairs of low resemblance and high containment.

For minwise hashing, instead of only using $\Pr(z_1 = z_2)$, where z_1 and z_2 are two hashed values, we can combine it

with $\Pr(z_1 < z_2)$ and $\Pr(z_1 > z_2)$ to form a three-cell multinomial estimation problem, whose maximum likelihood estimator (MLE) is the solution to a cubic equation. For b -bit minwise hashing, we formulate a $2^b \times 2^b$ -cell multinomial problem, whose MLE requires a simple numerical procedure.

7.4. The new LSH family

Applications such as near neighbor search, similarity clustering, and data mining will significantly benefit from b -bit minwise hashing. It is clear that b -bit minwise hashing will significantly improve the efficiency of simple linear algorithms (for near neighbor search) or simple quadratic algorithms (for similarity clustering), when the key bottleneck is main-memory throughput.

Techniques based on $LSH^{1, 5, 13}$ have been successfully used to achieve sub-linear (for near neighbor search) or sub-quadratic (for similarity clustering) performance. It is interesting that b -bit minwise hashing is a new family of LSH; hence, in this section, we would like to provide more theoretical properties in the context of LSH and approximate near neighbor search.

Consider a set S_1 . Suppose there exists another set S_2 whose resemblance distance $(1-R)$ from S_1 is at most d_0 , that is, $1-R \leq d_0$. The goal of c -approximate d_0 -near neighbor algorithms is to return sets (with high probability) whose resemblance distances from S_1 are at most $c \times d_0$ with $c > 1$.

Recall z_1 and z_2 denote the minwise hashed values for sets S_1 and S_2 , respectively. The performance of the LSH algorithm depends on the difference (gap) between the following $P^{(1)}$ and $P^{(2)}$ (respectively corresponding to d_0 and cd_0):

$$\begin{aligned} \text{If } 1-R \leq d_0, \text{ then } R = \Pr(z_1 = z_2) &\geq 1-d_0 = P^{(1)}. \\ \text{If } 1-R \geq cd_0, \text{ then } R = \Pr(z_1 = z_2) &\geq 1-cd_0 = P^{(2)}. \end{aligned}$$

A larger gap between $P^{(1)}$ and $P^{(2)}$ implies a more efficient LSH algorithm. The following “ ρ ” value (ρ_M for minwise hashing) characterizes the gap:

$$\rho_M = \frac{\log 1/P^{(1)}}{\log 1/P^{(2)}} = \frac{\log(1-d_0)}{\log(1-cd_0)}. \quad (30)$$

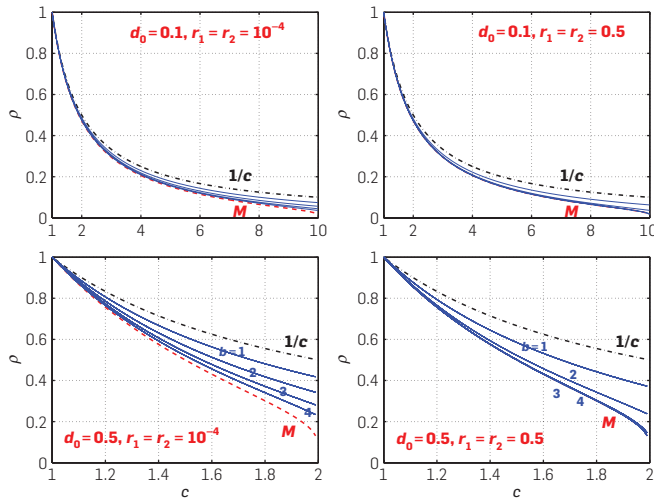
A smaller ρ (i.e., larger difference between $P^{(1)}$ and $P^{(2)}$) leads to a more efficient LSH algorithm and $\rho < \frac{1}{c}$ is particularly desirable.^{1, 13} The general LSH theoretical result tells us that the query time for c -approximate d_0 -near neighbor is dominated by $O(N^\rho)$ distance evaluations, where N is the total number of sets in the collection.

Recall P_b , as defined in (5), denotes the collision probability for b -bit minwise hashing. The ρ_b value for c -approximate d_0 -near neighbor search can be computed as follows:

$$\begin{aligned} 1-R \leq d_0 &\Rightarrow P_b \geq C_{1,b} + (1-C_{2,b})(1-d_0) \\ 1-R \geq cd_0 &\Rightarrow P_b \geq C_{1,b} + (1-C_{2,b})(1-cd_0) \\ \rho_b &= \frac{\log(C_{1,b} + (1-C_{2,b})(1-d_0))}{\log(C_{1,b} + (1-C_{2,b})(1-cd_0))}. \end{aligned} \quad (31)$$

Figure 11 suggests that b -bit minwise hashing can potentially achieve very similar ρ values compared to the original minwise hashing, when the applications care mostly about

Figure 11. ρ_M and ρ_b defined in (30) and (31) for measuring the potential performance of LSH algorithms. “M” denotes the original minwise hashing.



highly similar sets (e.g., $d_0 = 0.1$, the top panels of Figure 11), even using merely $b = 1$. If the applications concern sets that are not necessarily highly similar (e.g., $d_0 = 0.5$, the bottom panels), using $b = 3$ or 4 will still have similar ρ values as using the original minwise hashing.

We expect that these theoretical properties regarding the ρ values will potentially be useful in future work. We are currently developing new variants of LSH algorithms for near neighbor search based on b -bit minwise hashing.

Subsequent documents will be made available at www.stat.cornell.edu/~li/b-bit-hashing, which is a repository for maintaining the papers and technical reports related to b -bit minwise hashing.

8. CONCLUSION

Minwise hashing is a standard technique for efficiently estimating set similarity in massive datasets. In this paper, we gave an overview of *b-bit minwise hashing*, which modifies the original scheme by storing the lowest b bits of each hashed value. We proved that, when the similarity is reasonably high (e.g., resemblance ≥ 0.5), using $b = 1$ bit per hashed value can, even in the worst case, gain a 21.3-fold improvement in storage space (at similar estimation accuracy), compared to storing each hashed value using 64 bits. As many applications are primarily interested in identifying duplicates of reasonably similar sets, these improvements can result in substantial reduction in storage (and consequently computational) overhead in practice.

We also compared our scheme to other approaches that map the hashed objects to single bits, both in theory as well as experimentally.

Our proposed method is simple and requires only minimal modification to the original minwise hashing algorithm. It can be used in the context of a number of different applications, such as duplicate detection, clustering, similarity search, and machine learning, and we expect that it will be adopted in practice.

Acknowledgment

This work is supported by NSF (DMS-0808864), ONR (YIP-N000140910911), and a grant from Microsoft. We thank the Board Members for suggesting a direct comparison with Kushilevitz et al.¹⁵

References

- Andoni, A., Indyk, P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM* 51 (2008), 117–122.
- Broder, A.Z. On the resemblance and containment of documents. In *The Compression and Complexity of Sequences* (Positano, Italy, 1997), 21–29.
- Broder, A.Z., Charikar, M., Frieze, A.M., Mitzenmacher, M. Min-wise independent permutations. *J. Comput. Syst. Sci.* 60, 3 (2000), 630–659.
- Broder, A.Z., Glassman, S.C., Manasse, M.S., Zweig, G. Syntactic clustering of the web. In *WWW* (Santa Clara, CA, 1997), 1157–1166.
- Charikar, M.S. Similarity estimation techniques from rounding algorithms. In *STOC* (Montreal, Quebec, Canada, 2002), 380–388.
- Cohen, E., Datar, M., Fujiwara, S., Gionis, A., Indyk, P., Motwani, R., Ullman, J.D., Yang, C. Finding interesting associations without support pruning. *IEEE Trans. Knowl. Data Eng.* 13, 1 (2001), 64–78.
- Fetterly, D., Manasse, M., Najork, M., Wiener, J.L. A large-scale study of the evolution of web pages. In *WWW* (Budapest, Hungary, 2003), 669–678.
- Forman, G., Eshghi, K., Suemondt, J. Efficient detection of large-scale redundancy in enterprise file systems. *SIGOPS Oper. Syst. Rev.* 43, 1 (2009), 84–91.
- Gamon, M., Basu, S., Belenko, D., Fisher, D., Hurst, M., König, A.C. Blews: Using blogs to provide context for news articles. In *AAAI Conference on Weblogs and Social Media* (Redmond, WA, 2008).
- Gionis, A., Gupopulos, D., Koudas, N. Efficient and tunable similar set retrieval. In *SIGMOD* (Santa Barbara, CA, 2001), 247–258.
- Goemans, M.X., Williamson, D.P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM* 42, 6 (1995), 1115–1145.
- Indyk, P. A small approximately min-wise independent family of hash functions. *J. Algorithms* 38, 1 (2001), 84–90.
- Indyk, P., Motwani, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC* (Dallas, TX, 1998), 604–613.
- Itoh, T., Takei, Y., Tarui, J. On the sample size of k -restricted min-wise independent permutations and other k -wise distributions. In *STOC* (San Diego, CA, 2003), 710–718.
- Kushilevitz, E., Ostrovsky, R., Rabani, Y. Efficient search for approximate nearest neighbor in high dimensional spaces. In *STOC* (Dallas, TX, 1998), 614–623.
- Li, P., Church, K.W. A sketch algorithm for estimating two-way and multi-way associations. *Comput. Linguist.* 33, 3 (2007), 305–354 (Preliminary results appeared in HLT/EMNLP 2005).
- Li, P., Church, K.W., Hastie, T.J. One sketch for all: Theory and applications of conditional random sampling. In *NIPS* (Vancouver, British Columbia, Canada, 2008) (Preliminary results appeared in NIPS 2006).
- Li, P., Hastie, T.J., Church, K.W. Improving random projections using marginal information. In *COLT* (Pittsburgh, PA, 2006), 635–649.
- Li, P., König, A.C., Gui, W. b-Bit minwise hashing for estimating three-way similarities. In *NIPS* (Vancouver, British Columbia, Canada, 2010).
- Li, P., Moore, J., König, A.C. b-Bit minwise hashing for large-scale linear SVM. Technical report, 2011. <http://www.stat.cornell.edu/~li/b-bit-hashing/HashingSVM.pdf>
- Cherkasova, L., Eshghi, K., Morrey III, C.B., Tucek, J., Veitch, A. Applying Syntactic similarity algorithms for enterprise information management. In *KDD* (Paris, France, 2009), 1087–1096.
- Manasse, M., McSherry, F., Talwar, K. Consistent weighted sampling. Technical Report MSR-TR-2010-73, Microsoft Research, 2010.
- Pandey, S., Broder, A., Chierichetti, F., Josifovski, V., Kumar, R., Vassilvitskii, S. Nearest-neighbor caching for content-match applications. In *WWW* (Madrid, Spain, 2009), 441–450.
- Rajaraman, A., Ullman, J. *Mining of Massive Datasets*. <http://i.stanford.edu/ullman/mmds.html>
- Urvoy, T., Chauveau, E., Filoche, P., Laverigne, T. Tracking web spam with html style similarities. *ACM Trans. Web* 2, 1 (2008), 1–28.

Ping Li, Department of Statistical Science, Faculty of Computing and Information Science, Cornell University, Ithaca, NY.

Arnd Christian König, Microsoft Research, Microsoft Corporation, Redmond, WA.

Technical Perspective

Skinroducing the Future

By Scott Klemmer

TWO CRITICAL GOALS for mobile devices seem intrinsically in conflict. For carrying, the smaller the better. Yet for interacting, more real estate is generally better. This tension makes for great sketch comedy: absurdly tiny phones that are impossible to interact with, or giant touch screens that are back-breaking to carry.

Chris Harrison and colleagues may have the last laugh. What if the body itself could be an input surface? The average body surface area of an adult (1.73 m², according to Wikipedia) is 400 times greater than a touch-screen phone (0.004 m², by my estimate). Sailors and tattoo parlors have long seen opportunities for the body as a display. Skinput adds interactivity via a pico-projector and vibration sensing: tap an image projected on your arm, and the resulting arm vibrations control an application.

How is this a harbinger of a fundamental change, and what makes its appeal more than...skin deep? One powerful contribution of the graphical interface is input on output: direct manipulation. In the coming years, pervasive direct manipulation—where Skinput is an early foray—will likely mature and become a major force. Every surface is a potential site for both projection and input, breaking the picture frame of the desktop interface. Phenomenologically, the change induced by ubiquitous projection is that the computer disappears by seamlessly weaving computing into the physical world. Skinput showcases three key tools for building disappearing computers: rich sensing, machine learning, and flexible projection. Systems like Skinput that flexibly sense body pose, movements, and gestures illustrate how interaction design benefits from innovating both software and hardware.

Does Skinput spell doom for touchscreens? Maybe not. The discourse around interactive systems often

frames technical evolution in terms of “generations” of interfaces. That there were punch cards. Then the terminal. Then the mouse and graphical interface. Each supplanting the previous one. On this view, the logical question to ask is: “What’s next?” With input, this is often phrased as: “What will replace the keyboard and mouse?” Of course, different paradigms are good for different tasks. While new tools reshape the landscape and supplant some old tools, people benefit from a diverse interface ecosystem. Today, one’s computing likely spans direct manipulation, gestures, keyboard commands, and search. The screwdriver does not obviate the value of a hammer. In some cases, ubiquitous projection and sensing will enable fluid interactive experiences. In other cases, like text messaging, technologies can become powerful and pervasive even though the *interface itself* is quite primitive.

Isn’t an interactive forearm a little ridiculous? (“Come on! People won’t *really* interact this way.”) Watch the video (<http://research.microsoft.com/cue/skinput>); it’s amazing. Also, Skinput is an early prototype in two important ways. First, it’s a sketch of a possible future: suggestive rather than complete. The viewer’s imagination is key to filling in the details. Menu selection is just one of many things this approach enables. Second, it instantiates a time-honored computer science research strategy: Build the bulky, expensive thing now to understand what it’s like to live in a world with that technology; future revisions will get smaller and cheaper. It pays to be broad when prototyping the future. Explore 10 future realities, and if any come to pass, that’s a win. Furthermore, research can succeed by inspirational value beyond its direct utility. Expanding the input repertoire will pay broad dividends.

With the forearm as the input surface, Skinput is very literally embod-

ied interaction. Embodied interactions can offer incredible power by leveraging the amazing implicit intelligence of the human perceptuo-motor system. At the same time, bodies have clear physical limitations; you get tired holding your arm still. Unless the goal is to get into better shape, such mundane factors impose real constraints on what interfaces you’re likely to actually adopt.

One enabling insight that can’t be ignored: the tap sensing is really creative. (By which I mean, “I wish I’d thought of that.”) Tapping on skin yields both transverse waves (ripples) and longitudinal waves (bone vibration). These subtle waves generally elude people’s notice, but high-frequency sensors can track them reliably. (So can high-speed cameras—another reason to watch the video.) The authors use piezoelectric sensors to measure the deformation. Today, such sensors are commonly used as guitar pick-ups. Increasingly diverse—and cheap—sensing technologies make this a really exciting time for inventing new interactive systems.

Research probes like Skinput currently require building bespoke systems. The next step is to flesh out the design space of alternatives, understand their trade-offs, and build theories. This exploration will require tools (and curricula) for rapidly and flexibly creating interfaces with rich sensing and machine learning. The DIY and research communities have made great strides here, and much exciting work remains.

Interactive tattoos?

That remains future work. 

Scott Klemmer (srk@cs.stanford.edu) is an assistant professor of computer science at Stanford University, where he co-directs the Human-Computer Interaction Group.

© 2011 ACM 0001-0782/11/08 \$10.00

Skinput: Appropriating the Skin as an Interactive Canvas

By Chris Harrison, Desney Tan, and Dan Morris

Abstract

***Skinput* is a technology that appropriates the skin as an input surface by analyzing mechanical vibrations that propagate through the body. Specifically, we resolve the location of finger taps on the arm and hand using a novel sensor array, worn as an armband. This approach provides an on-body finger input system that is always available, naturally portable, and minimally invasive. When coupled with a pico-projector, a fully interactive graphical interface can be rendered directly on the body. To view video of Skinput, visit <http://cacm.acm.org>.**

1. INTRODUCTION

Devices with significant computational power and capability can now be easily carried with us. These devices have tremendous potential to bring the power of information, computation, creation, and communication to a wider audience and to more aspects of our lives. However, this potential raises new challenges for interaction design. For example, miniaturizing devices has simultaneously reduced their interactive surface area. This has led to diminutive screens, cramped keyboards, and tiny jog wheels, all of which impose restrictions that diminish usability and prevent us from realizing the full potential of mobile computing. Consequently, mobile devices are approaching the computational capabilities of desktop computers, but are hindered by a human-computer I/O bottleneck.

Critically, this is a problem we cannot engineer ourselves out of. While we can make computer processors faster, LCD screens thinner, and hard drives larger, we cannot add surface area without increasing size—it is a physical constraint. This has trapped us in a device size paradox: we want more usable devices, but are unwilling to sacrifice the benefits of small size and mobility. In response, designers have walked a fine line, trying to strike a balance between usability and mobility.

One promising approach to mitigate this is to appropriate surface area from the environment for interactive purposes. This can offer larger interactive surface area with no increase in device size. For example, Harrison and Hudson⁷ describe a technique that allows (small) mobile devices to turn (large) tables into gestural finger input canvases. However, tables are not always present, and in a mobile context, users are unlikely to want to carry appropriated surfaces with them (at this point, one might as well just have a larger device). However, there is one surface that has been previously overlooked as an input surface, and one that happens to always travel with us: our skin.

Appropriating the human body as an input device is

appealing not only because we have roughly 2 m² of surface area, but also because much of it is easily accessible by our hands (e.g., arms, upper legs, torso). Furthermore, proprioception—our sense of how our body is configured in three-dimensional space—allows us to accurately interact with our bodies in an eyes-free manner. For example, we can readily flick each of our fingers, touch the tip of our nose, and clap our hands together without visual assistance.

In this paper, we present our work on *Skinput*—a method that allows the body to be appropriated for finger input using a novel, non-invasive, wearable bio-acoustic sensor. When coupled with a pico-projector, the skin can operate as an interactive canvas supporting both input and graphical output (Figures 1 and 2).

2. RELATED WORK

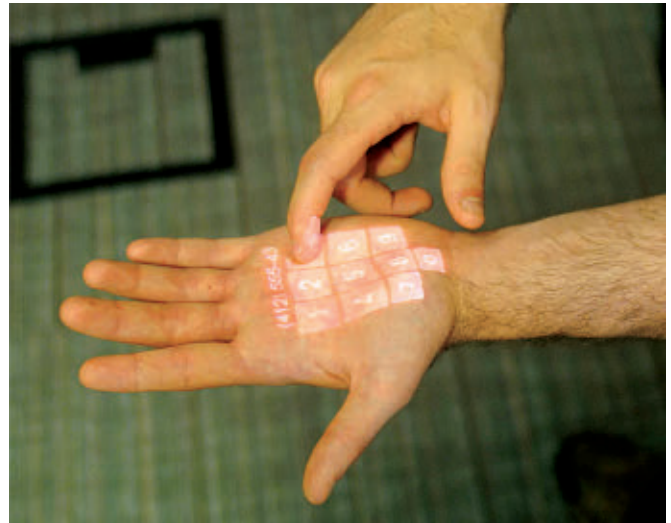
2.1. Always-available input

A primary goal of *Skinput* is to provide an always-available mobile input system—for example, an input system that does not require a user to carry or pick up a device. A number of alternative solutions to this problem have been proposed. Techniques based on computer vision are popular (e.g., Argyros and Lourakis,² Mistry et al.,¹⁶ Wilson,^{24, 25} see Erol et al.⁵ for a recent survey). These, however, are computationally expensive and error prone in mobile scenarios (where, e.g., non-input optical flow is prevalent), or depend on cumbersome instrumentation of the hands to enhance performance. Speech input (e.g., Lakshminath et al.⁹ and Lyons et al.¹¹) is a logical choice for always-available input, but is limited in its precision in unpredictable acoustic environments, suffers from privacy and scalability issues in shared environments, and may interfere with cognitive tasks significantly more than manual interfaces.²²

Other approaches have taken the form of wearable computing. This typically involves a physical input device built in a form considered to be part of one's clothing. For example, glove-based input systems (see Sturman and Zeltzer²³ for a review) allow users to retain most of their natural hand movements, but are cumbersome, uncomfortable, and disruptive to tactile sensation. Post and Orth¹⁹ present a “smart fabric” system that embeds sensors and conductors into fabric, but taking this approach to always-available input necessitates

A previous version of this paper was published in the *Proceedings of the 28th International Conference on Human Factors in Computing Systems* (CHI 2010).

Figure 1. Our sensing armband augmented with a pico-projector; this allows interactive elements to be rendered on the skin.



embedding technology in all clothing, which would be prohibitively complex and expensive.

The SixthSense project¹⁶ proposes a mobile, always-available I/O system by combining projected information with a color-marker-based vision tracking system. This approach is feasible, but suffers from the limitations of vision-based systems discussed above and requires instrumentation of the fingertips. Like SixthSense, we explore the combination of on-body sensing with on-body projection.

2.2. Bio-sensing

Skinput leverages the natural acoustic conduction properties of the human body to provide an input system, and is thus related to previous work in the use of biological signals for computer input. Signals traditionally used for diagnostic medicine, such as heart rate and skin resistance, have been appropriated for assessing a user's emotional state (e.g., Mandryk and Atkins,¹² Mandryk et al.,¹³ Moore and Dua¹⁷). These features are generally subconsciously driven and cannot be controlled with sufficient precision for direct input. Similarly, brain sensing technologies such as electroencephalography (EEG) and functional near-infrared spectroscopy (fNIR) have been used by HCI researchers to assess cognitive and emotional state (e.g., Grimes et al.,⁶ Lee and Tan¹⁰); this work also primarily looked at involuntary signals. In contrast, brain signals have been harnessed as a direct input for use by paralyzed patients (e.g., McFarland et al.¹⁵), but direct brain-computer interfaces (BCIs) still lack the bandwidth required for everyday computing tasks, and require levels of focus, training, and concentration that are incompatible with typical computer interaction.

There has been less work relating to the intersection of finger input and biological signals. Researchers have harnessed the electrical signals generated by muscle activation during normal hand movement through electromyography

(EMG) (e.g., Rosenberg²⁰ and Saponas et al.²¹). At present, however, this approach typically requires expensive amplification systems and the application of conductive gel for effective signal acquisition, which would limit the acceptability of this approach for most users.

The input technology most related to our own is that of Amento et al.,¹ who placed contact microphones on a user's wrist to assess finger movement. However, this work was never formally evaluated and is constrained to finger motions in one hand. The Hambone system⁴ employs a similar approach using piezoelectric sensors, yielding classification accuracies around 90% for four gestures (e.g., raise heels, snap fingers). Performance of false positive rejection remains untested in both systems.

Finally, bone conduction microphones and headphones—now common consumer technologies—represent an additional bio-sensing technology that is relevant to the present work. These leverage the fact that sound frequencies relevant to human speech propagate well through bone. Bone conduction microphones are typically worn near the ear, where they can sense vibrations propagating from the mouth and larynx during speech. Bone conduction headphones send sound through the bones of the skull and jaw directly to the inner ear, bypassing lossy transmission of sound through the air and outer ear. The mechanically conductive properties of human bones are also employed by Zhong et al.²⁷ for transmitting information through the body, such as from an implanted device to an external receiver.

2.3. Acoustic input

Our approach is also inspired by systems that leverage acoustic transmission through (non-body) input surfaces. Paradiso et al.¹⁸ measured the arrival time of a sound at multiple sensors to locate hand taps on a glass window. Ishii et al.⁸ use a similar approach to localize a ball hitting a table, for computer augmentation of a real-world game. Both of

these systems use acoustic time-of-flight for localization, which we explored, but found to be insufficiently robust on the human body, leading to the fingerprinting approach described in this paper.

3. SKINPUT

To expand the range of sensing modalities for always-available input systems, we developed *Skinput*, a novel input technique that allows the skin to be used as a finger input surface, much like a touchscreen. In our prototype system, we choose to focus on the arm, although the technique could be applied elsewhere. This is an attractive area to appropriate as it provides considerable surface area for interaction, including a contiguous and flat area for projection (discussed subsequently). Furthermore, the forearm and hands contain a complex assemblage of bones that increases acoustic distinctiveness of different locations. To capture this acoustic information, we developed a wearable armband that is non-invasive and easily removable (Figure 2).

3.1. Bio-acoustics

When a finger taps the skin, several distinct forms of acoustic energy are produced. Some energy is radiated into the air as sound waves; this energy is not captured by the *Skinput* system. Among the acoustic energy transmitted *through* the arm, the most readily visible are transverse waves, created by the displacement of the skin from a finger impact (Figure 3). When shot with a high-speed camera, these appear as ripples, which propagate outward from the point of contact (like a pebble into a pond). The amplitude of these ripples is correlated to the tapping force and the volume and compliance of soft tissues under the impact area. In general, tapping on soft regions of the arm creates higher-amplitude transverse waves than tapping on boney areas (e.g., wrist, palm, fingers), which have negligible compliance.

In addition to the energy that propagates on the surface of the arm, some energy is transmitted inward, toward the skeleton (Figure 4). These longitudinal (compressive) waves

Figure 2. Skinput rendering a list interface rendered on the arm. Pico-projector and sensing armband highlighted.

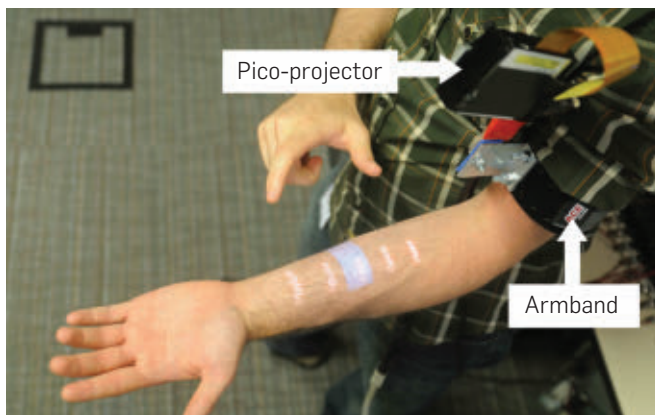


Figure 3. Transverse wave propagation: Finger impacts displace the skin, creating transverse waves (ripples). The sensor is activated as the wave passes underneath it.

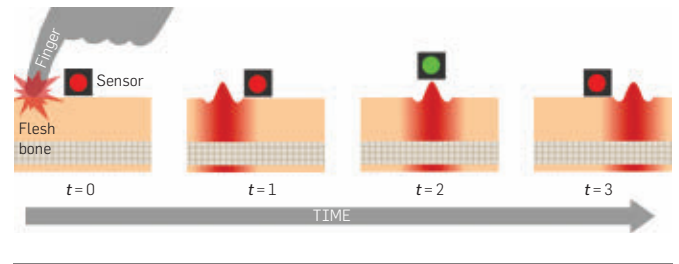
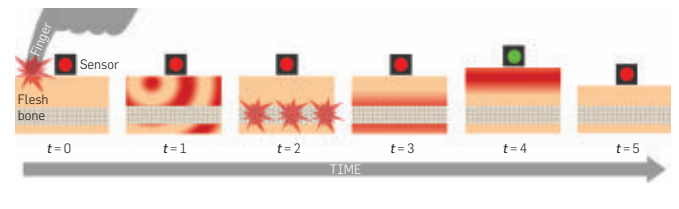


Figure 4. Finger impacts create longitudinal (compressive) waves that cause internal skeletal structures to vibrate. This, in turn, creates longitudinal waves that emanate outward from the bone (along its entire length) toward the skin.



travel through the soft tissues of the arm, exciting the bone, which is much less deformable than the soft tissue but can respond to mechanical excitation by rotating and translating as a rigid body. This excitation vibrates soft tissues surrounding the entire length of the bone, resulting in new longitudinal waves that propagate outward to the skin.

We highlight these two separate forms of conduction—transverse waves moving directly along the arm surface, and longitudinal waves moving into and out of the bone through soft tissues—because these mechanisms carry energy at different frequencies and over different distances. Roughly speaking, higher frequencies propagate more readily through bone than through soft tissue, and bone conduction carries energy over larger distances than soft tissue conduction. While we do not explicitly model the specific mechanisms of conduction, or depend on these mechanisms for our analysis, we do believe the success of our technique depends on the complex acoustic patterns that result from mixtures of these modalities.

Similarly, we also hypothesize that joints play an important role in making tapped locations acoustically distinct. Bones are held together by ligaments, and joints often include additional biological structures such as fluid cavities. This makes joints behave as acoustic filters. In some cases, these may simply dampen acoustics; in other cases, these will selectively attenuate specific frequencies, creating location-specific acoustic signatures. Finally, muscle contraction may also contribute to the vibration patterns recorded by our sensors,¹⁴ including both contraction related to posture maintenance and reflexive muscle movements in response to input taps.

3.2. Armband prototype

Our initial hardware prototype employed an array of tuned

mechanical vibration sensors; specifically small, cantilevered piezoelectric films (MiniSense100, Measurement Specialties, Inc.). By adding small weights to the end of the cantilever, we were able to alter the resonant frequency, allowing each sensing element to be responsive to a unique, narrow, low-frequency band of the acoustic spectrum. Each element was aligned with a particular frequency pilot study shown to be useful in characterizing bio-acoustic input. These sensing elements were packaged into 2 groups of 5–10 sensors in total.

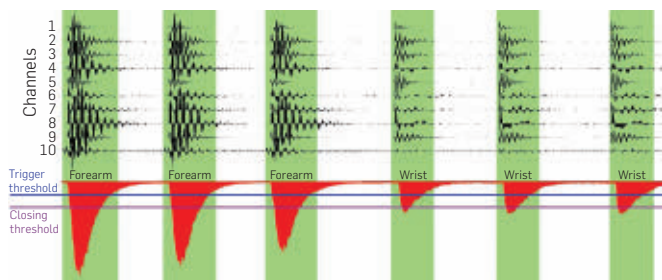
A Mackie Onyx 1200F audio interface was used to digitally capture data from the 10 sensors. Each channel was sampled at 5.5 kHz, a sampling rate that would be considered too low for speech or environmental audio, but was able to represent the relevant spectrum of frequencies transmitted through the arm. This reduced sample rate (and consequently low processing bandwidth) makes our technique readily portable to embedded processors. For example, the ATmega168 processor employed by the Arduino platform can sample analog readings at 77 kHz with no loss of precision, and could therefore provide the full sampling power required for *Skinput* (55 kHz in total).

3.3. Processing

The audio stream was segmented into individual taps using an absolute exponential average of all sensor channels (Figure 5, red waveform). When an intensity threshold was exceeded (Figure 5, upper blue line), the program recorded the timestamp as a potential start of a tap. If the intensity did not fall below a second, independent “closing” threshold (Figure 5, lower purple line) between 100 and 700 ms after the onset crossing (a duration we found to be the common for finger impacts), the event was discarded. If start and end crossings were detected that satisfied these criteria, the acoustic data in that period (plus a 60 ms buffer on either end) was considered an input event (Figure 5, vertical green regions). Although simple, this heuristic proved to be robust.

After an input has been segmented, the waveforms are analyzed. We employ a brute force machine learning approach, computing 186 features in total, many of which are derived combinatorially. For gross information, we

Figure 5. Ten channels of acoustic data generated by three finger taps on the forearm, followed by three taps on the wrist. The exponential average of the channels is shown in red. Segmented input windows are highlighted in green. Note how different sensing elements are activated by the two locations.



include the average amplitude, standard deviation and total (absolute) energy of the waveforms in each channel (30 features). From these, we calculate all average amplitude ratios between channel pairs (45 features). We also include an average of these ratios (1 feature). We calculate a 256-point FFT for all 10 channels, although only the lower 10 values are used (representing the acoustic power from 0 to 193 Hz), yielding 100 features. These are normalized by the highest-amplitude FFT value found on any channel. We also include the center of mass of the power spectrum within the same 0–193 Hz range for each channel, a rough estimation of the fundamental frequency of the signal displacing each sensor (10 features). Subsequent feature selection established the all-pairs amplitude ratios and certain bands of the FFT to be the most predictive features.

These 186 features are passed to a support vector machine (SVM) classifier. A full description of SVMs is beyond the scope of this paper (see Burges³ for a tutorial). Our software uses the implementation provided in the Weka machine learning toolkit.²⁶ It should be noted, however, that other, more sophisticated classification techniques and features could be employed. Thus, the results presented in this paper should be considered a baseline.

Before the SVM can classify input instances, it must first be trained to the user and the sensor position. This stage requires the collection of several examples for each input location of interest. When using *Skinput* to recognize live input, the same 186 acoustic features are computed on-the-fly for each segmented input. These are fed into the trained SVM for classification. We use an event model in our software—once an input is classified, an event associated with that location is instantiated. Any interactive features bound to that event are fired.

4. EXPERIMENT

4.1. Participants

To evaluate the performance of our system, we recruited 13 participants (7 female) from the Seattle area. These participants represented a diverse cross-section of potential ages and body types. Ages ranged from 20 to 56 (mean 38.3), and computed body mass indexes (BMIs) ranged from 20.5 (normal) to 31.9 (obese).

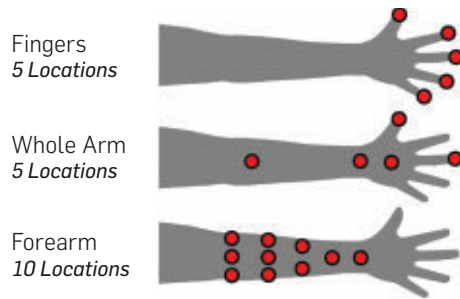
4.2. Experimental conditions

We selected three input groupings from the multitude of possible location combinations to test. We believe that these groupings, illustrated in Figure 6, are of particular interest with respect to interface design, and at the same time, push the limits of our sensing capability. From these three groupings, we derived five different experimental conditions, described below.

4.2.1. Fingers (five locations)

One set of gestures we tested had participants tapping on the tips of each of their five fingers (Figure 6, “Fingers”). The fingers offer interesting affordances that make them compelling to appropriate for input. Foremost, they provide clear, discrete interaction points, which are even

Figure 6. The three input location sets evaluated in the study.



well-named (e.g., “ring finger”). In addition to 5 finger tips, there are 14 knuckles (5 major, 9 minor), which, taken together, could offer 19 readily identifiable input locations on the fingers alone. Second, we have exceptional finger-to-finger dexterity, as demonstrated when we count by tapping on our fingers. Finally, the fingers are linearly ordered, which is potentially useful for interfaces like number entry, magnitude control (e.g., volume), and menu selection.

At the same time, fingers are among the most uniform appendages on the body, with all but the thumb sharing a similar skeletal and muscular structure. This drastically reduces acoustic variation and makes differentiating among them difficult. Additionally, acoustic information must cross as many as five (finger and wrist) joints to reach the forearm, which further dampens signals. For this experimental condition, we thus decided to place the sensor arrays on the forearm, just below the elbow.

Despite these difficulties, pilot experiments showed measurable acoustic differences among fingers, which we theorize is primarily related to finger length and thickness, interactions with the complex structure of the wrist bones, and variations in the acoustic transmission properties of the muscles extending from the fingers to the forearm.

4.2.2. Whole arm (five locations)

Another task investigated the use of five input locations on the forearm and hand: arm, wrist, palm, thumb, and middle finger (Figure 6, “Whole Arm”). We selected these locations for two important reasons. First, they are distinct and named parts of the body (e.g., “wrist”). This allowed participants to accurately tap these locations without training or markings. Additionally, these locations proved to be acoustically distinct during piloting, with the large spatial spread of input points offering further variation.

We used these locations in three different conditions. One condition placed the sensor above the elbow, while another placed it below. This was incorporated into the experiment to measure the accuracy loss across this significant articulation point (the elbow). Additionally, participants repeated the lower placement condition in an eyes-free context: participants were told to close their eyes and face forward, both for training and testing. This condition was included to gauge how well users could

target on-body input locations in an eyes-free context (e.g., driving).

4.2.3. Forearm (10 locations)

In an effort to assess the upper bound of our approach’s sensing resolution, our fifth and final experimental condition used 10 locations on just the forearm (Figure 6, “Forearm”). Not only was this a very high density of input locations (unlike the whole-arm condition), but it also relied on an input surface (the forearm) with a high degree of physical uniformity (unlike, e.g., the hand). We expected that these factors would make acoustic sensing difficult. Moreover, this location was compelling due to its large and flat surface area, as well as its immediate accessibility, both visually and for finger input. Simultaneously, this makes for an ideal projection surface for dynamic interfaces.

To maximize the surface area for input, we placed the sensor above the elbow, leaving the entire forearm free. Rather than naming the input locations, as was done in the previously described conditions, we employed small, colored stickers to mark input targets. This was both to reduce confusion (since locations on the forearm do not have common names) and to increase input consistency. As mentioned previously, we believe the forearm is ideal for projected interface elements; the stickers served as low-tech placeholders for projected buttons.

4.3. Design and setup

We employed a within-subjects design, with each participant performing tasks in each of the five conditions in randomized order: five fingers with sensors below elbow; five points on the whole arm with the sensors above the elbow; the same points with sensors below the elbow, both sighted and blind; and 10 marked points on the forearm with the sensors above the elbow.

Participants were seated in a conventional office chair, in front of a desktop computer that presented stimuli. For conditions with sensors below the elbow, we placed the armband ~3 cm away from the elbow, with one sensor package on the “thumb” side of the forearm and one on the “pinky” side. For conditions with the sensors above the elbow, we placed the armband ~7 cm above the elbow, such that one sensor package rested on the *biceps*. Right-handed participants had the armband placed on the left arm, which allowed them to use their dominant hand for finger input. For the one left-handed participant, we flipped the setup, which had no apparent effect on the operation of the system. Tightness of the armband was adjusted to be firm but comfortable. While performing tasks, participants could place their elbow on the desk, tucked against their body, or on the chair’s adjustable armrest; most chose the latter.

4.4. Procedure

For each condition, the experimenter walked through the input locations to be tested and demonstrated finger taps on each. Participants practiced duplicating these motions for approximately 1 min with each gesture set. This allowed participants to familiarize themselves with our naming conventions (e.g., “pinky,” “wrist”), and to practice tapping

their arm and hands with a finger on the opposite hand. It also allowed us to convey the appropriate tap force to participants, who often initially tapped unnecessarily hard.

To train the system, participants were instructed to comfortably tap each location 10 times, with a finger of their choosing. This constituted 1 training round. In total, 3 rounds of training data were collected per input location set (30 examples per location, 150 data points in total). An exception to this procedure was in the case of the 10 forearm locations, where only 2 rounds were collected to save time (20 examples per location, 200 data points in total). Total training time for each experimental condition was approximately 3 min.

We used the training data to build an SVM classifier. During the subsequent testing phase, we presented participants with simple text stimuli (e.g., “tap your wrist”), which instructed them where to tap. The order of stimuli was randomized, with each location appearing 10 times in total.

The system performed real-time segmentation and classification, and provided immediate feedback to the participant (e.g. “you tapped your wrist”). We provided feedback so that participants could see where the system was making errors (as they would if using a real application). If an input was not segmented (i.e., the tap was too quiet), participants could see this and would simply tap again. Overall, segmentation error rates were negligible in all conditions, and not included in further analysis.

5. RESULTS

In this section, we report on the classification accuracies for the test phases in the five different conditions. Overall, classification rates were high, with an average accuracy across conditions of 87.6%.

5.1. Five fingers

Despite multiple joint crossings and ~ 40 cm of separation between the input targets and sensors, classification accuracy remained high for the five-finger condition, averaging 87.7% (SD = 10.0%) across participants. Segmentation, as in other conditions, was essentially perfect.

5.2. Whole arm

Participants performed three conditions with the whole-arm location configuration. The below-elbow placement performed the best, posting a 95.5% (SD = 5.1%) average accuracy. This is not surprising, as this condition placed the sensors closer to the input targets than the other conditions. Moving the sensor above the elbow reduced accuracy to 88.3% (SD = 7.8%), a drop of 7.2%. This is almost certainly related to the acoustic loss at the elbow joint and the additional 10 cm between the sensor and input targets.

The eyes-free input condition yielded lower accuracies than other conditions, averaging 85.0% (SD = 9.4%). This represents a 10.5% drop from its vision-assisted (but otherwise identical) counterpart condition. It was apparent from watching participants complete this condition that targeting precision was reduced. In sighted conditions, participants appeared to be able to tap locations

with perhaps a 2 cm radius of error. Although not formally captured, this margin of error appeared to double or triple when the eyes were closed. We believe that additional training data, which better captures the increased input variability, would remove much of this deficit. However, we also caution designers developing eyes-free, on-body interfaces to carefully consider the locations participants can tap accurately.

5.3. Forearm

Classification accuracy for the 10-location forearm condition stood at 81.5% (SD = 10.5%), a surprisingly strong result for an input set we purposely devised to tax our system’s sensing accuracy.

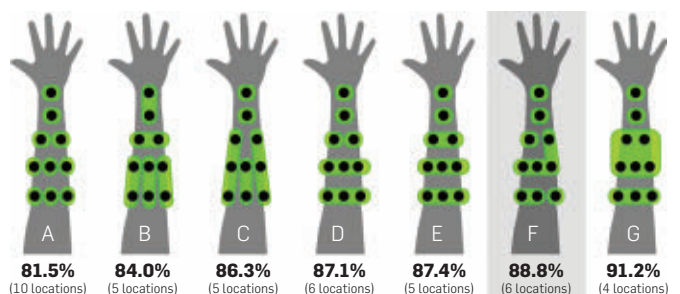
Using our experimental data, we considered different ways to improve accuracy by post hoc collapsing the 10 locations into input groupings. The goal of this exercise was to explore the tradeoff between classification accuracy and number of input locations on the forearm, which represents a particularly valuable input surface for application designers. We grouped targets into sets based on what we believed to be logical spatial groupings (Figure 7A–E and G). In addition to exploring classification accuracies for layouts that we considered to be intuitive, we also performed an exhaustive search (programmatically) over all possible groupings. For most location counts, this search confirmed that our intuitive groupings were optimal; however, this search revealed one plausible (although irregular) layout with high accuracy at six input locations (Figure 7F).

Unlike in the five-fingers condition, there appeared to be shared acoustic traits that led to a higher likelihood of confusion with adjacent targets than distant ones. This effect was more prominent laterally than longitudinally. Figure 7 illustrates this with lateral groupings consistently out-performing similarly arranged, longitudinal groupings (B and C vs. D and E). This is unsurprising given the morphology of the arm, with a high degree of bilateral symmetry along the long axis.

6. SUPPLEMENTAL EXPERIMENTS

We conducted a series of smaller, targeted experiments

Figure 7. Higher accuracies can be achieved by collapsing the 10 input locations into groups. A–E and G were designed to be spatially intuitive. F was created following analysis of per-location accuracy data.



to explore the feasibility of our approach for other applications. In the first additional experiment, which tested performance of the system while users walked and jogged, we recruited 1 male (age 23) and 1 female (age 26) for a single-purpose experiment. For the rest of the experiments, we recruited 7 new participants (3 female, mean age 26.9) from within our institution. In all cases, the sensor arm-band was placed just below the elbow. Similar to the previous experiment, each additional experiment consisted of a training phase, where participants provided between 10 and 20 examples for each input type, and a testing phase, in which participants were prompted to provide a particular input (10 times per input type). As before, input order was randomized; segmentation and classification were performed in real time.

6.1. Walking and jogging

With sensors coupled to the body, noise created during other motions is particularly troublesome, and walking and jogging represent perhaps the most common types of whole-body motion. This experiment explored the accuracy of our system in these scenarios.

Each participant trained and tested the system while walking and jogging on a treadmill. Three input locations were used to evaluate accuracy: arm, wrist, and palm. Additionally, the rate of false positives (i.e., the system believed there was input when in fact there was not) and true positives (i.e., the system was able to correctly segment an intended input) was captured. The testing phase took roughly 3 min to complete (four trials in total: two participants, two conditions). The male walked at 2.3 mph and jogged at 4.3 mph; the female at 1.9 and 3.1 mph, respectively.

In both walking trials, the system never produced a false-positive input. Meanwhile, true positive accuracy was 100%. Classification accuracy for the inputs (e.g., a wrist tap was recognized as a wrist tap) was 100% for the male and 86.7% for the female.

In the jogging trials, the system had four false-positive input events (two per participant) over 6 min of continuous jogging. True-positive accuracy, as with walking, was 100%. Considering that jogging is perhaps the hardest input filtering and segmentation test, we view this result as extremely positive. Classification accuracy, however, decreased to 83.3% and 60.0% for the male and female participants, respectively.

Although the noise generated from the jogging almost certainly degraded the signal (and in turn, lowered classification accuracy), we believe the chief cause for this decrease was the quality of the training data. Participants only provided 10 examples for each of 3 tested input locations. Furthermore, the training examples were collected while participants were jogging. Thus, the resulting training data was not only highly variable, but also sparse—neither of which is conducive to accurate machine learning classification. We believe that more rigorous collection of training data could yield even stronger results.

6.2. Single-handed gestures

In the experiments discussed thus far, we considered only

bimanual gestures, where the sensor-free arm, and in particular the fingers, are used to provide input. However, there are a range of gestures that can be performed with just the fingers of one hand. This was the focus of Amento et al.,¹ although this work did not evaluate classification accuracy.

We conducted three independent tests to explore one-handed gestures. The first had participants tap their index, middle, ring and pinky fingers against their thumb (akin to a pinching gesture) 10 times each. Our system was able to identify the four input types with an overall accuracy of 89.6% (SD = 5.1%). We ran an identical experiment using flicks instead of taps (i.e., using the thumb as a catch, then rapidly flicking the fingers forward). This yielded an impressive 96.8% (SD = 3.1%) accuracy in the testing phase.

This motivated us to run a third and independent experiment that combined taps and flicks into a single gesture set. Participants retrained the system, and completed an independent testing round. Even with eight input classes in very close spatial proximity, the system was able to achieve 87.3% (SD = 4.8%) accuracy. This result is comparable to the aforementioned 10-location forearm experiment (which achieved 81.5% accuracy), lending credence to the possibility of having 10 or more functions on the hand alone. Furthermore, proprioception of our fingers on a single hand is quite accurate, suggesting a mechanism for high-accuracy, eyes-free input.

6.3. Segmenting finger input

A pragmatic concern regarding the appropriation of fingertips for input was that other routine tasks would generate false positives. For example, typing on a keyboard strikes the finger tips in a very similar manner to the finger-tip input we proposed previously. Thus, we set out to explore whether finger-to-finger input sounded sufficiently distinct such that other actions could be disregarded.

As an initial assessment, we asked participants to tap their index finger 20 times with a finger on their other hand, and 20 times on the surface of a table in front of them. This data was used to train our classifier. This training phase was followed by a testing phase, which yielded a participant-wide average accuracy of 94.3% (SD = 4.5%, chance = 50%).

7. EXAMPLE INTERFACES AND INTERACTIONS

We conceived and built several prototype interfaces that demonstrate our ability to appropriate the human body, in this case the arm, and use it as an interactive surface.

While the bio-acoustic input modality is not strictly tethered to a particular output modality, we believe the sensor form factors we explored could be readily coupled with visual output provided by an integrated pico-projector. There are two nice properties of wearing such a projection device on the arm that permit us to sidestep many calibration issues. First, the arm is a relatively rigid structure—the projector, when attached appropriately, will naturally track with the arm. Second, since we have fine-grained control of the arm, making minute adjustments to align the projected image with the arm is trivial

(e.g., projected horizontal stripes for alignment with the wrist and elbow).

To illustrate the utility of coupling projection and finger input on the body (as researchers have proposed to do with projection and computer vision-based techniques¹⁶), we developed four proof-of-concept projected interfaces built on our system's live input classification. In the first interface, we project a series of buttons onto the forearm, on which a user can tap to navigate a hierarchical menu (Figure 1). In the second interface, we project a scrolling menu (Figure 2), which a user can navigate by tapping at the top or bottom to scroll up and down one item. Tapping on the selected item activates it. In a third interface, we project a numeric keypad on a user's palm and allow them to, for example, dial a phone number (Figure 1). Finally, as a true test of real-time control, we ported Tetris to the hand, with controls bound to different fingertips.

8. FUTURE WORK

In order to assess the real-world practicality of *Skinput*, we are currently building a successor to our prototype that will incorporate several additional sensors, particularly electrical sensors (allowing us to sense the muscle activity associated with finger movement, as per²¹) and inertial sensors (accelerometers and gyroscopes). In addition to expanding the gesture vocabulary beyond taps, we expect this sensor fusion to allow considerably more accuracy—and more robustness to false positives—than each sensor alone. This revision of our prototype will also allow us to benefit from anecdotal lessons learned since building our first prototype: in particular, early experiments with subsequent prototypes suggest that the hardware filtering we describe above (weighting our cantilevered sensors to create a mechanical band-pass filter) can be effectively replicated in software, allowing us to replace our relatively large piezoelectric sensors with micro-machined accelerometers. This considerably reduces the size and electrical complexity of our armband. Furthermore, anecdotal evidence has also suggested that vibration frequency ranges as high as several kilohertz may contribute to tap classification, further motivating the use of broadband accelerometers. Finally, our multi-sensor armband will be wireless, allowing us to explore a wide variety of usage scenarios, as well as our general assertion that always-available input will inspire radically new computing paradigms.

9. CONCLUSION

In this paper, we presented our approach to appropriating the human body as an interactive surface. We have described a novel, wearable, bio-acoustic sensing approach that can detect and localize finger taps on the forearm and hand. Results from our experiments have shown that our system performs well for a series of gestures, even when the body is in motion. We conclude with descriptions of several prototype applications that graze the tip of the rich design space we believe *Skinput* enables. 

References

1. Amento, B., Hill, W., Terveen, L. The sound of one hand: A wrist-mounted bio-acoustic fingertip gesture interface. In *Proceedings of the CHI '02 Extended Abstracts* (2002), 724–725.
2. Argyros, A.A., Lourakis, M.I.A. Vision-based interpretation of hand gestures for remote control of a computer mouse. In *Proceedings of the ECCV 2006 Workshop on Computer Vision in HCI, LNCS 3979* (2006), 40–51.
3. Burges, C.J. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* 2, 2 (June 1998), 121–167.
4. Deyle, T., Palinko, S., Poole, E.S., Starner, T. Hambone: A bio-acoustic gesture interface. In *Proceedings of the ISWC'07* (2007), 1–8.
5. Erol, A., Bebis, G., Nicolescu, M., Boyle, R.D., Twombly, X. Vision-based hand pose estimation: A review. *Comput. Vision Image Underst.* 108 (Oct. 2007), 52–73.
6. Grimes, D., Tan, D., Hudson, S.E., Shenoy, P., Rao, R. Feasibility and pragmatics of classifying working memory load with an electroencephalograph. In *Proceedings of the CHI'08* (2008), 835–844.
7. Harrison, C., Hudson, S.E. Scratch input: Creating large, inexpensive, unpowered and mobile finger input surfaces. In *Proceedings of the UIST'08* (2008), 205–208.
8. Ishii, H., Wisneski, C., Orbanes, J., Chun, B., Paradiso, J. PingPongPlus: Design of an athletic-tangible interface for computer-supported cooperative play. In *Proceedings of the CHI'99* (1999), 394–401.
9. Lakshminpathy, V., Schmandt, C., Marmasse, N. TalkBack: A conversational answering machine. In *Proceedings of the UIST'03* (2003), 41–50.
10. Lee, J.C., Tan, D.S. Using a low-cost electroencephalograph for task classification in HCI research. In *Proceedings of the CHI'06* (2006), 81–90.
11. Lyons, K., Skeels, C., Starner, T., Snoeck, C.M., Wong, B.A., Ashbrook, D. Augmenting conversations using dual-purpose speech. In *Proceedings of the UIST'04* (2004), 237–246.
12. Mandryk, R.L., Atkins, M.S. A fuzzy physiological approach for continuously modeling emotion during interaction with play environments. *Int. J. Hum. Comput. Stud.* 6, 4 (2007), 329–347.
13. Mandryk, R.L., Inkpen, K.M., Calvert, T.W. Using psychophysiological techniques to measure user experience with entertainment technologies. *Behav. Inf. Technol.* 25, 2 (Mar. 2006), 141–158.
14. Matheson, G.O., Maffey-Ward, L., Mooney, M., Ladly, K., Fung, T., Zhang, Y.T. Vibromyography as a quantitative measure of muscle force production. *Scand. J. Rehabil. Med.* 29, 1 (Mar. 1997), 29–35.
15. McFarland, D.J., Sarnacki, W.A., Wolpaw, J.R. Brain-computer interface (BCI) operation: Optimizing information transfer rates. *Biol. Psychol.* 63, 3 (Jul. 2003), 237–251.
16. Mistry, P., Maes, P., Chang, L. WUW—Wear ur world: A wearable gestural interface. In *Proceedings of the CHI '09 Extended Abstracts* (2009), 4111–4116.
17. Moore, M., Dua, U. A galvanic skin response interface for people with severe motor disabilities. In *Proceedings of the ACM SIGACCESS Accessibility and Computers '04* (2004), 48–54.
18. Paradiso, J.A., Leo, C.K., Checka, N., Hsiao, K. Passive acoustic knock tracking for interactive windows. In *Proceedings of the CHI '02 Extended Abstracts* (2002), 732–733.
19. Post, E.R., Orth, M. Smart fabric, or "wearable clothing." In *Proceedings of the ISWC'97* (1997), 167.
20. Rosenberg, R. The biofeedback pointer: EMG control of a two dimensional pointer. In *Proceedings of the ISWC'98* (1998), 4–7.
21. Saponas, T.S., Tan, D.S., Morris, D., Balakrishnan, R. Demonstrating the feasibility of using forearm electromyography for muscle-computer interfaces. In *Proceedings of the CHI'09* (2009), 515–524.
22. Starner, T. The role of speech input in wearable computing. *IEEE Perv. Comput.* 1, 3 (July 2002), 89–93.
23. Sturman, D.J., Zeltzer, D. A survey of glove-based input. *IEEE Comput. Graph. Appl.* 14, 1 (Jan. 1994).
24. Wilson, A. PlayAnywhere: A compact interactive tabletop projection-vision system. In *Proceedings of the UIST '05* (2005), 83–92.
25. Wilson, A.D. Robust computer vision-based detection of pinching for one and two-handed gesture input. In *Proceedings of the UIST'06* (2006), 255–258.
26. Witten, I.H., Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd edn. Morgan Kaufmann, San Francisco, CA, 2005.
27. Zhong, L., El-Daye, D., Kaufman, B., Tobaoda, N., Mohamed, T., Liebschner, M. OsteoConduct: Wireless body-area communication based on bone conduction. In *Proceedings of the ICST'07* (2007), 1–8.

Chris Harrison (chris.harrison@cs.cmu.edu), Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA.

Dan Morris (dan@microsoft.com), Microsoft Research, Redmond, WA.

Desney Tan (desney@microsoft.com), Microsoft Research, Redmond, WA.

Eastern New Mexico University Computer Science Instructor

Eastern New Mexico University is seeking qualified applicants for a full time tenure-track Instructor of Computer Science in the Department of Mathematical Sciences.

Requires a minimum Master's degree in computer science or equivalent course work.

Immediate review of applicants, open until filled. Apply at:

<http://agency.governmentjobs.com/enmu/>

KAIST Tenure-track Faculty

KAIST, the top Science & Engineering University in Korea, invites applications for tenure-track positions at all levels in Computer Science. We welcome applications from outstanding candidates in all major fields of computer science and in particularly its interdisciplinary areas. KAIST attracts nationwide top students pursuing B.S., M.S., and Ph.D. degrees.

KAIST offers a competitive start-up research fund and joint appointment with KAIST Institutes, which will expand opportunities in interdisciplinary research and funding. KAIST also provides housing for five years. KAIST is committed to increasing the num-

ber of female and non-Korean faculty members.

Applicants must have a Ph.D. or an equivalent degree in computer science or closely related field by the time of appointment. Strong candidates, who are expected to receive the Ph.D. degree within a year, can be offered our appointment. Applicants must demonstrate strong research potential and commitment to teaching undergraduate and graduate students. The teaching load is three hours per semester.

To apply, please submit the application form and CV along with the required documents to Head, Department of Computer Science (email: choi@cs.kaist.ac.kr). For more information on available positions, please visit our website: <http://cs.kaist.ac.kr/service/employment.cs>

Pacific Northwest National Laboratory Senior Software Architect

We're looking for a Senior Software Architect to lead the integration of multiple software technologies being developed in PNNL's Future Power Grid Initiative (FPGI). The successful candidate will define architectures and technology demonstrations that leverage FPGI components for clients, and lead proposals and collaborations with partners to exploit FPGI products.

BSc/MSc with 10 years experience OR PhD with 10 years experience.

Apply URL: <http://jobs.pnnl.gov/>

Riverbed Technology QA Software Engineer Network Integration-Applications

Like analyzing and breaking complex systems? Excited by virtualization and networking?

Ready to work on something big?

We've got the job for you.

Click our apply URL for more information and submit your resume.

<http://recruit.trovix.com/jobhostmaster/jobhost/ViewJobPostDetails.do?action=viewDetails&accountId=147a1477488137d390a50032705dd8f4b4dafcb9&jobId=2812&obSource=U+of+IL+-+Urbana-Champaign>

University of Cape Town Department of Computer Science Senior Lecturer or Associate Professor

University of Cape Town, the Department of Computer Science seeks to appoint a Senior Lecturer or Associate Professor to start in 2012.

The successful candidate should have a PhD, or expect to obtain their PhD by 2012. Candidates applying for a position at Associate Professor level should show evidence of high quality academic work in teaching, postgraduate supervision and research. Applicants for a Senior Lecturer post should provide clear information about their research and any teaching experience.

We seek a candidate with proven academic experience in Computer Science who will be expected to develop and teach courses, carry out research, contribute to departmental administration and supervise postgraduate students in their area of specialisation. Our Honours degrees are accredited by the British Computer Society and we have a large cohort of MSc and PhD students. The Department hosts the UCT Centre in ICT for Development, and also specialises in telecommunications, visual computing, high-performance computing, digital libraries, artificial intelligence and security. Candidates whose fields are compatible with these research groups are encouraged to apply.

The annual remuneration packages, including benefits for 2011 are as follows:

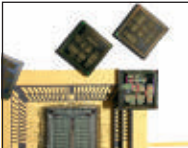
Senior Lecturer: R544 188

Associate Professor: R622 712

Please e-mail the completed application form at <http://web.uct.ac.za/depts/sapweb/forms/hr201.doc>, and all other relevant documentation as indicated on the form, to Ms Zulile Ncayiyana (Ref: SR137/11), Staff Recruitment and Selection, UCT, Cape Town by 22 August 2011.

E-mail: zulile.ncayiyana@uct.ac.za; Telephone: +27 21 650 3003; Department website: www.cs.uct.ac.za

UCT is committed to the pursuit of excellence, diversity and redress. Our Employment Equity Policy is available at <http://hr.uct.ac.za/policies/ee.php>.



Technical University of Denmark **DTU**

PROFESSOR

Software Engineering

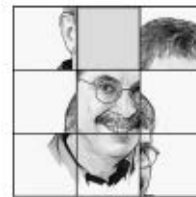
DTU Informatics. Applications are invited for a professorship in Software Engineering. Applicants should demonstrate the ability to combine theoretical knowledge with development of computational solutions within one of the following (or closely related) areas: Software Engineering for mobile and distributed computing (e.g., Internet-of-Things) and Software Engineering for high performance computing.

Application deadline: 15 September 2011

DTU is a leading technical university rooted in Denmark but international in scope and standard. Our total staff of 5,000 is dedicated to create value and to promote welfare for the benefit of society through science and technology; and our 7,000 students are being

trained to address the technological challenges of the future. While safeguarding academic freedom and scientific independence we collaborate with business, industry, government, public agencies as well as other universities around the world.

Further details: dtu.dk/career



DOI:10.1145/1978542.1978567

Peter Winkler

Puzzled

Uncommon Divisors

Welcome to three new puzzles. Solutions to the first two will be published next month; the third is (as yet) unsolved. In fact, this time it's a famously unsolved problem.

The theme is divisibility. A number n is said to divide a number m , written " $n|m$ ", if m is an integer multiple of n ; for example, " m is even" is the same as saying 2 divides m .

1. Does every positive integer divide some number whose representation (base 10) contains only zeroes and ones? For example, if we start computing multiples of 7, we find that $1,001 = 7 \times 143$ is among them. Then again, maybe 7 is just a lucky number.

2. Does every positive integer divide some Fibonacci number? Recall that the Fibonacci numbers begin 1, 1, after which each is the sum of the previous two; 1, 1, 2, 3, 5, 8, 13, 21, 34...; for example, 7 works (again), since $7 \times 3 = 21$, or the eighth Fibonacci number.

3. A perfect number m is a positive integer that is the sum of its proper divisors; that is, the sum of all n less than m , such that $n|m$. The first perfect number is 6; 1, 2, and 3 are its proper positive divisors, and $1 + 2 + 3 = 6$. The next perfect number is $28 = 1 + 2 + 4 + 7 + 14$, and the next six are

496;
8128;
33,550,336;
8,589,869,056;
137,348,691,328; and
2,305,843,008,139,952,128.

The first four perfect numbers were known in ancient Greece; the eighth goes back to Swiss mathematician Leonhard Euler in 1772.

Note that all these numbers—in fact all 47 known perfect numbers—are even. The even perfect numbers are in a sense well understood, each of the form $2^{p-1}(2^p-1)$, where both p and 2^p-1 are prime numbers (no divisor other than 1 and themselves).

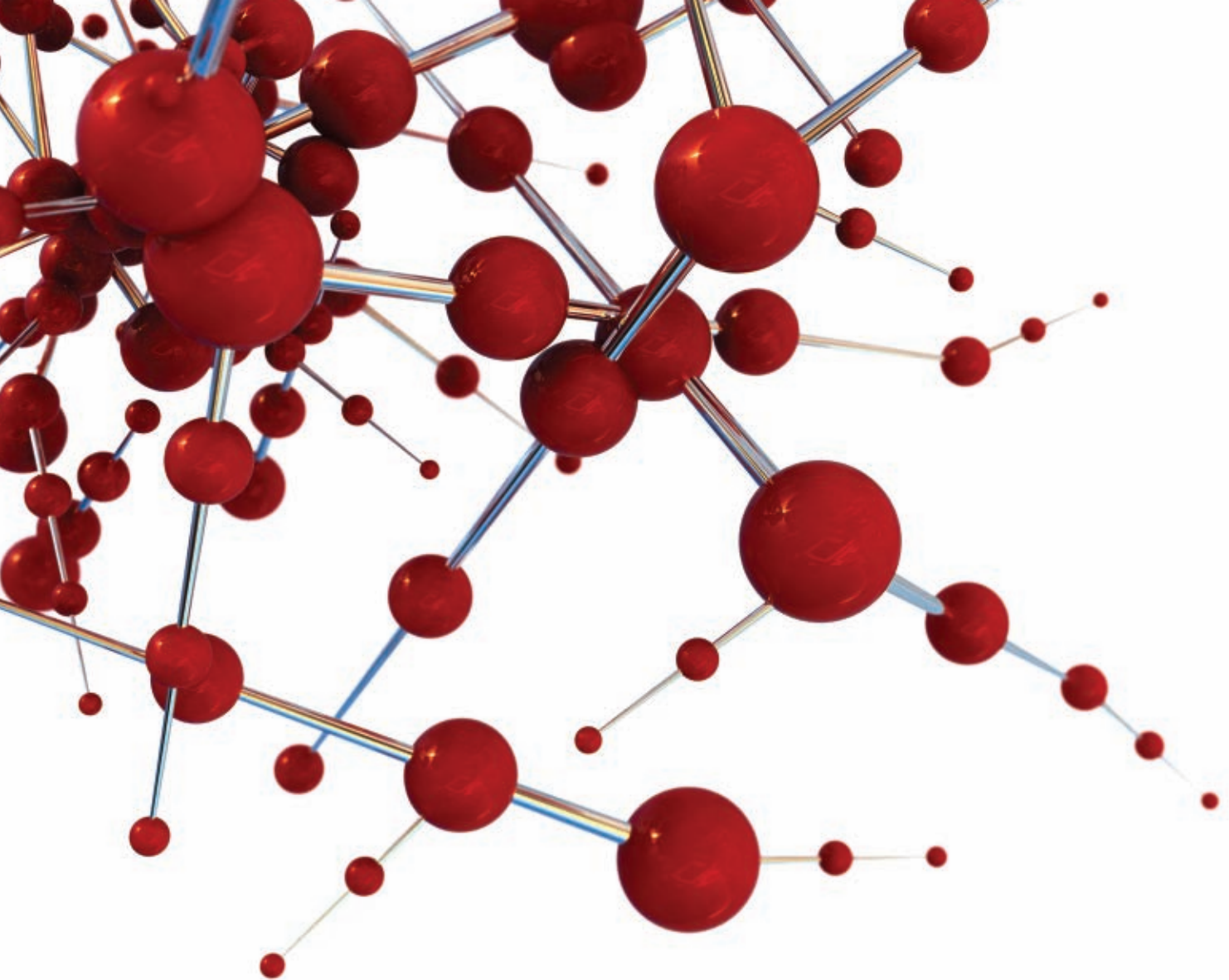
Unfortunately, it is difficult to tell for which primes p the number 2^p-1 is also prime; we don't even know whether there are infinitely many such primes p . Thus, we don't know if there are infinitely many perfect numbers, either.

Are there any *odd* perfect numbers? Major brainpower has been expended on this problem, over centuries, yet it remains tantalizingly open. Warning: If you are trying to find an odd perfect number by computer, please know that all odd numbers with 300 or fewer digits have already been checked.

Lots of information on perfect numbers is available from Wikipedia and other sources. My Dartmouth colleague Carl Pomerance, an expert in the area, can make a persuasive argument that odd perfect numbers probably don't exist. If you *prove* they don't exist, you will be famous!

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.



**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.

Fourth ACM SIGCHI Symposium on Engineering Interactive Computing Systems

ITU Copenhagen, Denmark

June 25–28, 2012

eics2012.org



EICS 2012 is the fourth international conference devoted to engineering usable and effective interactive computing systems. Topics of interest include multi-device interactive systems, new and emerging modalities (e.g., gesture), entertaining applications (e.g., mobile and ubiquitous games) and development methods (e.g., extreme programming).

EICS focuses on methods and techniques, and the tools that support designing and developing interactive systems. The Conference brings together people who study or practice the engineering of interactive systems, drawing from HCI, Software Engineering, Requirements Engineering, CSCW, Ubiquitous and Pervasive Systems, and Game Development communities.

The conference proceedings are published by the ACM.



Important dates

Long papers

December 19, 2011
(submission)

February 24 (notification)

April 2 (final submission)

Late-breaking results

March 9, 2012 (submission)

April 9 (notification)

April 18 (final submission)

Workshops

December 19, 2011
(proposal submission)

January 16 (notification)

March 5 (participant
submissions)

April 18 (organizers' final
submission)

Demonstrations, tutorials and the doctoral consortium

March 9, 2012 (submission)

April 9 (final decision)

April 18 (final submission)

Topics

- modeling interaction and interactive systems
- engineering processes for interactive systems (e.g. design, implementation, prototyping and testing)
- integrating interaction design into the software development process
- interactive systems specification
- requirements engineering for interactive systems
- software architectures for interactive systems
- frameworks and APIs for interactive systems (e.g. API usability, interaction-driven API design)
- dynamic generation and composition of interactive systems
- formal methods for HCI
- specifying users' activities
- innovative interactive applications
- designing usable software
- evaluation/testing of user interfaces
- end-user development of interactive systems
- engineering solutions for innovative applications

Organizing Committee

General	José C. Campos, Univ. Minho, Portugal Simone Barbosa, PUC Rio, Brazil
Organizing	Morten B. Harning, Dialogical, Denmark Jakob Bardram, ITU, Denmark
Papers	Philippe Palanque, Univ. Toulouse 3, France Rick Kazman, CMU/ Univ. Hawaii, USA
Late Breaking	Michael Harrison, Newcastle Univ., UK Steve Reeves, Univ. Waikato, New Zealand
Demonstrations	Gerrit Meixner, DFKI/ZMMI, Germany
Doctoral consort.	Laurence Nigay, Univ. Grenoble, France Alan Dix, Lancaster Univ., UK
Workshops	Kevin Schneider, Univ. Sask., Canada Panos Markopoulos, TU/e, The Netherlands
Tutorials	Gavin Doherty, TCD, Ireland Anke Dittmar, Univ. Rostok, Germany
Publicity	Greg Phillips, Royal Military College, Canada

