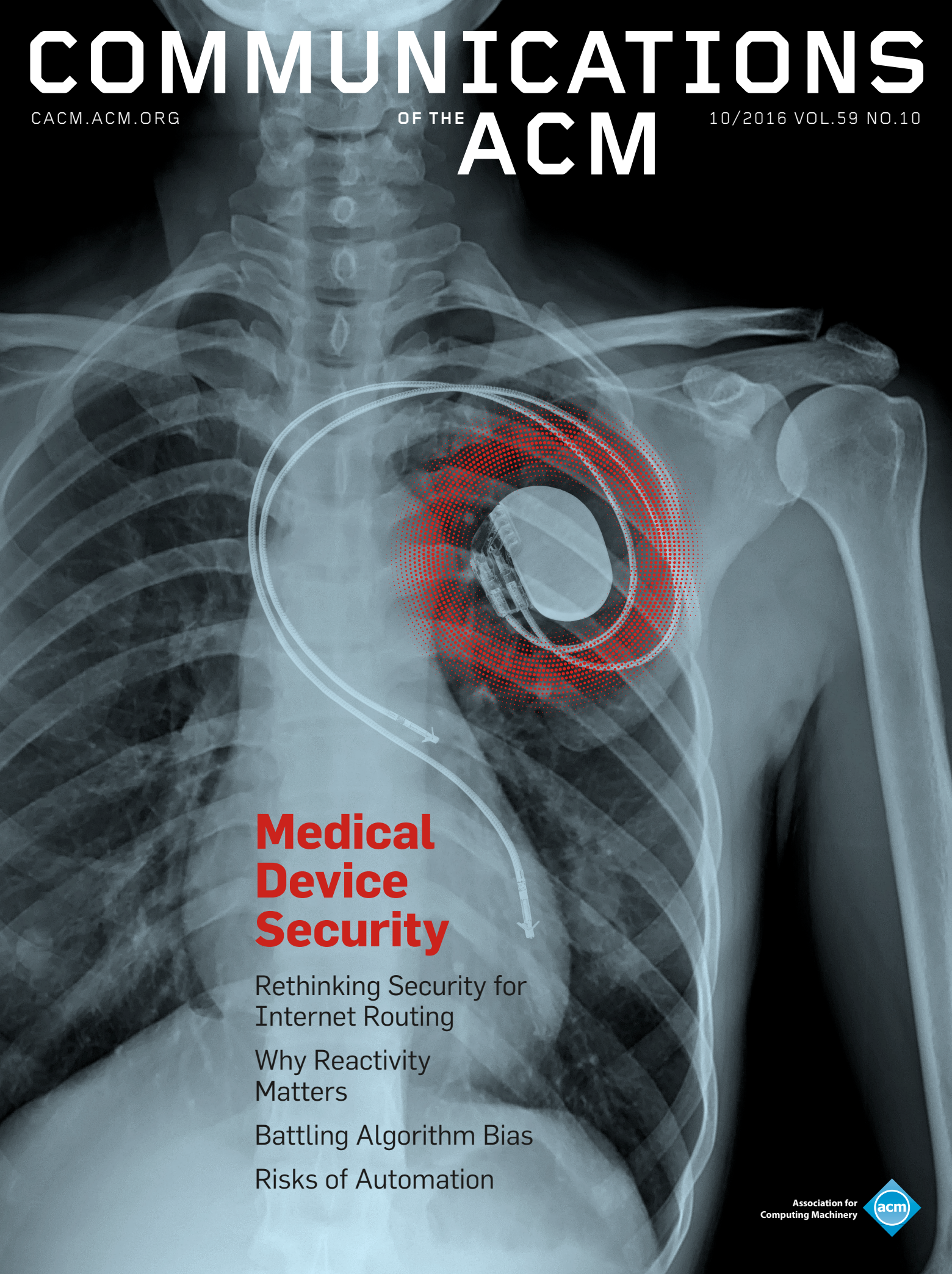


COMMUNICATIONS

CACM.ACM.ORG OF THE

ACM

10/2016 VOL.59 NO.10



Medical Device Security

Rethinking Security for
Internet Routing

Why Reactivity
Matters

Battling Algorithm Bias

Risks of Automation

Association for
Computing Machinery



HUMANS, MACHINES AND THE FUTURE OF WORK



The conference will focus on issues created by the impact of information technology on labor markets over the next 25 years, addressing questions such as:

- **What advances in artificial intelligence, robotics and automation are expected over the next 25 years?**
- **What will be the impact of these advances on job creation, job destruction and wages in the labor market?**
- **What skills are required for the job market of the future?**
- **Can education prepare workers for that job market?**
- **What educational changes are needed?**
- **What economic and social policies are required to integrate people who are left out of future labor markets?**
- **How can we preserve and increase social mobility in such an environment?**

RENOWNED SPEAKERS AND PANELISTS:

Diane Bailey

Associate Professor, School of Information, The University of Texas at Austin

Guruduth Banavar

Vice President, Cognitive Computing, IBM Research

John Seely Brown

Co-chairman, Deloitte's Center for the Edge; Adviser to the provost at USC

Daniel Castro

Vice President, Information Technology and Innovation Foundation

Stuart Elliott

Directorate for Education and Skills Organization for Economic Co-operation and Development (OECD)

Richard B. Freeman

Herbert Ascherman Chair in Economics, Harvard University

Eszter Hargittai

Delaney Family Professor, Communication Studies Department, Northwestern University

John Leslie King

W.W. Bishop Professor, School of Information, University of Michigan

Vijay Kumar

Nemirovsky Family Dean, School of Engineering and Applied Science, University of Pennsylvania

John Markoff

Senior Writer, The New York Times

Lawrence Mishel

President, Economic Policy Institute

Joel Mokyr

Robert H. Strotz Professor, Northwestern University; Sackler Professor by Special Appointment, Eitan Berglas School of Economics, Tel Aviv University

David Nordfors

Co-founder and Co-chair, Innovation for Jobs Summit

Debra Satz

Marta Sutton Weeks Professor of Ethics in Society, Professor of Philosophy, Senior Associate Dean for the Humanities and Arts, J. Frederick and Elisabeth Brewer Weintz University Fellow in Undergraduate Education, Stanford University

Manuela Veloso

Herbert A. Simon Professor, School of Computer Science, Carnegie Mellon University

Judy Wajcman

Anthony Giddens Professor of Sociology, The London School of Economics and Political Science

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

1966 A.J. Perlis
1967 Maurice Wilkes
1968 R.W. Hamming
1969 Marvin Minsky
1970 J.H. Wilkinson
1971 John McCarthy
1972 E.W. Dijkstra
1973 Charles Bachman
1974 Donald Knuth
1975 Allen Newell
1975 Herbert Simon
1976 Michael Rabin
1976 Dana Scott
1977 John Backus
1978 Robert Floyd
1979 Kenneth Iverson
1980 C.A.R Hoare
1981 Edgar Codd
1982 Stephen Cook
1983 Ken Thompson
1983 Dennis Ritchie
1984 Niklaus Wirth
1985 Richard Karp
1986 John Hopcroft
1986 Robert Tarjan
1987 John Cocke
1988 Ivan Sutherland
1989 William Kahan
1990 Fernando Corbató
1991 Robin Milner
1992 Butler Lampson
1993 Juris Hartmanis
1993 Richard Stearns
1994 Edward Feigenbaum
1994 Raj Reddy
1995 Manuel Blum
1996 Amir Pnueli
1997 Douglas Engelbart
1998 James Gray
1999 Frederick Brooks
2000 Andrew Yao
2001 Ole-Johan Dahl
2001 Kristen Nygaard
2002 Leonard Adleman
2002 Ronald Rivest
2002 Adi Shamir
2003 Alan Kay
2004 Vinton Cerf
2004 Robert Kahn
2005 Peter Naur
2006 Frances E. Allen
2007 Edmund Clarke
2007 E. Allen Emerson
2007 Joseph Sifakis
2008 Barbara Liskov
2009 Charles P. Thacker
2010 Leslie G. Valiant
2011 Judea Pearl
2012 Shafi Goldwasser
2012 Silvio Micali
2013 Leslie Lamport
2014 Michael Stonebraker
2015 Whitfield Diffie
2015 Martin Hellman

ACM A.M. TURING AWARD NOMINATIONS SOLICITED

Nominations are invited for the 2016 ACM A.M. Turing Award. This is ACM's oldest and most prestigious award and is given to recognize contributions of a technical nature which are of lasting and major technical importance to the computing field. The award is accompanied by a prize of \$1,000,000. Financial support for the award is provided by Google Inc.

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

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

**The deadline for nominations/endorsements is
November 30, 2016.**

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



Association for
Computing Machinery

Departments

- 5 **From the Publications Board**
Incentivizing Reproducibility
By Ronald F. Boisvert
-
- 7 **Cerf's Up**
'We're going backward!'
By Vinton G. Cerf
-
- 8 **BLOG@CACM**
Adding Art to STEM
Perry R. Cook considers the career path that led him to STEAM.
-
- 25 **Calendar**
-
- 93 **Careers**

Last Byte

- 96 **Upstart Puzzles**
Find Me Quickly
By Dennis Shasha

News



- 10 **Optical Fibers Getting Full**
Exploring ways to push more data through a fiber one-tenth the thickness of the average human hair.
By Don Monroe
-
- 13 **Bringing Holography to Light**
While 3D technologies that make headlines are not truly holographic, holographic techniques are furthering advances in important applications such as biomedical imaging.
By Marina Krakovsky
-
- 16 **Battling Algorithmic Bias**
How do we ensure algorithms treat us fairly?
By Keith Kirkpatrick

Viewpoints

- 18 **Technology Strategy and Management**
The Puzzle of Japanese Innovation and Entrepreneurship
Exploring how Japan's unique mixture of social, educational, and corporate practices influence entrepreneurial activity.
By Michael A. Cusumano
-
- 21 **Global Computing**
Mobile Computing and Political Transformation
Connecting increased mobile phone usage with political and market liberalization.
By Michael L. Best
-
- 24 **Code Vicious**
Cloud Calipers
Naming the next generation and remembering that the cloud is just other people's computers.
By George V. Neville-Neil
-
- 26 **Inside Risks**
Risks of Automation: A Cautionary Total-System Perspective of Our Cyberfuture
Where automation is inevitable, let's do it right.
By Peter G. Neumann
-
- 31 **Viewpoint**
Universities and Computer Science in the European Crisis of Refugees
Considering the role of universities in promoting tolerance as well as education.
By Kathrin Conrad, Nysret Musliu, Reinhard Pichler, and Hannes Werthner



Watch the authors discuss their work in this exclusive *Communications* video.
<http://cacm.acm.org/videos/universities-and-computer-science-in-the-european-crisis-of-refugees>

Practice



40

34 **Idle-Time Garbage-Collection Scheduling**

Taking advantage of idleness to reduce dropped frames and memory consumption.

By Ulan Degenbaev, Jochen Eisinger, Manfred Ernst, Ross McIlroy, and Hannes Payere

40 **Fresh Starts**

Just because you have been doing it the same way doesn't mean you are doing it the right way.

By Kate Matsudaira

42 **Dynamics of Change: Why Reactivity Matters**

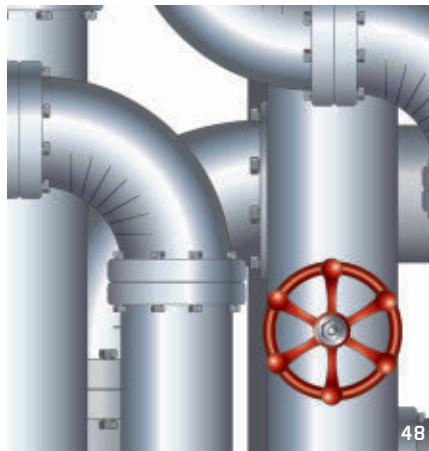
Tame the dynamics of change by centralizing each concern in its own module.

By Andre Medeiros



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

Contributed Articles



48

48 **Rethinking Security for Internet Routing**

Combine simple whitelisting technology, notably prefix filtering, in most BGP-speaking routers with weaker cryptographic protocols.

By Robert Lychev, Michael Schapira, and Sharon Goldberg

58 **Ethical Considerations in Network Measurement Papers**

The most important consideration is how the collection of measurements may affect a person's well-being.

By Craig Partridge and Mark Allman



About the Cover: Implanted medical devices have saved lives, extended lives, and enhanced the quality of life for patients worldwide. But, like most things software-driven, these devices pose privacy and security risks. This month's cover story traces medical device security issues and potential threats to the human body.

Review Articles

66 **A Brief Chronology of Medical Device Security**
With the implantation of software-driven devices comes unique privacy and security threats to the human body.

By A.J. Burns, M. Eric Johnson, and Peter Honeyman



Watch the authors discuss their work in this exclusive *Communications* video.
<http://cacm.acm.org/videos/a-brief-chronology-of-medical-device-security>

Research Highlights

74 **Technical Perspective**
Naiad

By Johannes Gehrke

75 **Incremental, Iterative Data Processing with Timely Dataflow**

By Derek G. Murray, Frank McSherry, Michael Isard, Rebecca Isaacs, Paul Barham, and Martin Abadi

84 **Technical Perspective**
The Power of Parallelizing Computations

By James Larus

85 **Efficient Parallelization Using Rank Convergence in Dynamic Programming Algorithms**

By Saeed Maleki, Madanlal Musuvathi, and Todd Mytkowicz



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

Bobby Schnabel

Deputy Executive Director and COO

Patricia Ryan

Director, Office of Information Systems

Wayne Graves

Director, Office of Financial Services

Darren Ramdin

Director, Office of SIG Services

Donna Cappel

Director, Office of Publications

Bernard Rous

Director, Office of Group Publishing

Scott E. Delman

ACM COUNCIL

President

Alexander L. Wolf

Vice-President

Vicki L. Hanson

Secretary/Treasurer

Erik Altman

Past President

Vinton G. Cerf

Chair, SGB Board

Patrick Madden

Co-Chairs, Publications Board

Jack Davidson and Joseph Konstan

Members-at-Large

Eric Allman; Ricardo Baeza-Yates;

Cherri Pancake; Radia Perlman;

Mary Lou Soffa; Eugene Spafford;

Per Stenstrom

SGB Council Representatives

Paul Beame; Jenna Neefe Matthews;

Barbara Boucher Owens

BOARD CHAIRS

Education Board

Mehran Sahami and Jane Chu Prey

Practitioners Board

George Neville-Neil

REGIONAL COUNCIL CHAIRS

ACM Europe Council

Dame Professor Wendy Hall

ACM India Council

Srinivas Padmanabhuni

ACM China Council

Jiaguang Sun

PUBLICATIONS BOARD

Co-Chairs

Jack Davidson; Joseph Konstan

Board Members

Ronald F. Boisvert; Karin K. Breitman;

Terry J. Coatta; Anne Condon; Nikil Dutt;

Roch Guerin; Carol Hutchins;

Yannis Ioannidis; Catherine McGeeoch;

M. Tamer Ozsu; Mary Lou Soffa; Alex Wade;

Keith Webster

ACM U.S. Public Policy Office

Renee Dopplick, Director

1828 L Street, N.W., Suite 800

Washington, DC 20036 USA

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

Computer Science Teachers Association

Mark R. Nelson, Executive Director

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

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

STAFF

DIRECTOR OF GROUP PUBLISHING

Scott E. Delman

cacm-publisher@cacm.acm.org

Executive Editor

Diane Crawford

Managing Editor

Thomas E. Lambert

Senior Editor

Andrew Rosenbloom

Senior Editor/News

Larry Fisher

Web Editor

David Roman

Rights and Permissions

Deborah Cotton

Art Director

Andrij Borys

Associate Art Director

Margaret Gray

Assistant Art Director

Mia Angelica Balaquiot

Designer

Iwona Usakiewicz

Production Manager

Lynn D'Addesio

Advertising Sales

Juliet Chance

Columnists

David Anderson; Phillip G. Armour;

Michael Cusumano; Peter J. Denning;

Mark Guzdial; Thomas Haigh;

Leah Hoffmann; Mari Sako;

Pamela Samuelson; Marshall Van Alstyne

CONTACT POINTS

Copyright permission

permissions@hq.acm.org

Calendar items

calendar@cacm.acm.org

Change of address

acmhhelp@acm.org

Letters to the Editor

letters@cacm.acm.org

WEBSITE

http://cacm.acm.org

AUTHOR GUIDELINES

http://cacm.acm.org/

ACM ADVERTISING DEPARTMENT

2 Penn Plaza, Suite 701, New York, NY

10121-0701

T (212) 626-0686

F (212) 869-0481

Advertising Sales

Juliet Chance

acmm mediasales@acm.org

For display, corporate/brand advertising:

Craig Pitcher

pitcherc@acm.org T (408) 778-0300

William Sleight

wsleight@acm.org T (408) 513-3408

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

eric@cacm.acm.org

NEWS

Co-Chairs

William Pulleyblank and Marc Snir

Board Members

Mei Kobayashi; Michael Mitzenmacher;

Rajeev Rastogi

VIEWPOINTS

Co-Chairs

Tim Finin; Susanne E. Hambrusch;

John Leslie King

Board Members

William Aspray; Stefan Bechtold;

Michael L. Best; Judith Bishop;

Stuart I. Feldman; Peter Freeman;

Mark Guzdial; Rachelle Hollander;

Richard Ladner; Carl Landwehr;

Carlos Jose Pereira de Lucena;

Beng Chin Ooi; Loren Terveen;

Marshall Van Alstyne; Jeannette Wing

Q PRACTICE

Co-Chair

Stephen Bourne

Board Members

Eric Allman; Peter Bailis; Terry Coatta;

Stuart Feldman; Benjamin Fried;

Pat Hanrahan; Tom Killalea; Tom Limoncelli;

Kate Matsudaira; Marshall Kirk McKusick;

George Neville-Neil; Theo Schlossnagle;

Jim Waldo

The Practice section of the CACM

Editorial Board also serves as

the Editorial Board of *Q PRACTICE*.

CONTRIBUTED ARTICLES

Co-Chairs

Andrew Chien and James Larus

Board Members

William Aiello; Robert Austin; Elisa Bertino;

Gilles Brassard; Kim Bruce; Alan Bundy;

Peter Buneman; Peter Druschel; Carlo Ghezzi;

Carl Gutwin; Yannis Ioannidis;

Gal A. Kaminka; James Larus; Igor Markov;

Gail C. Murphy; Bernhard Nebel;

Lionel M. Ni; Kenton O'Hara; Sriram Rajamani;

Mariel-Christine Rousset; Avi Rubin;

Krishan Sabnani; Ron Shamir; Yoav

Shoham; Larry Snyder; Michael Vitale;

Wolfgang Wahlster; Hannes Werthner;

Reinhard Wilhelm

RESEARCH HIGHLIGHTS

Co-Chairs

Azer Bestavros and Gregory Morrisett

Board Members

Martin Abadi; Amr El Abbadi; Sanjeev Arora;

Nina Balcan; Dan Boneh; Andrei Broder;

Doug Burger; Stuart K. Card; Jeff Chase;

Jon Crowcroft; Sandhya Dwaekadas;

Matt Dwyer; Alon Halevy; Norm Jouppi;

Andrew B. Kahng; Sven Koenig; Xavier Leroy;

Steve Marschner; Kobbi Nissim;

Steve Seitz; Guy Steele, Jr.; David Wagner;

Margaret H. Wright; Andreas Zeller

WEB

Chair

James Landay

Board Members

Marti Hearst; Jason I. Hong;

Jeff Johnson; Wendy E. MacKay

ACM Copyright Notice

Copyright © 2016 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@hq.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 \$269.

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 acmhhelp@acm.org.

COMMUNICATIONS OF THE ACM

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

POSTMASTER

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

Printed in the U.S.A.



Association for
Computing Machinery



DOI:10.1145/2994031

Ronald F. Boisvert

Incentivizing Reproducibility

A SCIENTIFIC RESULT is not truly established until it is independently confirmed. This is one of the tenets of experimental science. Yet, we have seen a rash of recent headlines about experimental results that could not be reproduced. In the biomedical field, efforts to reproduce results of academic research by drug companies have had less than a 50% success rate,^a resulting in billions of dollars in wasted effort.^b In most cases the cause is not intentional fraud, but rather sloppy research protocols and faulty statistical analysis. Nevertheless, this has led to both a loss in public confidence in the scientific enterprise and some serious soul searching within certain fields. Publishers have begun to take the lead in insisting on more careful reporting and review, as well as facilitating government open science initiatives mandating sharing of research data and code.

But what about experimental computer science? Fortunately, we haven't been in the headlines. But, it is rare for research studies in computing to be reproduced. On the surface this seems odd, since we have an advantage over science done in wet labs. For us, the object of study is often software, so it, along with the associated experimental scaffolding, is a collection of bits that can be easily shared for the purpose of audit and inspection, for an assisted attempt at replication, or for building upon the work to advance science further or to transfer technologies to commercial use. Certainly the situation is a bit more complex in practice, but there is no reason for us not to be leaders in practices that

enable audit and reuse when technically and legally possible.

Some communities within ACM have taken action. SIGMOD has been a true pioneer, establishing a reproducibility review of papers at the SIGMOD conference since 2008. The Artifact Evaluation for Software Conferences initiative has led to formal evaluations of artifacts (such as software and data) associated with papers in 11 major conferences since 2011, including OOPSLA, PLDI, and ISSTA. Here the extra evaluations are optional and are performed only after acceptance. In 2015 the *ACM Transactions on Mathematical Software* announced a Replicated Computational Results initiative,^c also optional, in which the main results of a paper are independently replicated by a third party (who works cooperatively with the author and uses author-supplied artifacts) before acceptance. The *ACM Transactions on Modeling and Computer Simulation* is also now doing this, and the *Journal of Experimental Algorithmics* has just announced a similar initiative. In all cases, successfully reviewed articles receive benefits, such as a brand on the paper and extra recognition at the conference.

To support efforts of this type, the ACM Publications Board recently approved a new policy on Result and Artifact Review and Badging.^d This policy defines two badges ACM will use to highlight papers that have undergone independent verification. *Results Replicated* is applied when the paper's main results have been replicated using artifacts provided by the author, or *Results Reproduced* if done completely independently.

Formal replication/reproduction is sometimes impractical. However,

both confidence in results and downstream reproduction are enhanced if a paper's artifacts (that is, code and datasets) have undergone a rigorous auditing process such as those being undertaken by ACM conferences. The new ACM policy provides two badges that can be applied here: *Artifacts Evaluated—Functional*, when the artifacts are found to be documented, consistent, complete, exercisable, and include appropriate evidence of verification and validation, and if, in addition the artifacts facilitate reuse and repurposing at a higher level, then *Artifacts Evaluated—Reusable* can be applied. When artifacts are made publicly available, further enhancing auditing and reuse, we apply an *Artifacts Available* badge. ACM is working to expose these badges in the ACM Digital Library on both the landing pages for articles and in search results.

Replication of results using author-supplied artifacts is no doubt a weak form of reproducibility, but it is an important first step. We believe that auditing that goes beyond traditional refereeing will help raise the bar for experimental research in computing, and that the incentives that we provide will encourage sharing and reuse of experimental artifacts.

This policy is but the first deliverable of the ACM Task Force on Data, Software and Reproducibility. Ongoing efforts are aimed at surfacing software and data as first-class objects in the DL, so it can serve as both a host and a catalog for not just articles, but the full range of research artifacts deserving preservation. ■

Ronald F. Boisvert (boisvert@acm.org) is chair of the ACM Publications Board's Digital Library Committee.

a *Nature Reviews Drug Discovery* 10, 643-644 (September 2011), doi:10.1038/nrd3545

b *Nature*, June 9, 2015, doi:10.1038/nature.2015.17711

c <http://toms.acm.org/replicated-computational-results.cfm>

d <http://www.acm.org/publications/policies/artifact-review-badging>

SHAPE THE FUTURE OF COMPUTING.

JOIN ACM TODAY.

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

SELECT ONE MEMBERSHIP OPTION

ACM PROFESSIONAL MEMBERSHIP:

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

ACM STUDENT MEMBERSHIP:

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

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

Priority Code: CAPP

Payment Information

Name _____

ACM Member # _____

Mailing Address _____

City/State/Province _____

ZIP/Postal Code/Country _____

Email _____

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

☐ AMEX ☐ VISA/MasterCard ☐ Check/money order

Total Amount Due _____

Credit Card # _____

Exp. Date _____

Signature _____

Purposes of ACM

ACM is dedicated to:

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

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

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

Satisfaction Guaranteed!

BE CREATIVE. STAY CONNECTED. KEEP INVENTING.



Association for
Computing Machinery

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

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

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



DOI:10.1145/2993746

Vinton G. Cerf

'We're going backward!'

In caves in Lascaux, France, magnificent artworks were discovered from 17,300 years ago. Cuneiform clay tablets written over 5,000 years ago are still readable today (if you happen

to know Akkadian, Eblaite, Elamite, Hattic, Hittite, Hurrian, Luwian, Sumerian, Urartian, or Old Persian). Egyptian hieroglyphic writing was more or less contemporary with cuneiform and papyrus manuscripts dating from about 4,600 years ago have survived. The Greeks and the Romans carved letters in stone and these are still eminently readable over 2,000 years later.

Vellum and parchment manuscripts dating to 4,400 years ago still exist, albeit in fragmentary form. On the other hand, illuminated manuscripts on parchment or vellum dating from 1000 A.D. are still magnificent in appearance and eminently readable if one is familiar with the Latin or Greek of the period and the stylized penmanship of the age.

In art galleries and museums, we enjoy paintings dating from the 15th century and frescoes from even earlier times. We find Chinese block printing on paper from the 8th century, 1,200 years ago. The rag paper used before the 19th century leaves us with books that are still well preserved. We even have photographs on glass plates or on tin that date to the 1800s.

Perhaps by now you are noticing a trend in the narrative. As we move toward the present, the media of our expression seems to have decreasing longevity. Of course, newer media have not been around as long as the older ones so their longevity has not been demonstrated but I think it is arguable that the more recent media do not have the resilience of stone or baked clay. Modern photographs may not last more than 150–200 years before they fade or

disintegrate. Modern books, unless archival paper is used, may not last more than 100 years.

I have written more than once in this column about my concerns for the longevity of digital media and our ability to correctly interpret digital content, absent the software that produced it. I won't repeat these arguments here, but a recent experience produced a kind of cognitive dissonance for me on this topic. I had gone to my library of science fiction paperbacks and pulled out a copy of Robert Heinlein's *Double Star* that I had purchased about 50 years ago for 35 cents. I tried to read it, but out of fear for breaking the binding, and noting the font was pretty small, I turned to the Kindle library and downloaded a copy for \$6.99, or something like that, and read the book on my laptop with a font size that didn't require glasses! So,

It seems inescapable that our society will need to find its own formula for underwriting the cost of preserving knowledge in media that will have some permanence.

despite having carefully kept the original paperback, I found myself resorting to an online copy for convenience and feeling lucky it was obtainable.

This experience set me to thinking again about the ephemeral nature of our artifacts and the possibility that the centuries well before ours will be better known than ours will be unless we are persistent about preserving digital content. The earlier media seem to have a kind of timeless longevity while modern media from the 1800s forward seem to have shrinking lifetimes. Just as the monks and Muslims of the Middle Ages preserved content by copying into new media, won't we need to do the same for our modern content?

These thoughts immediately raise the question of financial support for such work. In the past, there were patrons and the religious orders of the Catholic Church as well as the centers of Islamic science and learning that underwrote the cost of such preservation. It seems inescapable that our society will need to find its own formula for underwriting the cost of preserving knowledge in media that will have some permanence. That many of the digital objects to be preserved will require executable software for their rendering is also inescapable. Unless we face this challenge in a direct way, the truly impressive knowledge we have collectively produced in the past 100 years or so may simply evaporate with time. 

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

Copyright held by author.

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:10.1145/2967972

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

Adding Art to STEM

Perry R. Cook considers the career path that led him to STEAM.



Perry R. Cook
Combining Arts and Technology, Creating STEAM Curricula
<http://bit.ly/1rC47EO>
April 28, 2016

I grew up an artsy+nerdy kid, singing in choir, playing in band, as comfortable with a soldering iron, with fixing or hacking an old radio or electronic organ, as with chord progressions or improvising harmonies on the fly. In high school, I sang in every choir, played in every band, and did theater and speech. I also kept a keen and interested eye toward technology, especially music technology.

My original goal in going to conservatory in 1973 was to become a band/choir teacher, double-majoring in trombone and voice, with education and techniques courses for choir and band certification. But something fateful happened; I discovered my music school had an electronic music and recording studio. Also around that time, and at the urging of my trombone teacher, I became a voice major. What really happened is I became a de facto major in a recording and electronic music program that my music school did not have (yet). I spent every available minute in those studios, also doing location recordings, editing tapes, soldering patch cords, and reading

every book and journal I could find on audio, acoustics, recording, and electronic music.

I loved the studio work so much that in 1976, I ended up dropping out to become a sound engineer for about five years. I did lots of outdoor and indoor sound reinforcement gigs, some system designs, lots of building, installing, repair, and some studio work as well, both as an engineer and singer. All the while I was working feverishly in my own home studio, collecting (or building) a good variety of synthesizers, recording gear, and effects devices. I made lots of music, but the more I worked as a sound engineer, the more I realized there was math and science I needed to know to be as creative, and valuable, as possible.

So I went back to school in 1981, this time in electrical engineering (EE), but also finished my music degree in the process. Pretty much every course I took in my EE program, I was asking myself how it applied to sound, acoustics, and music. I finished with honors, and even though I was now dual-degreed, I knew there was much more that I did not know. I applied to graduate schools, and got into Stanford University, and found myself at the holy city (for nerds like me): the Center for Computer Research in Music and Acoustics, also called CCRMA.

There, I got to work with brilliant people like John Chowning (the inventor of FM sound synthesis, and pioneer of spatial sound and compositional uses of sound synthesis), DSP guru Julius O. Smith, Chris Chafe, Dexter Morrill, Max Mathews (the father of computer music), John Pierce (former Bell Labs Executive Director of Communications Sciences), and many others. I worked on physical modeling, new performance interfaces, created countless new software programs for all sorts of things, and researched and developed physics-based voice models for singing synthesis, which was the topic of my Ph.D. thesis.

CCRMA taught me so much about so many topics, but possibly the most important thing was that art, science, math, and engineering can (and should) be linked together. I observed students that study in this way learn differently, better, and create amazing and novel things just as part of their coursework. Pretty much all of the curricular elements of CCRMA are STEAM (science, technology, engineering, arts, math) in nature; math, music, physics, psychoacoustics, engineering(s), and other technical/design/art areas are woven together tightly and constantly.

When I moved to Princeton University in 1996, I got to take over a course Ken Steiglitz (EE/CS) and Paul Lansky (Music) had created, called "Transforming Reality Through Computer." It was really an applied DSP course, but with musical examples and projects. For quite a while I had been

teaching a CCRMA short course every summer with Xavier Serra called “Introduction to Spectral (Xavier) and Physical (Perry) Modeling.” My 10 lectures had turned into a fairly formal introduction, a set of notes, and eventually book chapters, to which I added a couple of spectrum analysis chapters, and a couple more on applications, and it became the book *Real Sound Synthesis for Interactive Applications*. That book and course was my first “scratch-built” STEAM curriculum, cross-listed in CS, EE, and Music at Princeton. The focal topic of the book is sound effects synthesis for games, VR, movies, etc. That topic also earned me a National Science Foundation (NSF) CAREER grant.

At Princeton, I also introduced a course called “Human Computer Interface Technology,” developed jointly with Ben Knapp and Dick Duda at San Jose State University (they got an NSF grant for this), Chris Chafe and Bill Verplank at CCRMA, and other faculty at the University of California, Davis, and the Naval Postgraduate School in Monterey. The emphasis at Stanford and Princeton was on creating NIMes (New Interfaces for Musical Expression), putting sensors on anything and everything to make new expressive sound and music controllers. Another STEAM course was born.

I continued to weave musical and artistic examples into all of my teaching and student advising. The next major new STEAM curriculum creation was the Princeton Laptop Orchestra (PLOrk), founded in 2005 by Dan Trueman (a former grad student who then joined the music faculty at Princeton) and myself. This course combined art, programming, live performance (some of it live coding in front of an audience!), engineering, listening, recording and studio techniques, and much more. Dan and I begged and cajoled around the Princeton campus to get money to get it off the ground, getting funds from Music, CS, the Dean of Engineering, the Freshman Seminar Fund, the Sophomore Experience Fund, and other sources to put together an ensemble of 15 “instruments” consisting of a laptop, a six-channel hemispherical speaker, amps, and controllers. Result? BIG success. As just one exam-

ple of hundreds, here is a quote from a PLOrk member, a female undergraduate music major, a cellist who had never programmed before:

“However, when everything worked the way it was supposed to, when my spontaneous arrangement of computer lingo transformed into a musical composition, it was a truly amazing experience. The ability to control duration and pitch with loops, integers, and frequency notation sent me on a serious power trip ... This is so much better than memorizing French verbs.”

Within a year or so, we had applied for and won a \$250,000 MacArthur Digital Learning Initiative grant, allowing PLOrk to build custom six-channel speakers with integrated amps; buy more laptops, controllers, and hardware, and grow to 45 total seats in the orchestra. We also toured, played Carnegie Hall, hosted and worked with world-famous guest artists, and inspired a horde of new laptop orchestras (LORks) around the world. Dan also worked on modifying the Princeton undergrad music curriculum to incorporate PLOrk courses, and I worked to see some of the PLOrk course sequence would count for Princeton CS and Engineering credit.

For his Ph.D. thesis in Computer Science at Princeton, Ge Wang created a new programming language called Chuck. It was designed from the ground up to be real-time, music/audio-centric, and super-friendly to inputs from external devices ranging from trackpads and tilt sensors to joysticks and music keyboards. Chuck was the native teaching language of PLOrk, and then SLOrk (the Stanford Laptop Orchestra, formed by Wang when he became a CCRMA faculty member), and many other LORks. It also was and is used for teaching beginning programming in a number of art schools and other contexts.

A few years ago, Ajay Kapur and I won an NSF grant for “A Computer Science Curriculum for Arts Majors” at the California Institute for the Arts. We created and crafted the curriculum, and taught it with careful assessments to make sure the art students were really learning the CS concepts. We iterated on the course, and it became a book (by Ajay, me, Spencer Salazar, and Ge). The course also be-

came a massive open online course (MOOC) whose first offering garnered over 40,000 enrolled students.

Now to Kadenze, which is a company Ajay, myself, and others co-founded and launched a year ago. Kadenze’s focus is to bring arts and creative technology education to the world, by assembling the best teachers, topics, and schools online. My *Real Sound Synthesis* topic is a Kadenze course offered by Stanford. The CalArts ChuckK course is there, as are courses on music tools, other programming languages, and even machine learning, all created for artists and nerds who want to use technology to be creative.

The genesis of Kadenze is absolutely STEAM. Artists need to know technical concepts. They need to program, build, solder, design, test, and use technology in their art-making. Engineers and scientists can benefit greatly from knowing more about art and design. Cross-fertilizing the two is good, but it is my feeling having both in one body is the best of all. Not all students need to get multiple degrees like I did, one in music, one (or more) in EE, but all of the names I mentioned in this short “STEAM teaching autobiography” actually are considered both artists and scientists by all those around them. They do concerts and/or create multimedia art works. They research and publish papers. They create both technology-based works of art and artistic works of code, design, and technology. The “Renaissance Person” can and should be. We need many more.

Specialization is necessary to garner expertise, but striving and working to become a skilled multidisciplinary generalist creates a whole person that can create, cope, build, refine, test, and use in practice. Plus, they can explain difficult concepts to novices, and carry the magic of combining art and technology to others. In other words, they are good teachers, too.

That has been my goal in life, and I think I am succeeding (so far). 

ACM Fellow **Perry R. Cook** is Professor (Emeritus) of Computer Science, with a joint appointment in Music, at Princeton University. He also serves as Research Coordinator and IP Strategist for SMule, and is co-founder and executive vice president of Kadenze, an online arts/technology education startup.

© 2016 ACM 0001-0782/16/10 \$15.00

Optical Fibers Getting Full

Exploring ways to push more data through a fiber one-tenth the thickness of the average human hair.

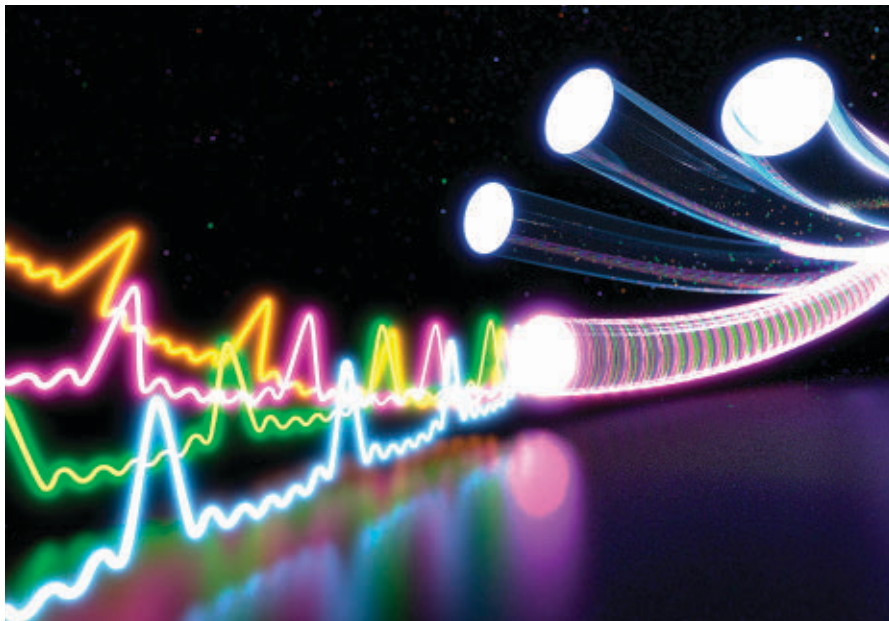
SINCE OPTICAL FIBERS were first deployed for communications in the 1970s, the number of bits per second a single fiber can carry has grown by the astonishing factor of 10 million, permitting an enormous increase in total data traffic, including cellular phone calls that spend most of their lives as bits traveling in fiber.

The exponential growth resembles Moore's Law for integrated circuits. Technology journalist Jeff Hecht has proposed calling the fiber version "Keck's Law" after Corning researcher Donald Keck, whose improvements in glass transparency in the early 1970s helped launch the revolution. The simplicity of these "laws," however, obscures the repeated waves of innovation that sustain them, and both laws seem to be approaching fundamental limits.

Fiber researchers have some cards to play, though. Moreover, if necessary the industry can install more fibers, similar to the way multiple processors took the pressure off saturating clock rates.

However, the new solutions may not yield the same energy and cost savings that have helped finance the telecommunication explosion.

Optical fiber became practical when



researchers learned how to purify materials and fabricate fibers with extraordinary transparency, by embedding a higher refractive-index core to trap the light deep within a much larger cladding. Subsequent improvements reduced losses to their current levels, about 0.2 dB/km for light wavelengths (infrared "colors") near 1.55 μm . A laser beam that is turned on and off to encode bits can transmit voice or data

tens of kilometers before it must be detected and retransmitted. In ensuing years the bit rate increased steadily, driven both by faster transmitters and receivers and by fiber designs that minimized the spread of the pulses.

As the pace of improvements began to slow, researchers realized they could send more information through fiber by combining light of slightly different wavelengths, each carrying its own

stream of data. The beams are multiplexed into a single fiber and demultiplexed at the other end using high-tech devices akin to prisms that separate white light into colors.

Adoption of this wavelength-division multiplexing, or WDM, was greatly aided by erbium-doped fiber amplifiers. These devices splice in a moderate length of specialty fiber containing a trace of the rare-earth element, which is pumped with a nearby laser to amplify any passing light within a range of wavelengths. Crucially, this amplification occurs with no need to convert the light to an electrical signal and back again, or even to separate the different colors. Signals can thus travel thousands of kilometers in the form of light.

The widespread adoption of WDM in the 1990s transformed the conception of optical communication from a single modulated beam to a complete spectrum like that familiar for radio waves. The seemingly narrow “C-band” of erbium used in most amplifiers corresponds to a bandwidth of roughly 10 THz, theoretically enough to carry as much as 20 trillion bits (Tb) per second of on/off data. Systems offering scores of wavelength channels were built to take advantage of this previously unheard-of capacity.

Unfortunately, the rapid fiber installation boom was motivated by extraordinary demand projections that proved unrealistic, resulting in a period of excess fiber capacity. Nonetheless, overall traffic has continued to double every two years or less, so after a few years increased capacity was once again needed in high-traffic parts of the network.

To provide this capacity, companies adopted a long-standing research vision of “coherent communication” into the marketplace in about 2010. Rather than representing bits as the presence or absence of light, this technique, widely used in the radio spectrum, encodes data in the phase and the amplitude of the light wave. Although the number of symbols per second is still limited by the available bandwidth, coherent communication allows each symbol to represent multiple bits of information, so the total bit rate increases. Typical systems now transmit 100 Gb/s on each wavelength, or 8 Tb/s over 80 WDM channels, in a single fiber.

A critical—and still open—question is whether systems can become cheaper with SDM than with multiple separate fibers.

Nonlinear Shannon Limit

In theory (the information theory attributed to Claude Shannon at Bell Laboratories in 1948), the number of bits that can be packed into a symbol is limited by the base-2 logarithm of the signal-to-noise ratio. Increasing the power can increase the bit rate, but only gradually.

For optical fibers, however, increased optical power changes the dielectric constant, and thus the propagation, of the optical signal. “There are extra distortions, and some of them you cannot compensate,” said René-Jean Essiambre of Bell Laboratories in Crawford Hill, NJ, which was recently acquired by Nokia (and named Nokia Bell Labs). “These distortions act like noise,” and ultimately nullify any advantage of increased power.

Interestingly, because the nonlinear effects caused by the data on one wavelength channel affect all other channels in the fiber, the net result is a limit on the total number of bits per second in all channels combined. Essiambre and his colleagues have calculated this limit for specific network configurations, and have concluded modern coherent systems are quite close to it.

The limitations on bit rate become especially stringent for very long distances. In addition, realistic reductions in fiber nonlinearity cause only a modest improvement in capacity, Essiambre said. “To increase that number is very difficult because it’s a logarithm.”

To reduce the nonlinearity of conventional fibers, researchers have tried making the core out of pure silica or spreading the light over a larger cross-sectional area, said David Richardson, deputy director of the Optoelectronics

Research Centre at University of Southampton in the U.K. “Significant progress has been made,” Richardson said, but “you’re not going to get a factor of 10” reduction in nonlinearity.

In contrast, a 1,000-fold reduction in the nonlinearity has been demonstrated using a fiber that confines the light to an empty core within a periodic “photonic bandgap” material for the cladding. Unfortunately, because of the logarithm and other effects, “the benefits don’t scale linearly,” Richardson said, so “you maybe get a factor of three” improvement in performance. Moreover, the fibers have so far shown an order of magnitude greater loss than conventional fibers, so photonic bandgap fibers are in “the dim and distant future.”

Space-Division Multiplexing

An approach that is “perhaps a little less radical,” space-division multiplexing (SDM), could involve either multiple cores within a single cladding or a fiber that supports several spatial modes rather than just one. Multicore fibers, for example, are “not particularly controversial,” Richardson said, adding that most people accept that the fibers can be operated independently. Even if spatial modes get mixed during their travel, the digital signal processing used in coherent systems can disentangle them as it does for polarization modes and in current application to multiple-antenna radio systems.

A critical—and still open—question is whether systems can become cheaper with SDM than with multiple separate fibers. Researchers have demonstrated simultaneous amplification of different spatial modes by incorporating optical gain into the cladding they all share. “This is where the technology may provide an advantage,” Richardson said, as erbium amplifiers did for WDM.

One company already championing integrated components is Infinera Corp., but Geoff Bennett, the company’s director of Solutions and Technology, is skeptical about SDM. “I’m not going to say never, but for the foreseeable time horizon it’s just not going to happen.”

A major problem is that SDM requires different fibers. “Deploying new fiber is literally the last resort that any operator would consider,” Bennett said, noting recent submarine cable installations have used large-area fibers be-

cause their lower nonlinearity is particularly advantageous on those long links.

SDM systems would also require different connectors, splicing, and other infrastructure. “None of that ecosystem that’s been developed over the last 20 years will work” for SDM, Bennett said. Although some links are heavily oversubscribed, “in general there’s plenty of unlit fiber out there” from the boom of the early 2000s. Lighting up a spare fiber from a cable containing scores of them will require a chain of amplifiers every 80 km or so, he admits, but “they’re not that expensive and they never break.”

Lower-Hanging Fruit

Coherent technology has expanded the raw capacity of existing fiber, Bennett said, but there are still opportunities to improve the operational and cost dimensions of network performance. Digital processing was first introduced at receivers, allowing for greater capacity as well as compensation for signal distortions. In what Bennett calls the “second coherent era,” processing is being incorporated at transmitters as well. “That gives you a number of options.”

One such option is the construction of “superchannels,” multiple wavelengths that can be squeezed closer in frequency without interference by shaping the pulses. Tapping the frequency space between neighboring channels “allows you to unlock a lot more capacity in the fiber,” Bennett said; in a typical case, growing from about 8 Tb/s to about 12 Tb/s.

Sean Long, director for Product Management at Huawei, also regards SDM as a question mark for the future, although his company has a “small group” looking at it. “Theoretically, that’s the direction we need to go,” but “there’s a lot of things that we need to develop,” he said. “It’s still too complicated.”

Also, “We still have things we can do before that,” Long said, potentially including erbium amplifiers in the unused spectral region known as L band. “Currently we are more focusing on the spectral efficiency” by exploiting transmission-side digital signal processing. “The flexibility is there already. Now we need to figure out how we can make the best combination for certain applications.”

Energy Crisis

However industry addresses bit-rate limits, other challenges are coming, which were the subject of a May 2015 meeting on “Communications networks beyond the capacity crunch.” Co-organizer Andrew Ellis of Aston University in Birmingham, U.K., had previously analyzed the implications of the nonlinear Shannon limit. Unfortunately, “there are equal problems across the rest of the network,” such as software protocols, he said.

If fiber nonlinearities require the use of duplicate fibers and other components, “it’s difficult to see how you’re going to sustain” the historical reduction in energy cost per bit that has driven network expansion, Ellis said. “Every time we’ve introduced a new generation,

there’s been a factor-of-four improvement in performance and the energy cost has only gone up by a factor of two.”

Even if energy reduction continues, the total energy use by communications networks is projected to rival all other energy use within two or three decades, Ellis said. “We are going to use a greater and greater amount of energy if the demand keeps growing.” **G**

Further Reading

Hecht, J.

Great Leaps of Light, *IEEE Spectrum*, February 2016, p. 28.

Ellis, A.D., Suibhne, N. M., D. Saad, D., and Payne, D.N.

Communication networks beyond the capacity crunch, *Philosophical Transactions of The Royal Society A* 2016 374 20150191; DOI: 10.1098/rsta.2015.0191. Published 25 January 2016, <http://rsta.royalsocietypublishing.org/content/374/2062/20150191>

Richardson, D.J.

New optical fibres for high-capacity optical communications, *Philosophical Transactions of The Royal Society A*, 2016 374 20140441; DOI: 10.1098/rsta.2014.0441. Published 25 January 2016, <http://rsta.royalsocietypublishing.org/content/374/2062/20140441>

Andrew Ellis

Boosting Bandwidth, *Physics World*, April 2016, p. 17, http://www.unloc.net/images/news/AndrewEllis_PhysicsWorld_finalarticle.pdf

Don Monroe is a science and technology writer based in Boston, MA.

© 2016 ACM 0001-0782/16/10 \$15.00

Milestones

Matsudaira Receives NCWIT Symons Innovator Award

The National Center for Women & Information Technology (NCWIT) recently named Kate Matsudaira 2016 recipient of its Symons Innovator Award, which promotes women’s participation in information technology (IT) and entrepreneurship by honoring an outstanding woman who has successfully built and founded an IT business.

A software engineer who has led work on distributed systems,

cloud computing, and mobile development, Matsudaira worked in a number of companies and startups before starting her own firm, Popforms, to create content and tools to help employees and managers be more productive. Safari Books Online, owned by O’Reilly Media, purchased Popforms in 2015.

Matsudaira currently is a principal of Urban Influence, a Seattle-based brand and

interactive development firm. She is a published author, keynote speaker, a member of the editorial board of *ACM Queue*, and maintains a personal blog at katemats.com.

NCWIT said Matsudaira has exhibited leadership through managing entire product teams and research scientists, and by building her own profitable business.

The award is named for

Jeanette Symons, founder of Industrious Kid, Zhone Technologies, and Ascend Communications, and an NCWIT Entrepreneurial Hero whose pioneering work made her an inspiration to many.

NCWIT hopes the Symons Award inspires other women to pursue IT entrepreneurship, and increases awareness of the importance of women’s participation in IT.

Bringing Holography to Light

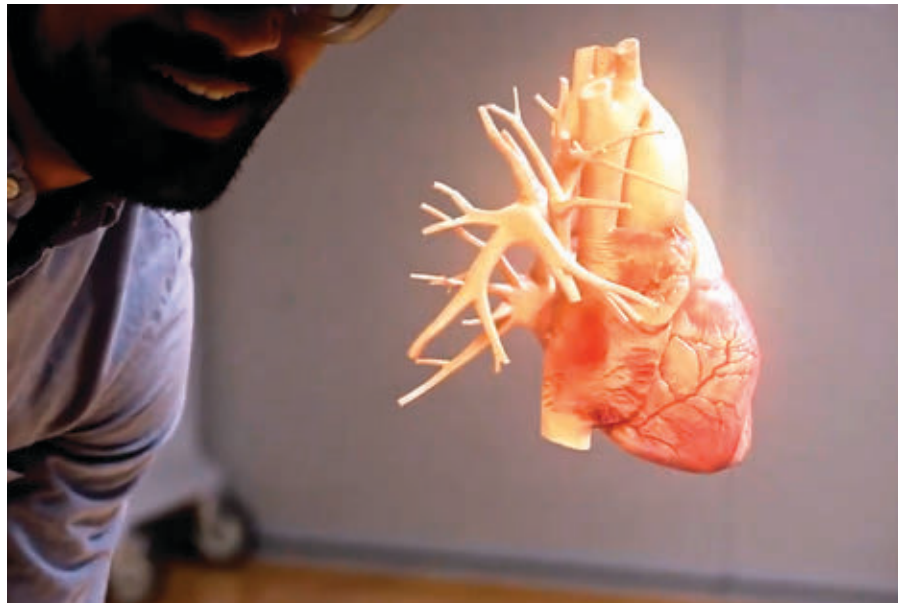
While 3D technologies that make headlines are not truly holographic, holographic techniques are furthering advances in important applications such as biomedical imaging.

IN RECENT MONTHS, one company after another has come out with products that appear to create holograms—but according to optics experts, most do not use true holography to create their three-dimensional (3D) effects.

“A lot of people abuse the word ‘holography,’” says James R. Fienup, Robert E. Hopkins Professor of Optics, and a professor of Electrical and Computer Engineering at the University of Rochester. “It’s kind of a catchy thing”—a quick way to evoke the futuristic coolness of this sci-fi staple—“so they call things ‘holograms’ that have nothing to do with holography.”

A notorious example is the so-called “Tupac hologram,” which stunned audiences at the 2012 Coachella music festival by appearing to show the rapper Tupac Shakur performing on stage years after he had been killed. The stunt, which became an Internet sensation, only reinforced the public’s misconception of what a hologram is. In fact, the effect didn’t use holography at all; rather, it repurposed a classic magician’s trick called Pepper’s Ghost, an illusion created through the clever use of carefully angled mirrors.

More recently, people have been using the word “holograms” for anything seen when you put on an augmented reality (AR) or virtual reality (VR) headset, says David Fattal, CEO of LEIA, an HP spinoff that has been developing a 3D display for smartphones. For example, Microsoft markets its HoloLens virtual reality headset as a form of “holographic computing,” and mainstream media typically describe the images seen through the device as “holograms,” though it is not clear what role holography plays in the technology. (Microsoft officials declined to be interviewed for this article.) Oculus, a competing prod-



Learning medicine in three dimensions with Microsoft’s HoloLens.

uct, also often gets described as holographic.

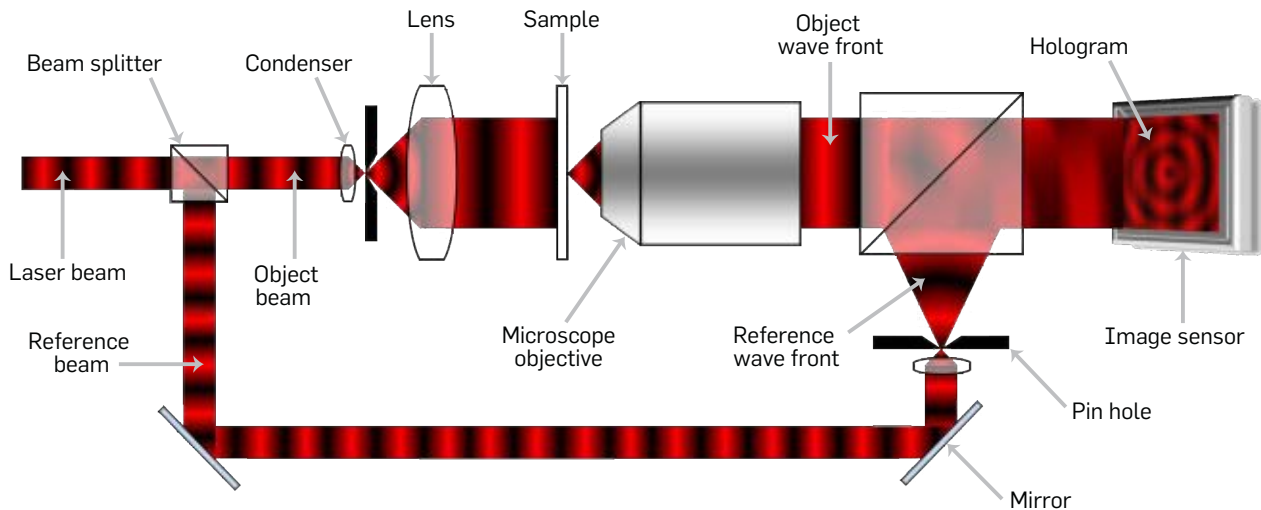
To most people, a hologram is any virtual object appearing in 3D form—even the images created using the simple stereoscopic effects seen through plastic 3D glasses. “That’s not the scientific definition,” Fattal says, adding that LEIA, too, is sometimes slammed at academic conferences for not using true holography.

True holography, in the scientific sense, refers to a process that uses wave interference effects to capture and display a three-dimensional object. The method, which goes back to the 1960s, uses two beams of coherent light, typically lasers. “You shine a laser on something, and the light scattered from that comes to your holographic sensor, and you also shine on that same sensor a beam from the same laser that hasn’t struck the object,” explains Fienup. “You interfere those two together and you capture the whole electromagnetic

field.” In fact, the “holo” in holography means “whole.”

The result is a set of interference fringes on the holographic film—a pattern of dark and bright regions that, unlike a photographic image, look nothing like the original object; therefore, seeing an image resembling the original object requires reconstruction. This happens by exposing laser light through the interference pattern, which functions as a diffraction grating that splits the light in different directions.

The key to getting the whole electromagnetic field—including the impression of depth—is holography’s capture of phase information, or the degree to which the light wave from the reference beam is out of step with the wave from the object beam. “What that provides is these interesting characteristics of three-dimensionality,” says Raymond Kostuk, a professor of Electrical and Computer Engineering, and of Optical Sciences, at



The optical setup of digital holographic microscopy.

the University of Arizona, who is using holography to develop more efficient processes for solar energy conversion, and cheaper methods of ovarian cancer detection. By capturing both phase and amplitude (intensity) information, holography shows more than do photographs, which capture information only about the intensity of the light.

Much of this process is now often done computationally, using CCD or CMOS cameras and algorithmic reconstruction. “Instead of recording on film, you record on the CCD camera, and then you store the information on a computer as a matrix,” explains Partha Banerjee, a professor of Electrical and Computer Engineering, and of Electro-optics, at the University of Dayton. To reconstruct the image, you process that matrix using well-known diffraction equations, which model how light waves propagate from one place to another—from the original object to the light sensor. “That’s digital holography,” says Banerjee, who was also general chair of this year’s Digital Holography and 3-D Imaging Conference, and who has used holography to capture the shape of raindrops or ice particles as they strike airplanes, to determine the three-dimensional characteristics of dents created from such impact.

One of the most popular applications of digital holography these days, says Banerjee and other experts, is in digital holographic microscopy (DHM), which aims at getting precise pictures of microscopic objects, particularly living cells and tiny industrial compo-

nents such as the ever-shrinking transistors printed on silicon wafers.

For example, Laura Waller, a professor of computer science and electrical engineering at the University of California, Berkeley, runs a Computational Imaging Lab that designs DHM tools for biological imaging, creating hardware and software simultaneously. “We’ve carefully designed our optical system so we’re getting enough information about the phase into our measurement,” she says, “and because we know the wave-optical physics model of the microscope, we can throw [the data we capture] into a non-linear, non-convex optimization problem so we can solve for the phase from these measurements.”

Living cells are completely transparent, but they are thick enough to delay the phase of a light beam; by measuring phase delays, researchers can map the shapes and densities of cells.

Using phase delays to make transparent specimens visible is not new—Frits Zernike earned the Nobel Prize in physics for similar work back in 1953—but traditional phase-contrast microscopy has drawbacks that DHM can overcome. “The Zernike phase-contrast microscope is a way of seeing those two things—the variations in thickness and the variations in density—but it’s not quantitative,” says Fienup. “It turns these phase variations into light and dark patterns, but you can’t tell exactly how much phase there was, how much thicker was it, or how much thinner was it—but with digital holography, you can actually measure

the density and thickness.”

Phase-contrast microscopes are also complicated pieces of machinery typically costing thousands of dollars. The DHM systems in Waller’s lab are more than an order of magnitude less expensive, she says; “they’re dirt-cheap, easy to use, and don’t have any special requirements. Then we use the computation to take on the burden that’s caused by doing that.”

Speed is of the essence when imaging biological samples. “We have about a half-second before the cells start moving around and everything gets blurred out,” Waller says. “We can’t just throw more and more data at it because the amount of data is constrained by how fast the camera can read it out.” One technique developed in Waller’s lab gets around the inherent trade-off between resolution and field of view by taking multiple low-resolution images of live cell samples across a wide field of view and computationally combining them to create high-resolution (gigapixel-scale) images.

In a related development called 4D holography, holographers add the dimension of time to show 3D objects in motion—for example, a holographic reconstruction of embryonic blood flow.

Although all these holographic techniques promise to aid both basic research and biomedical applications like early disease detection, what interests most people are moving images of people and ordinary, non-microscopic objects—to bring sci-fi effects into our daily lives. Unlike pseudo-holography,

a true holographic display would simulate a crucial characteristic of the way we see 3D objects in the real world: objects appear different from different points of view (parallax), and as we change our perspective this parallax experience is continuous, not jumpy, explains David Fattal of LEIA Inc. However, true holographic displays are currently impractical, Fattal says.

For one thing, creating diffraction patterns requires very small pixels—on the order of 100 nanometers, he says, whereas on today's screens the smallest pixel size is about 20 to 50 microns. "You're two or three orders of magnitude off, which means you'd need a screen of trillions of pixels, which is just ridiculous," Fattal says.

Real-time motion is even harder: making a holographic image move at a normal video rate requires recomputing the diffraction fringe to every 1/60th of a second—too fast for anything short of a supercomputer, even with the fastest available algorithms.

Yet Fattal is aiming to achieve holographic video effects not on a supercomputer or even a desktop machine, but on the smartphone, the most popular computing platform on Earth. LEIA, which will make its screens available to consumers through deals with mobile device manufacturers, has announced plans to ship its first screens by the end of 2017.

The trick, Fattal says, is breaking the hologram down into pieces, rather than treating it as a single image. "We take a generic hologram—you can think of it as a linear superposition of different arrays of light or different pieces of light coming from the different regions on the diffracting plane—and we manage to simplify the hologram, to think of it as different pieces," he says.

"The diffraction pattern can cater to different scenes—all we have to do is change the relative intensity of each portion," Fattal explains. "It's taking the best of holography in terms of image quality, but it's simplifying it and stripping it of superfluous information, and therefore we can make it move very quickly." Eventually, users will be able to interact with such 3D images by hovering over the smartphone screen rather than touching it, he says.

Such simplification is good enough, Fattal says, because of the limitations

Living cells, while transparent, are sufficiently thick to delay the phase of a light beam; by measuring phase delays, researchers can map the shapes and densities of cells.

of the human visual system. A hologram that contains all the information about a certain scene, he points out, contains too much information—including information to which your eye would never be sufficiently sensitive. "So if you know how to simplify the holographic rendering process, then you don't have to carry all the extra information, and that helps to make things move faster." **C**

Further Reading

Nehmettiah, G., and Banerjee, P.P. Applications of digital and analog holography in three-dimensional imaging, *Advances in Optics and Photonics*, Vol. 4, Issue 4, pp. 472-553 (2012) <https://www.osapublishing.org/aop/abstract.cfm?uri=aop-4-4-472>

Fattal, D., Peng, Z., Tran, T., Vo, S., Fiorentino, M., Brug, J., and Beausoleil, R.G. A multi-directional backlight for a wide-angle, glasses-free three-dimensional display, *Nature*, Vol. 497, March 21, 2013, <http://www.nature.com/nature/journal/v495/n7441/full/nature11972.html>

Kim, M.K. Principles and techniques of digital holographic microscopy, *SPIE Review*, May 14, 2010, <http://faculty.cas.usf.edu/mkkin/papers.pdf/2010%20SR%201%20018005.pdf>

Tian, L., Li, X., Ramachandran, K., and Waller, L. Multiplexed coded illumination for Fourier Ptychography with an LED array microscope, *Biomedical Optics Express*, Vol. 5, Issue 7, pp. 2376-2389 (2014), <https://www.osapublishing.org/boe/abstract.cfm?uri=boe-5-7-2376>

Based in San Francisco, **Marina Krakovsky** is the author of *The Middleman Economy: How Brokers, Agents, Dealers, and Everyday Matchmakers Create Value and Profit* (Palgrave Macmillan).

© 2016 ACM 0001-0782/16/10 \$15.00

ACM Member News

LEVERAGING THE CLOUD TO BE FRIENDLIER TO THE ENVIRONMENT



Babak Falsafi is a professor of Computer and Communication Sciences at the Ecole Polytechnique

Federale de Lausanne (EPFL) in Switzerland, where he directs the Parallel Systems Architecture Lab, which aims to bring parallel systems and design to the mainstream through research and education. This is fitting, as earlier in his career Falsafi designed a scalable multiprocessor architecture prototyped by Sun Microsystems (now Oracle).

As an undergraduate at the State University of New York at Buffalo, Falsafi earned degrees in computer science and electrical engineering. He garnered Master's and Ph.D. degrees in computer science at the University of Wisconsin, Madison, before taking a teaching position in electrical and computer engineering at Purdue University.

After three years at Purdue, he took a teaching post at Carnegie Mellon University, where he worked on the implications of power on design, and building shared memory systems. "I then moved to EPFL in 2008, after a sabbatical there in 2007."

Falsafi is founding director of the EcoCloud Center at EPFL, which works on energy-efficient and environmentally friendly cloud technologies. "My specific contributions are looking at server benchmarking with Cloudsuite, a benchmark suite for emerging scale-out applications, and designs like Cavium ThunderX," a new ARM-based server processor that is opening new doors for scale-out server workloads.

Babak also is interested in design for dark silicon, the transistors on a chip that must remain passive (dark) in order to stay within the chip's power budget.

—John Delaney

Battling Algorithmic Bias

How do we ensure algorithms treat us fairly?

COMPUTERIZED ALGORITHMS HAVE become an integral part of everyday life. Algorithms are able to process a far greater range of inputs and variables to make decisions, and can do so with speed and reliability that far exceed human capabilities. From the ads we are served, to the products we are offered, and to the results we are presented with after searching online, algorithms, rather than humans sitting behind the scenes, are making these decisions.

However, because algorithms simply present the results of calculations defined by humans using data that may be provided by humans, machines, or a combination of the two (at some point during the process), they often inadvertently pick up the human biases that are incorporated when the algorithm is programmed, or when humans interact with that algorithm. Moreover, algorithms simply grind out their results, and it is up to humans to review and address how that data is presented to users, to ensure the proper context and application of that data.

A key example is the use of risk scores used by the criminal justice system to predict the likelihood of an individual committing a future crime, which can be used to determine whether a defendant should be allowed to post bond and in what amount, and may also be used to inform sentencing if the defendant is convicted of a crime.

Pro Publica, a nonprofit investigative journalism organization, early this year conducted a study of risk scores assigned to more than 7,000 people arrested in Broward County, FL, during 2013 and 2014, to see how many arrestees were charged with new crimes over the next two years.

The risk scores were created by Northpointe, a company whose software algorithm is used widely within the U.S. criminal justice system. The scores were the result of 137 questions either answered by defendants or



pulled from criminal records, though the defendant's race is not one of the questions. Nonetheless, some of the questions highlighted by *Pro Publica*—"Was one of your parents ever sent to jail or prison?" "How many of your friends/acquaintances are taking drugs illegally?"—may be seen as being disproportionately impacting blacks.

Northpointe's founder, Tim Brennan, told *Pro Publica* it is challenging to develop a score that does not include items that can be correlated with race, such as poverty, joblessness, and social marginalization, since such negative traits that may indicate a propensity for criminal activity are correlated with race.

Still, according to *Pro Publica*, the risk scores examined across 2013 and 2014 proved unreliable in forecasting violent crimes, with just 20% of those predicted to commit such crimes actually doing so within two years. *Pro Publica* also claimed the algorithm falsely flagged black defendants as future criminals, wrongly labeling them this way at almost twice the rate of white defendants.

For its part, Northpointe disputed *Pro Publica*'s analysis, and the publication admitted the algorithm proved to be more accurate at predicting overall recidivism, with 61% percent of defendants being rearrested for committing a crime within two years.

It is not only the criminal justice

system that is using such algorithmic assessments. Algorithms also are used to serve up job listings or credit offers that can be viewed as inadvertently biased, as they sometimes utilize end-user characteristics like household income and postal codes that can be proxies for race, given the correlation between ethnicity, household income, and geographic settling patterns.

The New York Times in July 2015 highlighted several instances of algorithmic unfairness, or outright discrimination. It cited research conducted by Carnegie Mellon University in 2015 that found Google's ad-serving system showed an ad for high-paying jobs to men much more often than it did for women. Similarly, a study conducted at the University of Washington in 2015 found that despite women holding 27% of CEO posts in the U.S., a search for "CEO" using Google's Image Search tool returned results of which just 11% depicted women. A 2012 Harvard University study published in the *Journal of Social Issues* indicated advertisements for services that allow searching for people's arrest records were more likely to come up when searches were conducted on traditionally African-American names.

For their part, programmers seem to recognize the need to address these issues of unfairness, particularly with respect to algorithms that have the potential to adversely impact protected groups, such as those in specific ethnic groups, religious minorities, and others that might be subject to inadvertent or deliberate discrimination.

"Machine learning engineers care deeply about measuring accuracy of their models," explains Moritz Hardt, a senior research scientist at Google. "What they additionally need to do is to measure accuracy within different subgroups. Wildly differing performance across different groups of the population can indicate a problem. In the context of fairness, it can actually help to make models more com-

plex to account for cultural differences within a population.”

Tal Zarsky, a law professor at the University of Haifa, notes in a 2014 paper published in the *Washington Law Review* that identifying and eliminating cases of both explicit discrimination (cases in which the algorithm is specifically designed to treat some groups unfairly) and implicit discrimination (where the results of the algorithm wind up treating protected groups unfairly) may be challenging, but ultimately achievable. “While setting forth rules which ban such practices might be relatively easy, enforcing such a ban in a world in which the nature of the algorithm used is secret might prove to be a challenge,” Zarsky wrote.

Indeed, some observers have called on the organizations that write and use algorithms to be more transparent in terms of clearly spelling out the data collected, identifying which pieces of data are used in the algorithm, and disclosing how this data is weighted or used in the algorithm. Such insights may help to pinpoint areas of discrimination that may not be apparent otherwise.

“The blessing and the curse of being transparent is that you’re really clear, and with that clarity, sometimes you find discrimination,” explains Jana Eggers, CEO of Nara Logics, a Cambridge, MA-based artificial intelligence platform provider. “Because it’s uncovered, we go in and fix it, even if we have a lot to fix. Before, when we had the unconscious bias of people [making decisions], it was hard, if not impossible, to track down and understand.”

One solution for handling discrimination is to monitor algorithms to determine fairness, though it may be difficult to establish a common definition of fairness, due to a variety of competing interests and viewpoints. Indeed, business decisions (such as the decision to offer a mortgage or credit card) are often predicated on criteria that disproportionately impact some minority communities, while making sense for the company that wants to maximize profit and reduce risk.

“Our normative understanding of what is ‘fair’ is constantly changing, and therefore the models must be revisited,” Zarsky says.

Fairness is not necessarily clean-cut,

It may be difficult to establish a common definition of fairness, due to a variety of competing influences and viewpoints.

given the competing interests, whether looking at commercial interests (profit versus access to products and services) or within the justice system, which must balance public safety, administrative efficiency, and the rights of defendants.

That is why algorithms likely need to be reviewed and revised regularly with human input, at least for the time being, particularly with respect to their impact on protected classes. The U.S. federal government has established race, gender, and national origin as protected classes, and some states have added additional groups, such as sexual orientation, age, and disability status.

Common wisdom among programmers is to develop a pure algorithm that does not incorporate protected attributes into the model, and there are currently no regulations governing inadvertent discrimination as a result of an algorithm. However, Hardt says, “what my [research] collaborators and I realized early on is that in order to detect and prevent discrimination, it may actually help to take protected attributes into account. Conversely, blindly ignoring protected attributes can lead to undesirable outcomes.”

Despite their widespread use and potential to complicate the lives of many, it may be too early to establish a regulatory body for algorithms, given their complexity.

“Even the very notion of what we’re trying to regulate is delicate as many machine learning systems are complex pipelines that, unlike food ingredients, cannot be described succinctly,” Hardt says. “It would be more effective right now to invest in research on fairness, accountability, and transparency in machine learning.”

Indeed, the high potential costs associated with regulation may stall any reg-

ulatory activity, at least in the near term.

“Although the agency’s direct costs could be relatively low, the potential costs to some regulated entities could be relatively high,” says Andrew Tutt, a Washington, D.C.-based attorney and former Visiting Fellow at the Yale Law School Information Society Project. Tutt has suggested the creation of a federal regulator that would oversee certain algorithms in an effort to help prevent unfairness or discrimination, in much the way the National Highway Traffic Safety Administration (NHTSA) or the Food and Drug Administration regulate automobiles and pharmaceuticals, respectively, for safety.

“There is no doubt that in the formation of such an agency, a difficult balance will need to be struck between innovation on the one hand and other values, like safety, on the other,” Tutt says. “But I think that on balance, the benefits would outweigh the costs.”

Nevertheless, Tutt’s proposal only recommends oversight over algorithms that directly impact human safety, such as algorithms used to direct autonomous vehicles, rather than algorithms that may result in discrimination.

Hardt is not completely opposed to regulatory oversight, given that algorithms, and the way they are used, can do significant harm to many people. “I would like to see meaningful regulation eventually,” Hardt says. “However, I’m afraid that our technical understanding is still so limited that regulation at this point in time could easily do more harm than good.”

Further Reading

Zarsky, T.

Understanding Discrimination in the Scored Society, *Washington Law Review*, Vol. 89, No. 4, 2014. http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2550248

Big Data: Seizing Opportunities, Preserving Values, Executive Office of the President, May 2014, https://www.whitehouse.gov/sites/default/files/docs/big_data_privacy_report_may_1_2014.pdf

Narayanan, A.

CITP Luncheon Speaker Series: Arvind Narayanan – Algorithmic society, Center for Information Technology Policy, <https://www.youtube.com/watch?v=hujgRt9AsJQ>

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

© 2016 ACM 0001-0782/16/10 \$15.00



DOI:10.1145/2988441

Michael A. Cusumano

Technology Strategy and Management

The Puzzle of Japanese Innovation and Entrepreneurship

Exploring how Japan's unique mixture of social, educational, and corporate practices influence entrepreneurial activity.

AFTER LIVING IN Japan for seven of the past 40 years, I recently returned for an institutional development project at Tokyo University of Science. Tokyo University of Science is a private university founded in 1881 with over 20,000 students, and is the largest source of engineers and scientists for Japanese industry. The university is also the Japan host for an educational and research initiative called MIT REAP (MIT Regional Entrepreneurship Development Program).^a

We have been dealing with the following puzzle: Japan was once renowned for creating powerful, global companies, especially in manufacturing industries like automobiles, consumer electronics, semiconductors, and computer hardware. Japanese

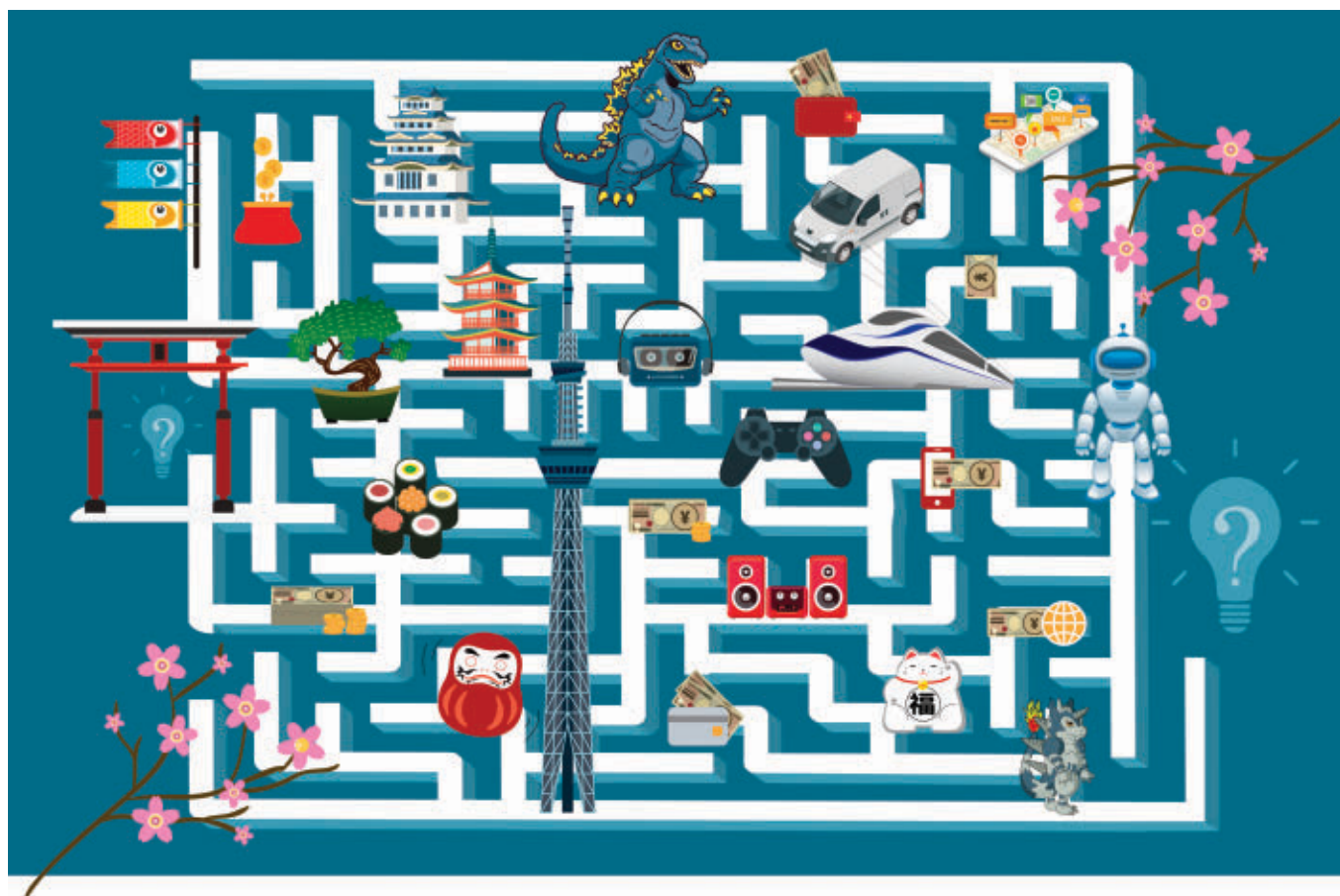
government and industry partnerships also once promised to revolutionize information technology, with bold initiatives such as the VLSI (Very Large-Scale Integration) Project of the 1970s for semiconductors and the Fifth Generation Computing Project of the 1980s for artificial intelligence. Japanese companies have since developed admirable hardware skills and competence in many aspects of software. But we no longer see bold innovation initiatives in Japan, nor do we see much entrepreneurial activity. What happened?

After opening the country to the West in the 1860s, a first generation of Japanese entrepreneurs organized large industrial conglomerates known as the “zaibatsu”—led by the Mitsui, Mitsubishi, and Sumitomo groups. They centered around mining, trading, and banking. Other firms appeared around World War I, led by Toshiba, NEC, Hi-

tachi, Nikon, Shiseido, Kobe Steel, and Matsushita (Panasonic). These firms brought in Western technologies and business practices. In the 1920s and 1930s, younger Japanese entrepreneurs started more technology-driven companies such as Toyota, Nissan, Fujitsu, Ricoh, and Canon.⁵ After World War II, another generation founded Honda, Sony, Nippon Telephone and Telegraph (NTT), and many other new firms.

Sony in particular combined advanced consumer electronics with sleek product designs, and inspired no less than a young Steve Jobs at Apple. Either the Sony Walkman, introduced in 1979, or NTT DoCoMo's i-mode “feature phone,” introduced in 1999, might have evolved into what became the Apple iPod and iPhone. They did not, as Japanese companies lagged behind in software, networking, and digital technologies. We still see this gap today, despite (or

^a See <http://reap.mit.edu/>



perhaps because of) Japan's penchant for quality, discipline, and detail in computer programming (see "The Puzzle of Japanese Software," *Communications*, July 2005).² As in software development, innovation and entrepreneurship require experimentation and risk-taking, and those attributes do not seem to be highly valued in today's Japan.

Japan still boasts many of the world's largest companies and iconic brands. Any visitor to Tokyo can also see that the country still possesses enormous wealth, creativity, and vitality. But interest in launching bold innovation initiatives and establishing pioneering new companies seems to have waned, especially compared to other developed countries.⁷ The Global Entrepreneurship Monitor even ranked Japan last among 24 developed countries in terms of entrepreneurial activity.⁶ The scarcity of new firms no doubt has contributed to some 30 years of sluggish, and sometimes negative, economic growth.

Recent data on venture capital shows Japan far behind China and the rest of Asia, as well as the U.S., though the Japanese do seem to understand that they need to create more startups that can

help grow the economy. Japanese venture funding in 2015 totaled just \$629 million. This compares to \$59 billion in the U.S.—nearly a 100-fold difference, even though the U.S. has only 2.5 times Japan’s population.⁹ The number of Japanese companies going public did reach an eight-year high in 2015 at 98.⁹ However, the total number of new Japanese companies being founded peaked in 2006 at 1,359 and fell to 809 in 2015, with stagnant levels of total invested capital. There has been relatively little infrastructure in Japan to promote entrepreneurship, such as in education and innovation centers at universities or private and public startup incubators, although this is changing.

The MIT REAP program likes to analyze regions in terms of “innovation capacity” (I-Cap) and “entrepreneurial capacity” (E-Cap). One measure of I-Cap, for example, is the number of patents a country or region produces given its population. One can also look at relative investment in R&D, networking infrastructure, universities, and other factors. One measure of E-Cap is the number of new firms being established. One can also look at availability

of private and public venture capital, availability of entrepreneurship education, or intentions of people at different ages to create new firms.

The Tokyo team is still gathering data, but Japan clearly appears to have the potential to create many more new companies than it does, especially in technology. Data comparing patent rates per population in different countries versus new firm creation shows Japan near the top among countries in this measure of innovation capacity but near the bottom in entrepreneurial activity. I suspect the Japanese can do better because historical data indicates Japan's low rate of startup creation is a relatively recent phenomenon. There were periods of very high activity following World War I and then again before and after World War II, as Japan modernized, militarized, and then rebuilt its post-war economy.^b

One reason for low entrepreneurial activity may have been the large amount of capital previously required

b This data was collected from the Tokyo Stock Exchange, the Japanese Ministry of Finance, and Mizuho Securities.

to register a company, now reduced from as much as \$100,000 to the equivalent of one cent.³ Another reason Japanese startup numbers seem low compared to the total number of firms may be because, unlike in the U.S., the Japanese are less inclined to dissolve existing firms, probably for tax reasons.¹ Other data suggests that Japan creates slightly larger companies than the average among OECD countries but then these companies tend not to grow very much, probably because of the paucity of venture capital until recently.⁴

Other factors inhibiting Japanese entrepreneurs are more difficult to quantify, such as social expectations combined with demographic trends and large-firm employment practices. For example, Japan has very low levels of unemployment (just over 3% in 2016) and a declining population. Nearly everyone graduating from university is guaranteed a good job, many until retirement. Since the vast majority of startups do not succeed, in any country, it is an enormous risk for young Japanese to create new companies. What if they fail? In the U.S., even people with failed startup backgrounds are considered to have valuable experience and can usually get good jobs in established companies. In Japan, companies recruit new employees mainly from new university graduates. In addition, in the U.S., entrepreneurs can separate corporate bankruptcy from private bankruptcy. In Japan, this is much more difficult to do. There is also a strong social stigma attached with failing as well as not following a conventional career path. Japanese parents expect their children (or spouses of their children) to get stable jobs with big companies or the government.

Startups from American universities also seem to benefit greatly from several practices that are rare in Japan. Classes mixing students from multiple schools (for example, engineering, science, and management) are common in the U.S. but infrequent and sometimes prohibited in Japan. Rigid rules often limit students and professors to classes and appointments in their individual faculties. It is difficult to launch an effective startup if all the members have only technical or only management backgrounds. Research on MIT startups showed this many years ago, indicating the single most important factor predicting the

Japan has continued to produce entrepreneurs, but they have not had much access to growth capital or experienced venture capitalists.

success of a technical venture was the existence of a founding team member with a background in sales or marketing (see my column “Evaluating a Startup Venture,” *Communications*, Oct. 2013, and Rogers).¹⁰

Direct government initiatives have played a relatively minor role in U.S. innovation and entrepreneurship, apart from massive defense spending and some medical science initiatives. However, in countries that lack large venture capital communities, or many private startup accelerators and incubators, government can play a big role. Japanese government ministries have taken various measures to encourage venture activities, entrepreneurship education, and funding. Key national research organizations have adopted modest programs to facilitate spin-offs. In 2016, there were also some 200 business plan competitions in Japan. Many connected to private initiatives such as Slush Asia, Samurai Venture Summit and Samurai Incubator, and the MIT Venture Forum of Japan.

Japanese universities have been slow to support entrepreneurship and have very limited funds of their own, but they are moving forward, too. A private venture fund (UTEC) closely connected with the University of Tokyo has been the clear leader, but Keio, Waseda, and a few other universities have also been active. They have received government support and established university-industry liaison programs, design labs, venture incubators, educational initiatives, and even small venture funds.

Japan’s particular mix of social, educational, and corporate practices, along with demographic realities, will

continue to hinder individual entrepreneurs who do not have government, university, or corporate support. At the same time, there is potential for more university-led and corporate entrepreneurship in the form of “spin-outs.” Large, established firms and some venture funds have the resources to fund new initiatives and run experiments, though big companies are usually not the best settings to tackle risky technologies and potentially disruptive business models.

So, what is the answer to our puzzle? The reality is that Japan has continued to produce entrepreneurs, but they have not had much access to growth capital or experienced venture capitalists. Nor have they gotten much encouragement and support from the government and universities, or society more broadly. The situation is now changing, and we should see Japan nurture yet another generation of entrepreneurs. This time, they will probably come more from large firms and a few leading universities rather than from the general population. It is an open question how much impact they will have on Japanese economic growth and venture creativity in the future, but I am hopeful. **C**

References

1. Beacon Reports. *Entrepreneurship in Japan: Separating Fact from Fiction* (Dec. 1, 2015); <http://bit.ly/2aZfs9b>
2. Cole, R.E. and Nakata, Y. The Japanese software industry: What went wrong, and what can we learn from it? *California Management Review* 57, 1 (2014), 16–43.
3. Corbin, D. Meet Yoshiaki Ishii, the Government Official Who Can Save Startups in Japan. *Techasia.com* (July 30, 2014); <http://bit.ly/2bHdqJT>
4. Criscuolo, C., Gal, P.N., and Menon, C. The dynamics of employment growth: New evidence from 18 countries. *OECD Science, Technology, and Industry Policy Papers* 14, OECD Publishing, 2014; <http://bit.ly/1jBWMNU>.
5. Cusumano, M.A. ‘Scientific Industry’: Strategy, technology, and entrepreneurship in prewar Japan.” In W. Wray, Ed., *Managing Industrial Enterprise: Cases from Japan’s Prewar Experience* (Harvard, 1989).
6. Entrepreneurs in Japan: Time to get started. *The Economist*, (Aug. 31, 2013).
7. *Global Entrepreneur Monitor 2014 Global Report*; <http://bit.ly/1SkQTnT>
8. Japan Venture Research, *Japan Venture Research Report 2015* (in Japanese, No. R0044-2, 3/30/2016), p. 7.
9. Martin, A. Japan tech hunts for restart button. *Wall Street Journal* (Apr. 10, 2016).
10. Roberts, E.B. *Entrepreneurs in High Technology: Lessons from MIT and Beyond* (Oxford, 1991).

Michael A. Cusumano (cusumano@mit.edu) is a vice president and dean at Tokyo University of Science, on leave as a professor at the MIT Sloan School of Management.

The author would like to thank the MIT REAP staff and the Tokyo team, especially Iris Wieczorek, Bill Aulet, Jun Tsusaka, Atsuko Fish, and Yoshiaki Ishii for help with information and insights, as well as Tommy Goji, founder of University of Tokyo Edge Capital.

Copyright held by author.

Global Computing

Mobile Computing and Political Transformation

Connecting increased mobile phone usage with political and market liberalization.

IN MUCH OF the world personal computing happens with mobile phones. In 2010 the South East Asian country of Myanmar (aka Burma) had the world's lowest rate of mobile phone penetration. Only 1% of its population had a mobile phone subscription, about the same as its landline penetration.⁶ The same year war-torn Somalia experienced 7% mobile penetration, and even North Korea had 1.8%. By 2014 Myanmar had jumped to 54% mobile penetration, Somalia had 50%, and the North Koreans just 11%. The government of Myanmar anticipates 80% penetration sometime this year.¹

After 20 years of military rule, Myanmar held its first election in 2010. The political party aligned with the government won 80% of contested seats but international observers called the election a sham.⁹ The opposition National League for Democracy (NLD, the party of Peace Prize laureate Aung San Suu Kyi) didn't even bother to contest. Contrast November of 2015 when Myanmar held another national election and this time it was viewed as successful.¹² The NLD won just under 80% of the contested seats and today is standing up a new government with a close Suu Kyi confidently tapped as president.^a

^a Thanks to a shrewdly crafted national constitution the military of Myanmar still enjoys considerable power including set-asides in the legislature and key cabinet posts and a prohibition against Suu Kyi herself serving as President.



Myanmar's National League for Democracy party leader Suu Kyi is shown on a cellphone screen held by a supporter celebrating election results last November.

It is a stunning set of transformations: In just six years Myanmar increased its mobile phone use by fifty-fold and went from strongman military control to democratic rule. These transformations are both concomitant and connected. After the 2010 sham election the military-controlled government of Myanmar embarked on a series of political transformations. These included real, multi-party elections and a set of market liberalizations that included telecommunication sector reform. The hoped-for outcome of these reforms was inclusive growth brought about by rigorous, well-regulated and

nondiscriminatory competition in both the electoral and telecommunication systems.^b

In Myanmar's electoral system, competition came from vigorous participation (and ultimately the landslide victory) of the opposition NLD party. In telecommunications, competition arose when two private sector opera-

^b National monopolies are fine for some systems (perhaps healthcare and education for instance) but are fraught as a political system.¹¹ In telecommunications, well-regulated nondiscriminatory competition has demonstrated broad subscriber benefit though unsound deregulation can temper this.⁸

**Sex as an Algorithm:
The Theory of Evolution
Under the Lens of
Computation**

**Technology and
Academic Lives**

**Recommender Systems—
Beyond Matrix
Completion**

**Is It Time to Reinspect
the Foundations?**

**Spark: Building
a Unified Engine for
Big Data Processing**

**Increasing Password
Strength in the
'Don't Care' Region**

**A Theory on Power
in Networks**

**Extracting 3D Objects
from Photographs
Using 3-Sweep**

Plus the latest news about
blockchain as an enabler,
smart farming with robots,
and deceiving AI.

tors (Ooredoo from Qatar and Telenor of Norway) came in to compete with the preexisting state operator, MPT. The result was explosive growth in services and a precipitous decline in the cost to use a mobile phone. The average price to purchase a SIM card dropped from \$150 in 2013 to \$1.5 today.² Today many people in Myanmar can afford mobile phones and are using data plans along with voice apps, such as Viber and Skype. They are also using text-based apps, such as Whatsapp, to stay in touch. One-third of recent survey respondents reported accessing Facebook from their phones.⁵

Further development of Myanmar's mobile phone market is expected as sector liberalization proceeds along an internationally established path:

► In 2013, the enactment of a new Telecommunications Law allowing non-state entities including foreign companies to bid for telecommunication service licenses.

► In 2014, the introduction of service by private operators Telenor and Ooredoo using licenses provided for under the new Telecom Law.

► In 2015, steps toward establishing an independent sector regulator. While the Telecom Law made provisions for the establishment of an independent regulator by October 2015, instead by Presidential directive an interim commission was created with the sole task of ensuring that the law to establish the regulator was prepared. In the interim, the Post and Telecommunications Depart-

Myanmar's Digital Gender Divide

In May 2015, 47% of men in Myanmar owned a mobile phone but only 33% of women did.¹⁴ A logistic regression shows that being a woman reduces the odds of owning a phone by 42%, even after controlling for gender differences in education, income, having a TV and electricity at home, having friends with mobile phones and a host of other variables that impact phone ownership.

Because women who don't own a phone are willing to borrow someone else's phone for basic services, they are still able at times to make and receive calls and SMSs. However, they are less likely to borrow a phone for Internet browsing. The odds of using the Internet increases 8% with every unit increase in phone ownership. Not owning a phone negatively impacts Internet use and getting more women to own their own handset is a key step in getting them online.

When asked why they don't own a phone, the top two reasons Myanmar women gave were that they "cannot afford a handset" (38% of non-owning women) or they "have no use for it" (34%). Many emerging economies record gender gaps in ownership, and similar reasons have been cited in Asian household ICT surveys in the past.

However, unlike in some parts of Asia, women in Myanmar have a (well-documented) strong position in the household. Culturally the chief financial officers of the family, women in Myanmar are directly involved in spending decisions, including whether to buy a phone. When purchased, the phone is given to the person who "needs" it the most—often defined as someone who lives or travels outside the home. This is generally the man (who works outside) or son/daughter (who studies outside). A second phone in the household is therefore important to increase women's Internet access.

Women also typically lack digital skills and know-how compared to men. So even when women are involved in the financial decision to buy a phone, it is the man who ultimately chooses the specific model, operator, and apps. Many women in Myanmar do not possess the skills and knowledge to begin using data services and have to rely on others (primarily male relatives or men who work in phone shops) for instructions. This limits access and use as many women, especially in rural settings, feel uncomfortable asking men for help.

Technological adoption is always contoured by politics, economics, and social norms. While more people today in Myanmar are benefitting from mobile telephony the benefits are not equally distributed. Age, gender, and economic standing all come into play. Smart policies and programs are needed to narrow not just the access gap but the digital gender gap in Myanmar.

Helani Galpaya is (helani@lirneasia.net) is the chief executive at LIRNEasia, a pro-poor, pro-market organization working across the emerging Asia Pacific on ICT policy and regulatory issues.

ment in the Ministry of Communications and Information Technology continues to be responsible for the regulatory function (T.D. Norbhu, personal communication, February 29, 2016).

► And ultimately reform of the incumbent national operator, the state-owned MPT, which is slowly restructuring into a commercial entity and may eventually be privatized.

Political transformation in Myanmar enabled this market liberalization in the telecommunication sector and the subsequent explosion of mobile phone use. Now the presence of these phones is producing explosive change to politics. During the 2015 election cycle much was made of Myanmar's "digital election;" citizens traded po-

litical information online and election-monitoring apps proliferated.⁷ But this change was not all positive: new communication technologies served as tools of democratic progress in 2015, but were also used to propagate hate speech, inflame ethnic strife, and diminish democratic growth. Most notoriously, online anti-Muslim messages originated from prominent Buddhist monks, prompting a countermovement by a coalition of civil society activists called the "flower speech" campaign to combat online hate speech and promote communal understanding.¹³

Scholars have long discussed the Janus-faced role of communication technologies on electoral politics and democratic development.⁴ As seen in other countries, mobile phones and

the Internet can be catalysts of positive democratic change, but they can also be tools for minority subjugation and state control.³ Myanmar offers yet another example of the multiple valences these technologies embody. Social media can be a tool for democratic deepening, hate speech, and political control, all at once.¹⁰

Today, Myanmar may be the world's most exciting telecommunications sector in addition to being one of the world's most quickly changing political environments. Technologists cannot ignore political and policy environments. They often trump technology. Moreover, policy-makers and politicians cannot ignore Internet and mobile phone technologies. They must ensure the digital revolution supports and does not undermine positive political transformations and inclusive growth. Political and digital transformation go hand-in-hand—you cannot have one without the other. 

References

1. Abt, M. Foreign operators seek to unlock Burmese potential. GSMA Intelligence, London, 2013.
2. Alliance for Affordable Internet. Delivering Affordable Internet Access in Myanmar. A4AI, Washington, D.C., 2015.
3. Best, M.L. and Meng, A. Twitter democracy: Policy versus identity politics in three emerging African democracies. In *Proceedings of the Seventh International Conference on Information and Communication Technologies and Development*, ACM, New York, 2015, pp. 20:1–20:10; <http://doi.org/10.1145/2737856.2738017>
4. Best, M.L. and Wade, K.W. The Internet and democracy: Global catalyst or democratic dud? *Bulletin of Science Technology Society* 29, 4 (2009), 255–271.
5. Galpaya, H., Zainudeen, A., and Suthaharan, P. A Baseline Survey of ICT and Knowledge Access in Myanmar. LIRNEasia, Colombo, Sri Lanka, 2015.
6. ITU. World Telecommunication/ICT Indicators Database 2015. Geneva, ITU.
7. Kyaw, K.P. and Thu, M.K. Myanmar's digital election. *Frontier Myanmar* (Oct. 27, 2015); <http://bit.ly/2bkCULv>
8. Laffont, J.J. and Tirole, J. Competition in telecommunications. MIT Press, 2001; <http://bit.ly/2bfbtGp>
9. Macfarquhar, N. U.N. doubts fairness of election in Myanmar. *The New York Times* (Oct. 21, 2010); <http://nyti.ms/2aS2ONU>
10. Pietropaoli, I. Myanmar: Facebook should warn users about risks of self-expression. *The Guardian* (Nov. 2, 2015); <http://bit.ly/20oMzsy>
11. Sen, A. *Democracy as Freedom*. Oxford University Press, 1999.
12. The Carter Center. Observing Myanmar's 2015 General Elections Final Report. Atlanta, GA, 2016.
13. Trautwein, C. Sticking it to hate speech with "flowers." *The Myanmar Times* (Mar. 2015). Yangon, Myanmar.
14. Zainudeen Z. and Galpaya H. Mobile phones, Internet and gender in Myanmar. London, GSMA, 2015.

Michael L. Best (mikeb@cc.gatech.edu) directs the United Nations University Institute on Computing and Society (UNU-CS) in Macau SAR, China. He is associate professor, on leave, with the Sam Nunn School of International Affairs and the School of Interactive Computing at Georgia Institute of Technology where he directs the Technologies and International Development Lab.

Copyright held by author.





DOI:10.1145/2988447

George V. Neville-Neil

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

Kode Vicious Cloud Calipers

Naming the next generation and remembering that the cloud is just other people's computers.

Dear KV,

Why do so many programmers insist on numbering APIs when they version them? Is there really no better way to upgrade an API than adding a number on the end? And why are so many systems named “NG” when they’re clearly just upgraded versions?

API2NG

Dear API2NG,

While software versioning has come a long way since the days when source-code control was implemented by tapping file names to hacky sacks in a bowl in the manager’s office, and file locking was carried out by digging through said bowl looking for the file to edit, programmers’ inventiveness with API names has not advanced very much. There are languages such as C++ that can handle multiple functions—wait, methods with the same names but different arguments—but these present their own problems, because now instead of a descriptive name, programmers have to look at the function arguments to know which API they’re calling.

Perhaps the largest sources of numbered APIs are the base systems to which everyone programs, such as operating systems and their libraries. These are written in C, a lovely, fancy assembler that has nothing to do with such fancy notions as variant function signatures. Because of this limitation of the language that actually does most of the work on all of our collective be-



halves, C programmers add whole new APIs when they only want to create a library function or system call with different arguments.

Take, for example, the creation of a pipe, a very common operation. Once upon a time, pipes were simple and returned a new pipe to the program, but then someone wanted new features in pipes, such as making them nonblocking and making the pipe close when a new sub-program is executed. Since `pipe()` is a system call defined both by the operating system and in the Posix standard, the meaning of `pipe()` was already set in stone. In order to add a flags argument, a new pipe-like API was required, and so we got `pipe2()`. I would say something like “Ta-da!” but it’s more like the sad trombone sound. Given that the system-call interface is

written in C, there was nothing to do but add a new call so that we could have some flags. The utter lack of naming creativity is shocking. So now there are two system calls, `pipe()` and `pipe2()`, but it could have been worse: we could have had `pipeng()`.

Perhaps the worst thing that Paramount ever did was name its *Star Trek* reboot *The Next Generation*, as this seems to have encouraged a generation of developers to name their shiny new thing, no matter what that thing is, ThingNG. Somehow, no one thinks about what the next, next version might be. Will the third version of something be ThingNGNG? If your software lasts a decade, will it eventually be a string of NGs preceded by a name? The use of “next generation” is probably the only thing more aggravating than numeric

indicators of versioned APIs.

The right answer to these versioning dilemmas is to create a descriptive name for the newer interface. After all, you created the new version for a good reason, didn't you? Instead of `pipe2()`, perhaps it might have made sense to name it `pipef()` for "pipe with a flags argument." Programmers are a notoriously lazy lot and making them type an extra character annoys them, which is another reason that versioned APIs often end in a single digit to save typing time.

For the time being, we are likely to continue to have programmers who version their functions as a result of the limitations of their languages, but let's hope we can stop them naming their next generations after the next generation.

KV

Dear KV,

My team has been given the responsibility of moving some of our systems into a cloud service as a way of reducing costs. While the cloud looks cheaper, it has also turned out to be more difficult to manage and measure because many of our former performance-measuring systems depended on having more knowledge about how the hardware was performing as well as the operating system and other components. Now that all of our devices are virtual, we find that we're not quite sure we're getting what we paid for.

Cloudy with a Chance

Dear Cloudy,

Remember the cloud is just other people's computers. Virtualized systems have existed for quite a while now and are deployed for an assortment of reasons, most of which have to do with lower costs and ease of management. Of course, the question is whose management is easier. For services that are not performance critical, it often makes good sense to move them off dedicated hardware to virtualized systems, since such systems can be easily paused and restarted without the applications knowing that they have been moved within or between data centers.

The problems with virtualized architectures appear when the applications have high demands in terms of storage or network. A virtualized disk might try

to report the number of IOPS (I/O operations per second), but since the underlying hardware is shared, it is difficult to determine if that number is real, consistent, and will be the same from day to day. Sizing a system for a virtualized environment runs the risk of the underlying system changing performance from day to day. While it's possible to select a virtual system of a particular size and power, there is always the risk that the underlying system will change its performance characteristics if other virtualized systems are added or if nascent services suddenly spin up in other containers. The best one can do in many of these situations is to measure operations in a more abstract way that can hopefully be measured with wall-clock time. Timestamping operations in log files ought to give some reasonable set of measures, but even here, virtualized systems can trip you up because virtual systems are pretty poor at tracking the time of day.

Working backward toward the beginning, if you want to know about performance in a virtualized system, you will have to establish a reliable time base, probably using NTP (Network Time Protocol) or the like, and on top of that, you will have to establish the performance of your system via logging the time that your operations require. Other tools may be available on various virtualized environments, but would you trust them? How much do you trust other people's computers?

KV

 **Related articles**
on queue.acm.org

APIs with an Appetite

Kode Vicious

<http://queue.acm.org/detail.cfm?id=1229903>

Arrogance in Business Planning

Paul Vixie

<http://queue.acm.org/detail.cfm?id=2008216>

Cybercrime 2.0: When the Cloud Turns Dark

Niels Provos, Moheeb Abu Rajab, and Panayiotis Mavrommatis

<http://queue.acm.org/detail.cfm?id=1517412>

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

Copyright held by author.

Calendar of Events

October 2-5

BCB '16: ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics
Seattle, WA,
Sponsored: ACM/SIG,
Contact: Umit V. Catalyurek,
Email: catalyurek.1@osu.edu

October 2-7

MODELS '16: ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems
Saint-Malo, France
Contact: Benoît Combemale,
Email: benoit.combemale@irisa.fr

October 5-7

SoCC '16: ACM Symposium on Cloud Computing
Santa Clara, CA
Co-Sponsored: ACM/SIG,
Contact: Brian Cooper,
Email: brianfrankcooper@gmail.com

October 3-7

MobiCom'16: The 21th Annual International Conference on Mobile Computing and Networking
New York City, NY,
Sponsored: ACM/SIG,
Contact: Marchco Oliver Gruteser,
Email: gruteser@winlab.rutgers.edu

October 10-12

MiG '16: Motion In Games
Burlingame, CA,
Sponsored: ACM/SIG,
Contact: Michael Neff,
Email: neff@cs.ucdavis.edu

October 11-14

RACS '16 : International Conference on Research in Adaptive and Convergent Systems
Odense, Denmark,
Sponsored: ACM/SIG,
Contact: Esmaeil S. Nadimi,
Email: esi@mmmi.sdu.dk

October 15-16

SUI '16: Symposium on Spatial User Interaction
Tokyo, Japan
Co-Sponsored: ACM/SIG,
Contact: Christian Sandor,
Email: christian@sandor.com

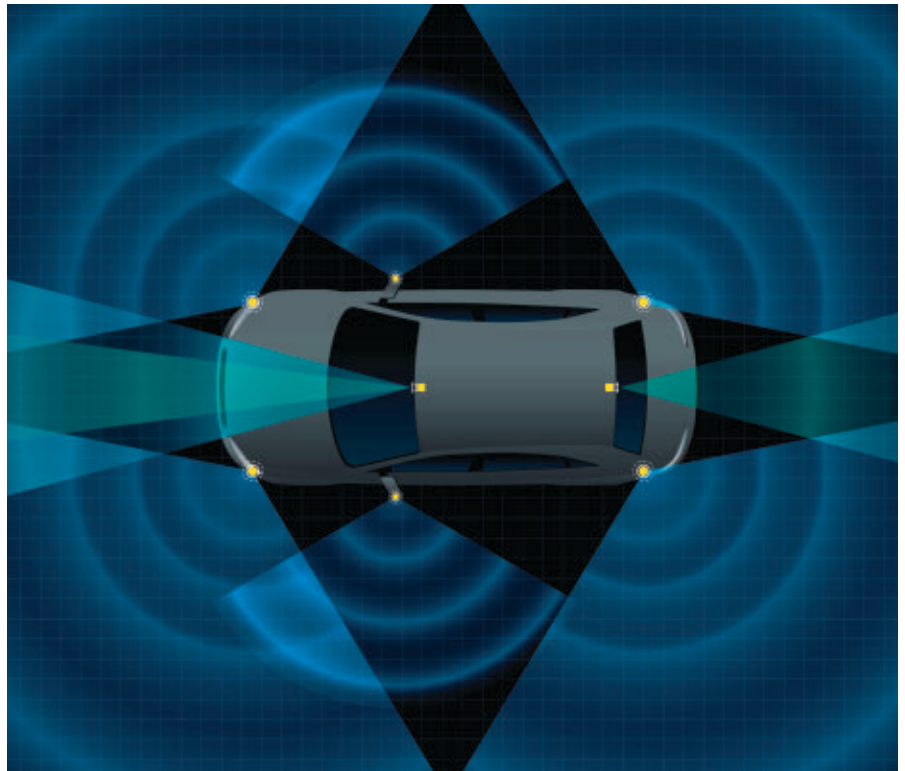
Inside Risks

Risks of Automation: A Cautionary Total-System Perspective of Our Cyberfuture

Where automation is inevitable, let's do it right.

MANY COMPUTER-RELATED RISKS discussed in past Inside Risks columns are still present today. These risks (and new ones) are likely to intensify even further as systems provide extensive automated or semi-automated operation. Significantly greater total-system trustworthiness will be required, encompassing better hardware, system software, and applications that are able to tolerate human limitations and environmental factors. Risks will continue to result from inadequate reliability, security, and privacy, as well as gullibility and general inability of users to cope with complex technology. We repeatedly discover unexpected risks resulting from lashing subsystems together (for example, see Beurdouche²), because of unexpected system behavior. Many advances in research, system development, and user friendliness are urgently needed. Also, some middle ground is desirable between the optimists (who believe there are easy answers to some of the problems posed here) and the pessimists (who have serious doubts about increasing uses of automation and artificial intelligence—especially when used by people who are more or less technologically queasy).

In this column, I examine certain approaches that might be economically desirable, but that have serious



potential risks. These include aviation safety and security; self-driving and semi-automated vehicles, and eventually automated highways; the so-called Internet of Things; and cloud computing and cloud storage.

Total-system trustworthiness must recognize requirements for human safety, security, reliability, robustness, and resilience despite adversities such as human error, attacks, and malware.

However, we also need proactive system architectures that inherently minimize the extent to which various components have to be trusted, and other requirements such as extensive monitoring, auditability, interoperability, compatibility, and predictable composability of components to enable facile multi-vendor systems. For example, voice and speech recognition and understanding, automatic translation, intelligent

dialogues, and automated responses have some potentials to compromise trustworthiness. Also, we must depend upon systems and networks that are intrinsically untrustworthy in various respects—and sometimes made even less so by human frailty, insider misuse, and potential governmental desires for exceptional accesses that bypass already marginal security (for example, see Abelson et al.¹). As a result, we need people-tolerant systems as well. Above all, we will need scalability of the implementations with respect to all of the requirements mentioned here (whether or not individual local control is also desired), plus the inevitable desire for remote upgrades to quickly remediate system vulnerabilities and to enable new applications. All of this is very daunting in light of the reality that we are trying to evolve incrementally from today's flaky platforms. Thus, we might wonder whether some of these desiderata are actually pipedreams that cannot be implemented, maintained, and used with sufficient assurance that the remaining risks will be acceptable. No system is ever going to be perfect—especially ones that require considerable autonomy in operation. However, the question of what is good enough always remains; it cannot be answered generally, largely because there are different answers depending on the specific applications.

Aviation Safety and Security

We are already pushing the edges with regard to aviation safety and security in the large. Developing avionics system hardware and software that cannot be subverted accidentally or intentionally is demonstrably nontrivial and expensive, but only a small part of the overall problem. This was originally conceived as the *Free-Flight* program, putting much greater smarts in cockpit control systems—so that air-traffic controllers on the ground might become less critical in real time. For example, collision-avoidance systems are now well established and generally reliable. Free-Flight has now morphed more generally into the total-system *NextGen* program, which will integrate ground- and air-based controls. However, the notion of having safe distributed heavily automated control among nearby aircraft in the broader context of airport and long-range en-

No system is ever going to be perfect—especially ones that require considerable autonomy in operation.

route scheduling, with real-time total traffic control (especially in times of inclement weather delays) could introduce many potential risks. In that air-traffic controllers and pilots today may be sorely pressed in times of heavy congestion and erratic weather conditions, providing them with more intelligent computer-aided relief should be beneficial—if it can be assuredly provided. For example, the new DO-178C certification tool suite has evolved significantly, and is considerably more advanced than its predecessors. It offers significant hopes that we can further increase flight safety and security.

Aviation safety and security are of course a worldwide concern, not just a domestic one, especially with many different countries and languages—and problems requiring emergency remediation. Enormous progress has been made along these lines, although there are still corner cases that may defy adequate control and require pilot attention (and possible intervention). However, putting most of the controls in the hands of integrated automation must encompass hardware, software, communications, pilots who might or might not be able to override computer controls in emergencies, ground controllers with excellent training and experience, and defenses against would-be intruders. Infotainment systems have tended to coexist on the same local network with the aircraft controls, perhaps without adequate separation. The total-system approach must therefore develop stronger network security to ensure that the flight-control systems are strongly isolated from the infotainment and systems.

Other problems within the total-system perspective include airport safety and security, passenger screening, timely preventive aircraft maintenance, and thorough pilot training that anticipates unexpected events. We tend to put our eggs in a few defense mechanisms (including those that were not previously present to thwart past compromises); however, that is not a viable strategy when there are too many vulnerabilities.

It is also necessary to consider the presence of remotely controlled drones sharing the air space, and all of the risks to human safety and privacy, in flight and on the ground. Drones (mostly semi-autonomously or manually controlled at present, although they could be fully autonomous in the future) will require better security to prevent subversion akin to that demonstrated in modern automobiles—particularly, drones carrying lethal weapons.

Automotive Safety in Automated Vehicles

Total-system safety and security concerns include the demonstrated ability to compromise the controls of conventional vehicles—for example, through the wireless maintenance port or otherwise gaining access to the internal local network. Those problems must be addressed in vehicles with self-driving or highly automated features. Note that a distinction is made here between self-driving cars (for example, Google, albeit with a surrogate driver during the current test and evaluation phases, but with the intention of becoming fully autonomous) and computer-augmented driver assistance (for example, Tesla) that goes way beyond more familiar features such as cruise control, airbags, anti-lock braking, parallel parking, rear-vision video, and other recent enhancements for safety and convenience, but that falls somewhat short of fully autonomous control with no ability for manual intervention.

Those of us who live in the California Bay Area frequently encounter self-driving Google cars. The accident rates thus far are very low, in part because the vehicles are programmed to aggressively observe traffic signs and environmentally changing road conditions—usually with the surrogate driver ready to override. (There are cases of Google vehicles being hit from behind

by human drivers—primarily because of Google’s conservative programming; it is thought that the cars running into them may be following too closely, with drivers who are not cognizant of the conservative nature of the Google car.) The desires for dramatically reducing accident rates through vehicle automation seem realistic, although there are always likely to be unanticipated corner cases. Incidentally, Google has monitored some of the surrogate drivers, and discovered they tended not to be paying strict enough attention—perhaps because the vehicles performed so well! In any case, the record of self-driving Google vehicles seems vastly better than that of old-fashioned human-driven ones. Recognizing that the evolving automation is still a work in progress, there is considerable hope.

Unfortunately, the “driver” of a Tesla S died on May 7, 2016, in a crash in Florida while his car was in the automated-assistance mode.^a This is reportedly the first known fatal accident involving a vehicle under automated control. Joshua Brown (a Navy veteran who had founded his own technology consulting firm) was in the driver’s seat with no hands on the steering wheel, and was an outspoken advocate of the safety of the automated controls. (Recent reports suggest that he was watching a Harry Potter movie.) The cited article states that “Neither the Autopilot nor the driver noticed the white side of a tractor-trailer [which made a left turn in front of the Tesla] against a brightly lit sky, so the brake was not applied.” The crash seems to cast doubts on whether autonomous vehicles in general can consistently detect all potential life-threatening situations. However, after a reported million miles of driving, a single fatality may not be particularly significant. This is far better than human driving. Although the details raise concerns, even seemingly perfect automation would still lead to accidents, injuries, and deaths; even with automation, nothing is actually perfect.

Karl Brauer (a Kelley Blue Book analyst) was quoted: “This is a bit of a wake-up call. People were maybe too aggressive in taking the position that we’re almost there, this technology is going to be in the market very soon, maybe need

Recognizing that the evolving automation is still a work in progress, there is considerable hope.

to reassess that.” However, Elon Musk has praised the Tesla Model S as “probably better than a person right now.” Also, a Tesla statement on June 30 noted that driving a Model S with this technology enabled (as a beta-tester!) “requires explicit acknowledgment that the system is new technology.”

An immediate reaction to the Tesla “Autopilot” is that it should not be called an *autopilot*, because it explicitly demands constant attention from the person in the driver’s seat. This misnomer has been raised repeatedly—especially in the aftermath of the recent accidents.

The Tesla involved in Brown’s death did not have LIDAR (Light Detection and Ranging) pulsed lasers, and was relying on the Mobileye camera and forward-facing radar.^b It is clear that many improvements can be added (such as LIDAR)—not just to the vehicle controls, but also by automating the sensors and signals in roadways and particularly in dangerous intersections themselves, dynamically establishing different speed limits under bad weather conditions, and much more.

On July 6, 2016, reports appeared that a Tesla X on “Autopilot” lost control on the Pennsylvania Turnpike, bounced off concrete guard rails, and flipped over; the passenger in the driver’s seat was reportedly not paying enough attention, and was injured.^c

John Quain⁷ notes there is significant evidence that a driver behind the wheel may not be ready to take over from the autopilot quickly enough to avert a disaster: “Experiments conducted last year by Virginia Tech re-

searchers and supported by the national safety administration found that it took drivers of Level 3 cars [in which the driver can fully cede control of all safety-critical functions in certain conditions] an average of 17 seconds (!!!) to respond to takeover situations. In that period, a vehicle going 65 mph would have traveled 1,621 feet—more than five football fields.”^d

Generalizing this situation, a huge question seems to arise regarding liability—where litigation tends to look for deep pockets. But there are many issues here. Perhaps when you buy an automated vehicle, the contract might stipulate that the car is *experimental* and that the maker disclaims liability, explicitly waiving responsibility. (This is somewhat akin to the providers of the most common operating systems declaring that these systems should not be used for critical applications—although that caveat seems to be widely ignored.) In that case, the maker’s lawyers might successfully claim that a driver was negligent by having too much faith in the software/hardware system. The legal issues are further complicated if highway patrols insist on backdoors to be able to redirect or stop vehicles for inspection or arrest, which itself might cause an accident or a violent action. And what happens when two or more totally driverless autonomous vehicles actually collide? Or when a remotely controllable vehicle is coopted for evil purposes? There are vastly too many risks to enumerate here, and much more research, development, and evaluation are needed.

In particular, consider two highly relevant papers by Don Norman^{3,6} well worth reading. Don contributed some pithy quotes for my ACM *Ubiquity* July 2016 article⁴ on this subject. He believes that partial automation is a disaster waiting to happen, and that total automation is essential. “To think otherwise is to ignore decades of solid research from the psychology and human factors fields (and the National Academy’s Human Systems Integration board). And there is no way to overcome it. The better the [partial] automation, the more dangerous it be-

a See <http://bit.ly/2aRzPqX>

b See <http://bit.ly/297eo4D>

c See <http://bit.ly/2aYNzBD>

d Vlasic, B. and Boudette, N. *The New York Times* (July 1, 2016), with follow-up posts by Bill Vlasic the following day and week; <http://nyti.ms/2b2QC91>

comes. It has to be full automation, not this silly Level 3.”

However, introducing automation into activities already regulated by standards that were not formulated with automation and security in mind can introduce risks. Also, lack of infrastructural investment and demands for incremental change with backward compatibility may be impediments to progress toward safety and security.

While writing this column, I learned of the Automotive Information Sharing and Analysis Center (Auto-ISAC, which has assembled a set of best practices) and The Billington Global Automotive Cybersecurity Summit (which had its inaugural meeting on July 22, 2016). These efforts seem to echo my concern that safety and security must be considered together throughout the automotive industry. Indeed, they claim to do so without seeking to make security a competitive advantage for individual companies, to learn what they can from other sectors, and to make fully autonomous cars available on an ordinary retail basis within the next 10 years.^e

Automated Highways

The concept of every vehicle on a highway being automated (without fear of accidents or frustrations from congestion) still may seem somewhat remote. It will ultimately rely on highly collaborative coordination among neighboring vehicles in addition to the automation and semi-automated assists noted in the preceding section, and trustworthy communications with neighboring vehicle controllers and road hazards. In addition, some sort of total-system traffic monitoring is going to be essential, especially in detecting and responding to accidents, extreme weather conditions, vehicles running out of fuel or battery, flat tires, and more. Another concern is of course introducing older vehicles (with minimal autonomy and real-time monitoring) into the mix, or perhaps living with a simpler solution—barring such legacy vehicles from the automated highway and forcing them

onto back roads. The two-dimensional control problems may be slightly less challenging than the three-dimensional aircraft flight control problems, but nonetheless important, particularly in potential emergencies. However, the separations among moving objects, the human vs. automated reaction times, and the ensuring risks differ widely among in-flight aircraft and ground vehicles. In some ways, the automation problem in aviation is simpler than that with automobiles. Pattern recognition for a variety of objects in confusing backgrounds is critical in automobiles; in airplanes, one simply has to detect the presence of an object—because the exact identity does not much matter (except in combat). Moreover, responses in driving may be needed within fractions of a second, whereas the time required in aviation is typically measured in minutes—or even hours for long-range anti-congestion planning.

The total-system concept applies acutely to automated highways, as many problems must be integrated. The entire environment may be laced with sensors and instrumentation that can interact with individual subsystems—signaling them appropriately in real time. This will create many complex interconnected system problems requiring scalable solutions and that avoid excessive energy consumption.

As a consequence, the legal, liability, privacy, and other issues noted here for automated vehicles are likely to be even more complicated when applied to distributed control of autonomous and semi-automated vehicles on automated highways, even if they

Any attempt to develop autonomous systems must have intensive monitoring to ensure that the systems are operating properly.

are not co-mingling with conventional manual vehicles.

The Internet of Things

The Internet of Things (IoT) has the potential that almost everything imaginable might have some sort of online presence. Therefore, the IoT must be considered in the context of the preceding discussion—particularly with respect to those Things that are actually directly accessible on the Internet. Devices may be completely autonomous or operated totally under human control (but with remote monitoring), or again in between. Some will be remotely controllable, or otherwise accessible over the Internet. (This seems to be an open invitation for undesirable manipulation and invasive privacy violations.) However, more sensibly, many such Things are likely to be hidden behind a firewall, but still potentially accessible remotely (for example, via SSH). If the cheapest solutions are sought, there might be no firewall, and each Thing would require its own protective environment. Otherwise, the existing “Dark Net” on the Internet (generally unsearchable) will grow significantly to accommodate all of the Things that might hide behind supposedly secure firewalls. This could also result in the development of firewalls that are penetrable for government surveillance in certain countries, which could open up misuse by others as well. Given the vulnerabilities in today’s firewalls, desktops, and mobile devices, significantly better security will be required, even in tiny systems in small and seemingly inconsequential Things, but especially in firewalls and internal routers. Indeed, perhaps those seemingly inconsequential ones will provide access to the others—because of the likelihood of unrestricted total access within the locally networked Things behind the firewall.

The privacy issues are somewhat murky. For example, a federal judge for the Eastern District of Virginia has ruled that the user of any computer that connects to the Internet should not have an expectation of privacy, because computer security is ineffectual at stopping hackers. The June 23, 2016, ruling came in one of the many cases resulting from the FBI’s infiltration of PlayPen, a hidden service on

e “‘Gene,’ Tesla Model X rolls over after crashing into concrete divider, driver claims Autopilot was activated,” (July 6, 2016); <http://bit.ly/2bcr9KM> and AFP news item, “Tesla crash: Model X flips while in autopilot mode, driver says”; <http://bit.ly/2aMWmTO>

the Tor network that acted as a hub for child exploitation, and the subsequent prosecution of hundreds of individuals. (The judge's ruling seems in conflict with other rulings, and could well be appealed.) To identify suspects, the FBI took control of PlayPen for two weeks and used a network investigative program that runs on visitors' computers to identify their Internet addresses.^f

We might suspect today that the IoT is largely a corporate marketing opportunity where each company seeks to have a valid approach. However, it also appears that there is no *there* there—at least not yet, and that you might expect a lot of snake-oil salesmen.

Clouds

Cloud computing and cloud storage make enormous sense in many operational environments. To most users, these resources would seem to be autonomous, with human inputs and computer-generated outputs. However, they raise many issues relating to the trustworthiness of the clouds and networks, and who or what needs to be trusted. Examples of what might be particularly thorny here are encryption and key management, exceptional access for law enforcement, and maintenance and remediation when something goes fundamentally wrong (for example, outages or compromise). In the last of these concerns, where might you (or the cloud provider) find suitably experienced system administrators rapidly in cases of crises? Most of these issues may be completely out of the control of user communities.

Surveillance

The Keys Under Doormats report¹ makes the technical argument that dumbing down security to simplify the job of law enforcement is a very bad idea: for example, it would open up huge potential vulnerabilities for exploitation, and would undoubtedly drive domestic system providers and their domestic customers in many different nations to find other sources of secure systems. Several former high U.S. government officials have supported the conclusions of that report.

^f See <http://nyti.ms/2aHGExM>

We need computer-related systems with significantly greater trustworthiness.

Any attempt to develop autonomous systems must have intensive monitoring to ensure that the systems are operating properly. As a consequence, the challenges of developing monitoring that is not only trustworthy, nonsubvertible, and privacy-aware, but also forensics-worthy will have to be addressed. The risks of dumbed-down security being compromised by other than the supposedly privileged surveillers (including privileged insiders) will add to the reality that automobiles and other devices could be remotely compromised. As a result, demands for surveillable autonomous systems that cannot be compromised by others seems to be an oxymoronic idea, or perhaps recursively difficult—as it would require much more secure systems in the first place!

Remediation

Some of these problems (except for “noncompromisable surveillance”) can be addressed by having hardware that enforces fine-grained access controls along with hardware-ensured virtualization, and scalable compartmentalization of software that may be less trustworthy. For example, mobile devices and laptops should not allow applications to have unfettered access to contact lists and other apps without explicit permission. Hardware that helps enforce strict security properties would be very beneficial. Similarly, the Internet of Things will require seriously secure firewalls and local networks, with subsystems scaled in cost and complexity according to the criticality of the Things. Advances in formal methods can also play a role in increasing the assurance of trustworthiness of the hardware and software of such systems, including formally based testing and evaluation. See the CHERI system architec-

ture⁸ for an example of what might be possible with clean-slate hardware design, with operating system and compiler variants that know how to take advantage of the hardware.

Conclusion

Purveyors of modern computer-based systems wish to make some great leaps forward with automation and real-time automated assistance—in some cases bringing beta-test versions into use prematurely. We need computer-related systems with significantly greater trustworthiness than we have today, especially for use in critical systems. We also need much more stringent total-system requirements and overall system architectures, better development engineering, total-system testing and evaluation, and—perhaps above all—proactive awareness and understanding of the risks for would-be customers. If we are routinely going to have fully automated systems—or even partially automated systems that may require instantaneous human interventions in certain cases—we must have much more advanced system research and development, as well as education relating to potential risks and how to deal with them when they arise. The old adage “Let the buyer beware” (Caveat Emptor) must be extended to users as well. **C**

References

1. Abelson, H. et al. *Journal of Cybersecurity* 1, 1 (Nov. 2015), Oxford University Press; <http://bit.ly/2bcj1dr>
2. Beurdouche, B. et al. A messy state of the union: Taming the composite state machines of TLS. In *Proceedings of the 36th IEEE Symposium on Security and Privacy*, San Jose, CA, May 18–20, 2015; <http://bit.ly/2bndXGz>
3. Casner, S.M., Hutchinson, E.L., and Norman, D. The challenges of partially automated driving: Car automation promises to free our hands from the steering wheel, but might demand more from our minds. *Commun. ACM* 59, 5 (May 2016).
4. Neumann, P.G. Automated car woes—Whoa there! *ACM Ubiquity*, July 2016; <http://bit.ly/2aYKDoT>
5. Neumann, P.G. *Computer-Related Risks*. Addison-Wesley and ACM Press, 1995.
6. Norman, D.A. The human side of automation. *Road Vehicle Automation 2*, Springer, 2015.
7. Quain, J.B. The autonomous car vs. human nature, a driver behind the wheel may not be ready to take it. *The New York Times* (July 8, 2016).
8. Watson, R.N.M. et al. CHERI: A hybrid capability-system architecture for scalable software compartmentalization. In *Proceedings of the 37th IEEE Symposium on Security and Privacy* (San Jose, CA, May 18–20, 2015).

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

Copyright held by author.

Viewpoint

Universities and Computer Science in the European Crisis of Refugees

Considering the role of universities in promoting tolerance as well as education.

THE CURRENT CRISIS of refugees has divided European countries and societies into those who welcome refugees and those who oppose taking them. In this Viewpoint, we reflect on the role of universities and of computer science in such situations. As a case study, we describe an activity taken at the TU Wien (Vienna University of Technology): when the crisis of refugees culminated in summer 2015, a group of professors and students of the Faculty of Informatics initiated computer courses for unaccompanied

young refugees. This project allowed the refugees to gain computer-related knowledge, and, equally important, to make contacts with local students. Another major goal of the project was to give a clear message that refugees are welcome. Considering the attention received in the media and in the public, we are convinced universities have significant influence and reputation in society that puts them in a favorable position to promote tolerance and to encourage other institutions to act similarly.

When this Viewpoint was written

in early 2016, the crisis of refugees due to the wars in the Middle East had become one of the main European problems. The regional governments in the neighboring countries and in Europe are overwhelmed and have not yet been able to find good, or at least feasible, solutions to this problem. Furthermore, this crisis has divided the European leaders and countries. On one hand, several states have been opening the borders for the refugees and many citizens have shown solidarity. On the other hand, a serious increase of intolerance toward the refu-



Students at an end-of-year celebration at the Faculty of Informatics of TU Wien, Austria, in June 2016.

gees in many European countries has to be reported. Furthermore, right-wing political parties in different regions have gained substantial popularity and have been (mis)using the current situation to reach their goals in elections.

In our opinion the universities have not been active enough (at least in the public) regarding the current refugee crisis. Here, we would like to inspire a discussion regarding the role of universities in such situations. We will briefly address the following questions: Should the universities take a more active role in such cases like the crisis of refugees by taking a clear position that promotes tolerance, and should the universities be more active to find solutions for such problems? What is the role of computer science regarding these issues? Further, we will express our opinion regarding these questions and describe an activity taken at the TU Wien (Vienna University of Technology) to support young refugees (between 14–18 years old) and to signal to them that they are welcome.

The Crisis of Refugees

Due to the current wars in Syria and Iraq the number of refugees has been increasing tremendously. According to the UNHCR the total number of Syrian refugees exceeded four million in July 2015 (UNHCR, Press Releases, July 9, 2015: <http://bit.ly/1G8LY0k>). Although most refugees from Syria have been fleeing to neighboring countries, hundreds of thousands of refugees (including those from Afghanistan, Iraq, and African countries) have fled this year to European states. Sad to say, thousands of refugees have lost their life on their way to Europe using boats or inside trucks. These tragic events and the continuous flow of refugees have sensitized Europe. However, as mentioned, Europe is deeply divided regarding the refugee crises. Several European countries have opened their borders for refugees and have been welcoming them, but unfortunately several other countries are not welcoming the refugees and oppose taking them. This division could also be observed among the inhabitants of Europe. Whereas an enormous number of

citizens (in the best sense) in several countries have shown solidarity and have been volunteering to help the refugees, many others have been spreading intolerance. Several right-wing political parties that made the issue of refugees the main topic in elections have been increasing their votes. The situation in Europe is expected to get tenser in the following months as the number of refugees is expected to increase further.

The Role of Universities in this Crisis

Universities are focused on research and teaching, on knowledge creation and distribution. This leaves them with an important role in the society. But this can not only be seen in “utility” or “usefulness” terms, in our view a university should also reflect on the society, its developments and problems, it should try to identify solutions and, finally, take position even in “rough times.” Currently in Europe an intensive political debate regarding the refugee crisis is taking place and clear positioning of universities regarding refugees is crucial, because their influence and reputation is significant. The necessity of engagement of universities in the refugee crisis has also been mentioned recently by Austrian politicians following the lecture of Dr. Jeffrey D. Sachs “Special Lecture: What Is the Role of a Modern University in the Fight Against Inequality?” in European Forum Alpbach 2015 (<http://bit.ly/2aPfnof>).

Regarding the refugee crisis we think university computer science

University computer science departments could take several possible actions to position themselves as supporters of human rights.

departments could take several possible actions to position themselves as supporters of human rights and also influence the opinions of other people regarding this sensitive issue. In our opinion, these actions could include, among others, the organization of specific courses for refugees and research projects dealing with specific problems regarding the refugees. The computer science departments in particular are in a good situation, because learning the basic IT skills and computer knowledge is essential in our society, rich of artifacts and applications. Therefore, several activities that fall under the umbrella of computing for the social good concept^{1,2} could be taken.

For example, the computer science departments could organize short-term computer courses for young refugees, offer online computer courses specialized for them (to enable them to integrate quickly into society and increase their chance to find jobs), consider student projects for developing specific apps for the refugees, and other initiatives. With this respect one could think of a multitude of useful applications in this field and even of specific social innovation activities, linked to IT. Another innovative issue, as in our case, could be to organize traineeships for refugees in start-up companies.

Case Study at the TU Wien

Following the controversial discussion regarding the refugees in Austria at the time when the number of refugees was increasing, a group of professors at the Faculty of Informatics at TU Wien took an initiative to organize summer courses for unaccompanied young refugees (age between 14 and 18). The student body of our faculty joined the action immediately and the project was named Welcome.TU.Code. The intention was to give a clear message that the refugees are welcome in Austria.

After the preparatory work in June/July 2015, the computer courses began in the end of July and took four weeks. Around 60 young refugees from different countries (including Syria, Afghanistan, Somalia, and Iraq) were among the participants of these courses. One of the challeng-

ing questions was which curriculum to teach in these courses. The initial curriculum included computer programming for kids, but due to the heterogeneity of the refugees and their different needs the curriculum was adapted and changed to better match the participants' background and interests. The complete curriculum for these four weeks included game programming for kids, basics of operating systems, Internet and basics of security and privacy, and office applications. Furthermore, for few advanced participants programming in Java and Python was taught individually. This was only possible because tutors—computer science students of the TU Wien—had knowledge from these different areas and were flexible in their approach and adapted the curriculum for different groups. The team of our 20 tutors/students had different nationalities and different language skills. It is important to note that none of the participating people was paid. At the conclusion of the courses, the young refugees received certificates for visiting the course and their feedback clearly suggested we should continue this project.

Impact of This Project

This was one of the first actions among universities in Austria that dealt directly with the refugees. Therefore, it attracted the attention of Austrian newspapers, state television, and radio. These media broadcast several reports about this activity during the time when the number of refugees coming to Austria was drastically increasing and the number of locals opposing the acceptance of refugees was increasing as well. Our action and similar actions at other Austrian universities gave a clear sign that intolerance toward refugees is not acceptable and that refugees are welcome. In addition, many people, mostly alumni, wrote us to show their readiness to help in this action.

We believe our activity in this field and some similar initial actions of other universities encouraged more departments and institutions to do similar projects, because the number of actions from different universities in support of refugees has been increasing continuously. Regarding

The project promoted the idea that refugees of different cultures and religions are welcome in Austria.

the direct impact of these courses on the participants, the young refugees showed an eager interest and most of them visited the courses regularly. They gained new knowledge, and it was also very important for them to make new contacts with local students. Furthermore, the project also had some other side effects. For example, one of the refugees was admitted to study computer science, some refugees got laptops, which were donated by several departments of TU Wien and other people, and some opportunities for internships arose.

In general, the action reached its goal of showing a clear position of our university regarding the issue of refugees. The project promoted the idea that refugees of different cultures and religions are welcome in Austria.

Lessons Learned

We observed that even such short-term action received attention of public media and many other people. It was very important to be flexible regarding the curriculum and we have learned that continuous adaptation is needed in order to better meet the needs and interests of refugees. It was also very important to have tutors of different nationalities and also have some tutors who speak the languages of refugees who had difficulties communicating in German or English. Although this action took only four weeks it needed good organization and the commitment of organizers and tutors. We also noted that such actions can inspire and motivate other people and organizations to do similar projects.

Finally, taking a stance in this case not only positioned the university in the public discussion but also influ-

enced this debate. We are convinced that social commitment and establishing a clear position in a sensitive political/societal discussion does not conflict at all with the “usual” goals of a university to strive for high reputation in research and education.

Future Projects

At the end of this project we had a meeting with the NGOs that take care of the teenagers who visited our courses. Their feedback was very positive and they strongly supported the idea of continuing with such courses at the TU Wien. In the Winter semester 2015 we continued with the new courses at three difficulty levels. Approximately 60 refugees had the opportunity to visit these courses and the TU-Wien students (about 25) who taught the refugees had course credits assigned. We plan to offer such courses on a regular basis, are also considering offering specific online courses for refugees, and are organizing traineeships for refugees in start-ups.

We think the issue of refugees is an important aspect to be considered when speaking about the idea of computing for the social good. Therefore, we hope this Viewpoint will motivate some discussion regarding the role of computer science departments and universities in such situations and animate universities in other countries to take similar actions that position universities as supporters of human rights. 

References

1. Goldweber, M. et al. Enhancing the social issues components in our computing curriculum: Computing for the social good. *ACM Inroads* 2, 1 (2011), 64–82.
2. Kaczmarczyk, L. *Computers and Society: Computing for Good*. CRC Press, 2011.

Kathrin Conrad (kaddi@fsinf.at) is a student representative at TU Wien.

Nysret Musliu (musliu@dbai.tuwien.ac.at) is a professor at TU Wien, Institute of Information Systems.

Reinhard Pichler (pichler@dbai.tuwien.ac.at) is a professor at TU Wien, Institute of Information Systems.

Hannes Werthner (hannes.werthner@ec.tuwien.ac.at) is a professor at TU Wien, Institute for Software Technology and Interactive Systems.

Copyright held by authors.



Watch the authors discuss their work in this exclusive *Communications* video.
<http://cacm.acm.org/videos/universities-and-computer-science-in-the-european-crisis-of-refugees>

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

Taking advantage of idleness to reduce dropped frames and memory consumption.

BY ULAN DEGENBAEV, JOCHEN EISINGER,
MANFRED ERNST, ROSS MCILROY, AND HANNES PAYER

Idle-Time Garbage-Collection Scheduling

GOOGLE'S CHROME WEB browser strives to deliver a smooth user experience. An animation will update the screen at 60FPS (frames per second), giving Chrome approximately 16.6 milliseconds to perform the update. Within these 16.6ms, all input events have to be processed, all animations have to be performed, and finally the frame has to be rendered. A missed deadline will result in dropped frames. These are visible to the user and degrade the user experience. Such sporadic animation artifacts are referred to here as *jank*.³

JavaScript, the lingua franca of the Web, is typically used to animate Web pages. It is a garbage-collected programming language where the application developer does not have to worry about memory management. The garbage collector interrupts the

application to pass over the memory allocated by the application, determine live memory, free dead memory, and compact memory by moving objects closer together. While some of these garbage-collection phases can be performed in parallel or concurrently to the application, others cannot, and as a result they may cause application pauses at unpredictable times. Such pauses may result in user-visible jank or dropped frames; therefore, we go to great lengths to avoid such pauses when animating Web pages in Chrome.

This article describes an approach implemented in the JavaScript engine V8 used by Chrome to schedule garbage-collection pauses during times when Chrome is idle.¹ This approach can reduce user-visible jank on real-world Web pages and results in fewer dropped frames.

Garbage Collection in V8

Garbage-collector implementations typically optimize for the *weak generational hypothesis*,⁶ which states that most of the allocated objects in applications die young. If the hypothesis holds, garbage collection is efficient and pause times are low. If it does not hold, pause times may lengthen.

V8 uses a generational garbage collector, with the JavaScript heap split into a small young generation for newly allocated objects and a large old generation for long-living objects. Since most objects typically die young, this generational strategy enables the garbage collector to perform regular, short garbage collections in the small young generation, without having to trace objects in the large old generation.

The young generation uses a semi-space allocation strategy, where new objects are initially allocated in the young generation's active semi-space. Once a semi-space becomes full, a scavenge operation will trace through the live objects and move them to the other semi-space.

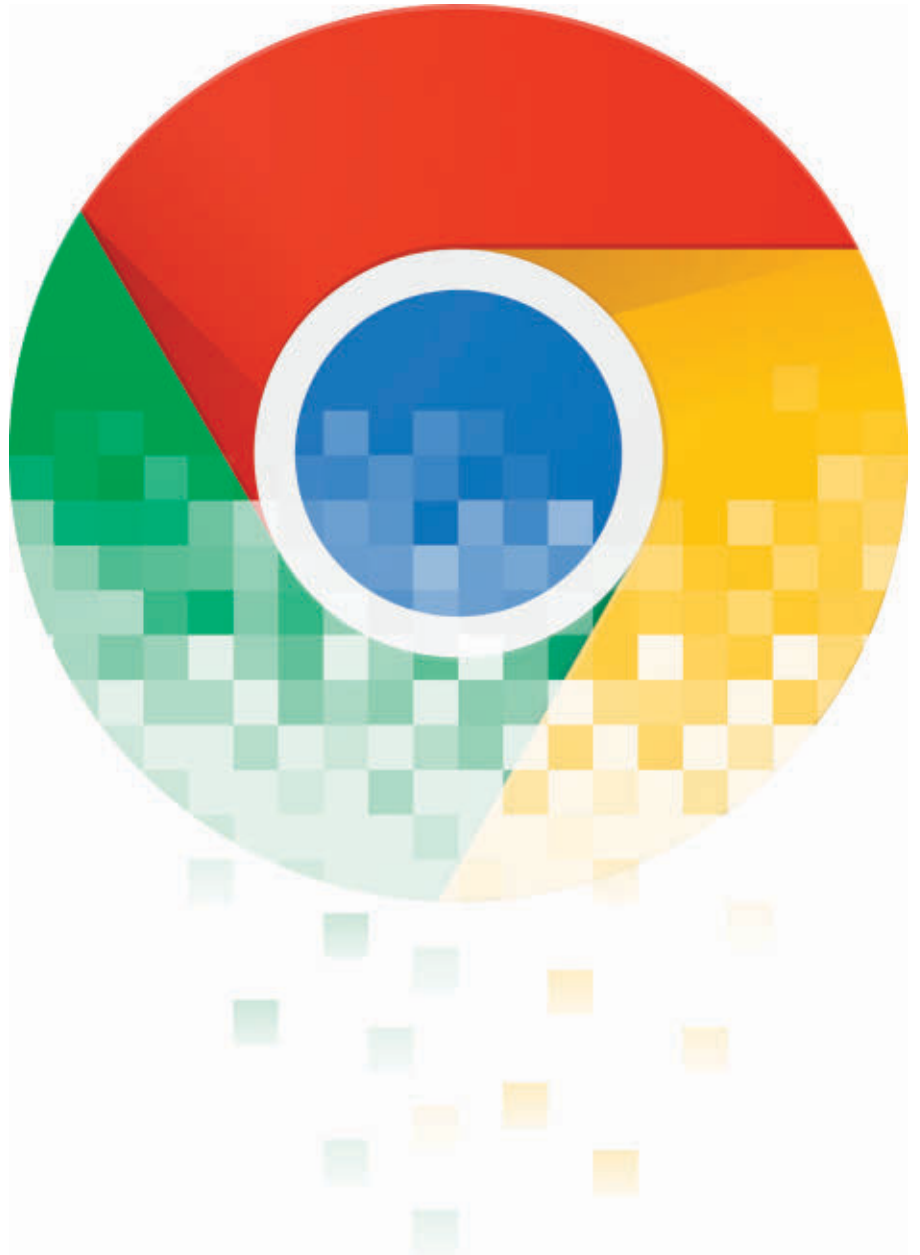
Such a semi-space scavenge is a *minor garbage collection*. Objects that

have already been moved in the young generation are promoted to the old generation. After the live objects have been moved, the new semi-space becomes active and any remaining dead objects in the old semi-space are discarded without iterating over them.

The duration of a minor garbage collection therefore depends on the size of the live objects in the young generation. A minor garbage collection is typically fast, taking no longer than one millisecond when most of the objects become unreachable in the young generation. If most objects survive, however, the duration of a minor garbage collection may be significantly longer.

A major garbage collection of the whole heap is performed when the size of live objects in the old generation grows beyond a heuristically derived memory limit of allocated objects. The old generation uses a mark-and-sweep collector with compaction. Marking work depends on the number of live objects that have to be marked, with marking of the whole heap potentially taking more than 100ms for large Web pages with many live objects.

To avoid such long pauses, V8 marks live objects incrementally in many small steps, pausing only the main thread during these marking steps. When incremental marking is completed the main thread is paused to finalize this major collection. First, free memory is made available for the application again by sweeping the whole old-generation memory, which is performed concurrently by dedicated sweeper threads. Afterward, the young generation is evacuated, since we mark through the young generation and have liveness information. Then memory compaction is performed to reduce memory fragmentation in old-generation pages. Young-generation evacuation and old-generation compaction are performed by parallel compaction threads. After that, the object pointers to moved objects in the remembered sets are updated in parallel. All these finalization tasks occur in



a single atomic pause that can easily take several milliseconds.

The Two Deadly Sins of Garbage Collection

The garbage-collection phases outlined here can occur at unpredictable times, potentially leading to application pauses that impact the user experience. Hence, developers often become creative in attempting to sidestep these interruptions if the performance of their application suffers. Here, we look at two controversial approaches that are often proposed and outline their potential problems. These are the two deadly sins of garbage collection.

Sin One: Turning off the garbage collector. Developers often ask for an API to turn off the garbage collector during a time-critical application phase where a garbage-collection pause could result in missed frames. Using such an API, however, complicates application logic and leads to it becoming more difficult to maintain. Forgetting to turn on the garbage collector on a single branch in the pro-

gram may result in out-of-memory errors. Furthermore, this also complicates the garbage-collector implementation, since it has to support a never-fail allocation mode and must tailor its heuristics to take into account these non-garbage-collecting time periods.

Sin Two: Explicit garbage-collection invocation. JavaScript does not have a Java-style `System.gc()` API, but some developers would like to have that. Their motivation is proactively to invoke garbage collection during a non-time-critical phase in order to avoid it later when timing is critical. The application, however, has no idea how long such a garbage collection will take and therefore may by itself introduce junk. Moreover, garbage-collection heuristics may get confused if developers invoke the garbage collector at arbitrary points in time.

Given the potential for developers to trigger unexpected side effects with these approaches, they should not interfere with garbage collection. Instead, the runtime system should endeavor to avoid the need for such

tricks by providing high-performance application throughput and low-latency pauses during mainline application execution, while scheduling longer-running work during periods of idleness such that it does not impact application performance.

Idle-Task Scheduling

To schedule long-running garbage collection tasks while Chrome is idle, V8 uses Chrome's task scheduler. This scheduler dynamically reprioritizes tasks based on signals it receives from a variety of other components of Chrome and various heuristics aimed at estimating user intent. For example, if the user touches the screen, the scheduler will prioritize screen rendering and input tasks for a period of 100ms to ensure the user interface remains responsive while the user interacts with the Web page.

The scheduler's combined knowledge of task queue occupancy, as well as signals it receives from other components of Chrome, enables it to estimate when Chrome is idle and how long it is likely to remain so. This knowledge is used to schedule low-priority tasks, hereafter called *idle tasks*, which are run only when there is nothing more important to do.

To ensure these idle tasks don't cause jank, they are eligible to run only in the time periods between the current frame having been drawn to screen and the time when the next frame is expected to start being drawn. For example, during active animations or scrolling (see Figure 1), the scheduler uses signals from Chrome's compositor subsystem to estimate when work has been completed for the current frame and what the estimated start time for the next frame is, based on the expected interframe interval (for example, if rendering at 60FPS, the interframe interval is 16.6ms). If no active updates are being made to the screen, the scheduler will initiate a longer idle period, which lasts until the time of the next pending delayed task, with a cap of 50ms to ensure Chrome remains responsive to unexpected user input.

To ensure idle tasks do not overrun an idle period, the scheduler passes a deadline to the idle task when it starts, specifying the end of the current idle

Figure 1. Idle period example.

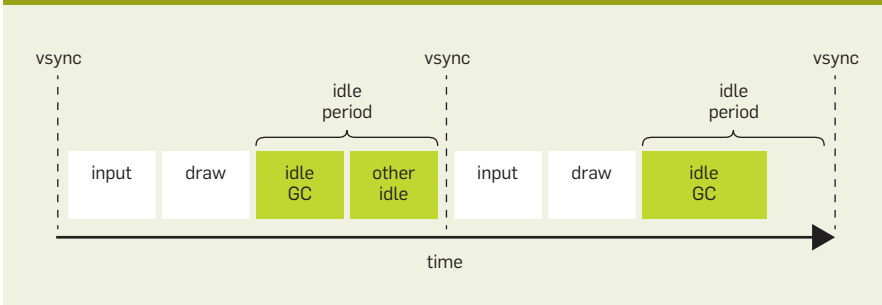
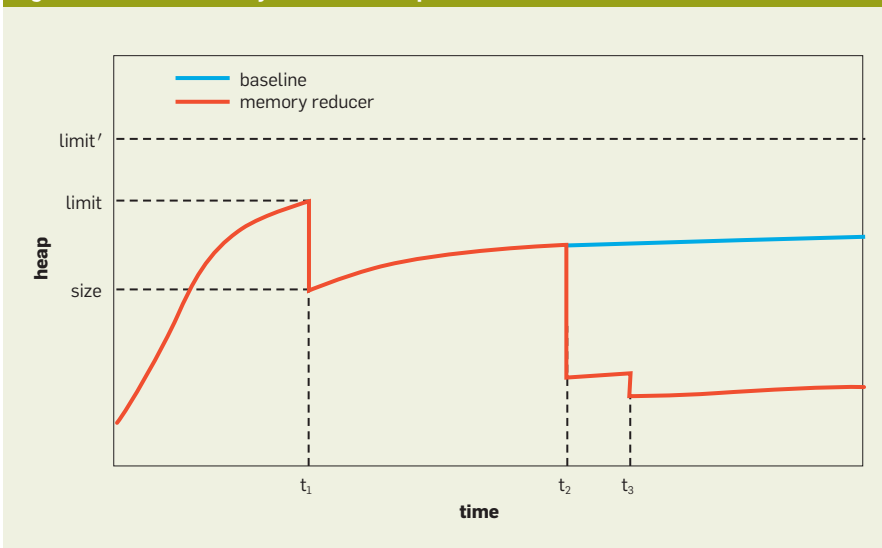


Figure 2. Effect of memory reducer on heap size.



period. Idle tasks are expected to finish before this deadline, either by adapting the amount of work they do to fit within this deadline or, if they cannot complete any useful work within the deadline, by reposting themselves to be executed during a future idle period. As long as idle tasks finish before the deadline, they do not cause jank in Web page rendering.

Idle-Time Garbage-Collection Scheduling in V8

Chrome's task scheduler allows V8 to reduce both jank and memory usage by scheduling garbage-collection work as idle tasks. To do so, however, the garbage collector needs to estimate both when to trigger idle-time garbage-collection tasks and how long those tasks are expected to take. This allows the garbage collector to make the best use of the available idle time without going past an idle-tasks deadline. This section describes implementation details of idle-time scheduling for minor and major garbage collections.

Minor garbage-collection idle-time scheduling. Minor garbage collection cannot be divided into smaller work chunks and must be performed either completely or not at all. Performing minor garbage collections during idle time can reduce jank; however, being too proactive in scheduling a minor garbage collection can result in promotion of objects that could otherwise die in a subsequent non-idle minor garbage collection. This could increase the old-generation size and the latency of future major garbage collections. Thus, the heuristic for scheduling minor garbage collections during idle time should balance between starting a garbage collection early enough that the young-generation size is small enough to be collectable during regular idle time, and deferring it long enough to avoid false promotion of objects.

Whenever Chrome's task scheduler schedules a minor garbage-collection task during idle time, V8 estimates if the time to perform the minor garbage collection will fit within the idle-task deadline. The time estimate is computed using the average garbage-collection speed and the current size of the young generation. It also estimates the young-generation growth rate and

Chrome's task scheduler allows V8 to reduce both jank and memory usage by scheduling garbage-collection work as idle tasks.

performs an idle-time minor garbage collection only if the estimate is that at the next idle period the size of the young generation is expected to exceed the size that could be collected within an average idle period.

Major garbage-collection idle-time scheduling. A major garbage collection consists of three parts: initiation of incremental marking, several incremental marking steps, and finalization. Incremental marking starts when the size of the heap reaches a certain limit, configured by a heap-growing strategy. This limit is set at the end of the previous major garbage collection, based on the heap-growing factor f and the total size of live objects in the old generation: $\text{limit} = f \cdot \text{size}$.

As soon as an incremental major garbage collection is started, V8 posts an idle task to Chrome's task scheduler, which will perform incremental marking steps. These steps can be linearly scaled by the number of bytes that should be marked. Based on the average measured marking speed, the idle task tries to fit as much marking work as possible into the given idle time. The idle task keeps reposting itself until all live objects are marked. V8 then posts an idle task for finalizing the major garbage collection. Since finalization is an atomic operation, it is performed only if it is estimated to fit within the allotted idle time of the task; otherwise, V8 reposts that task to be run at a future idle time with a longer deadline.

Memory reducer. Scheduling a major garbage collection based on the allocation limit works well when the Web page shows a steady allocation rate. If the Web page becomes inactive and stops allocating just before hitting the allocation limit, however, there will be no major garbage collection for the whole period while the page is inactive. Interestingly, this is an execution pattern that can be observed in the wild. Many Web pages exhibit a high allocation rate during page load as they initialize their internal data structures. Shortly after loading (a few seconds or minutes), the Web page often becomes inactive, resulting in a decreased allocation rate and decreased execution of JavaScript code. Thus, the Web page will retain more memory than it actually needs while it is inactive.

A controller, called *memory reducer*, tries to detect when the Web page becomes inactive and proactively schedules a major garbage collection even if the allocation limit is not reached. Figure 2 shows an example of major garbage-collection scheduling.

The first garbage collection happens at time t_1 because the allocation limit is reached. V8 sets the next allocation limit based on the heap size. The subsequent garbage collections at times t_2 and t_3 are triggered by the memory reducer before limit is reached. The dotted line shows what the heap size would be without the memory reducer.

Since this can increase latency, Google developed heuristics that rely not only on the idle time provided by Chrome's task scheduler, but also on whether the Web page is now inactive. The memory reducer uses the JavaScript

invocation and allocation rates as signals for whether the Web page is active or not. When the rate drops below a predefined threshold, the Web page is considered to be inactive and major garbage collection is performed in idle time.

Silky Smooth Performance

Our aim with this work was to improve the quality of user experience for animation-based applications by reducing jank caused by garbage collection. The quality of the user experience for animation-based applications depends not only on the average frame rate, but also on its regularity. A variety of metrics have been proposed in the past to quantify the phenomenon of jank—for example, measuring how often the frame rate has changed, calculating the variance of the frame durations, or simply using

the largest frame duration. Although these metrics provide useful information, they all fail to measure certain types of irregularities. Metrics that are based on the distribution of frame durations, such as variance or largest frame duration, cannot take the temporal order of frames into account. For example, they cannot distinguish between the case where two dropped frames are close together and the case where they are further apart. The former case is arguably worse.

We propose a new metric to overcome these limitations. It is based on the discrepancy of the sequence of frame durations. Discrepancy is traditionally used to measure the quality of samples for Monte Carlo integration. It quantifies how much a sequence of numbers deviates from a uniformly distributed sequence. Intuitively, it measures the duration of the worst jank. If only a single frame is dropped, the discrepancy metric is equal to the size of the gap between the drawn frames. If multiple frames are dropped in a row—with some good frames in between—the discrepancy will report the duration of the entire region of bad performance, adjusted by the good frames.

Discrepancy is a great metric for quantifying the worst-case performance of animated content. Given the timestamps when frames were drawn, the discrepancy can be computed in $O(N)$ time using a variant of Kadane's algorithm for the maximum subarray problem.

The online Web Graphics Library (WebGL) benchmark OortOnline (<http://oortonline.gl/#run>) demonstrates jank improvements of idle-time garbage-collection scheduling. Figure 3 shows these improvements: frame-time discrepancy, frame time, number of frames missed because of garbage collection, and total garbage-collection time compared with the baseline on the oortonline.gl benchmark.

Frame-time discrepancy is reduced on average from 212ms to 138ms. The average frame-time improvement is from 17.92ms to 17.6ms. We observed that 85% of garbage-collection work was scheduled during idle time, which significantly reduced the amount of garbage-collection work performed during time-critical phases. Idle-time garbage-collection scheduling increased the

Figure 3. Improvements to the OortOnline.gl benchmark.

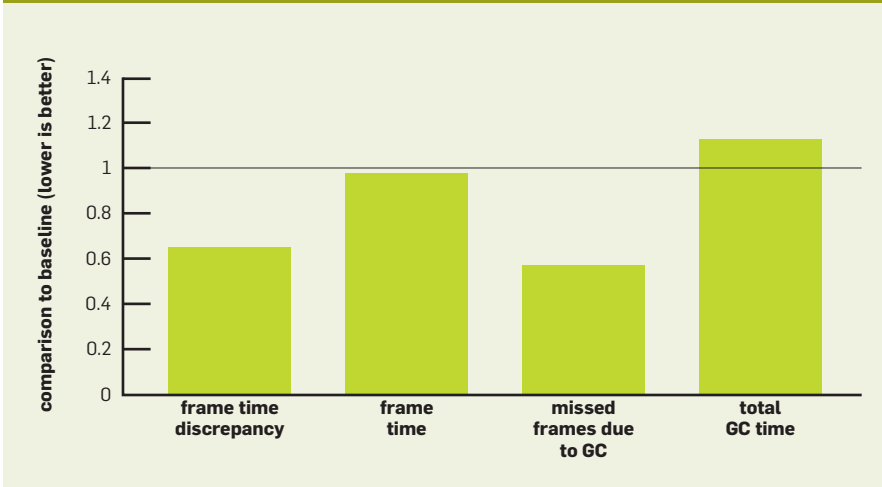
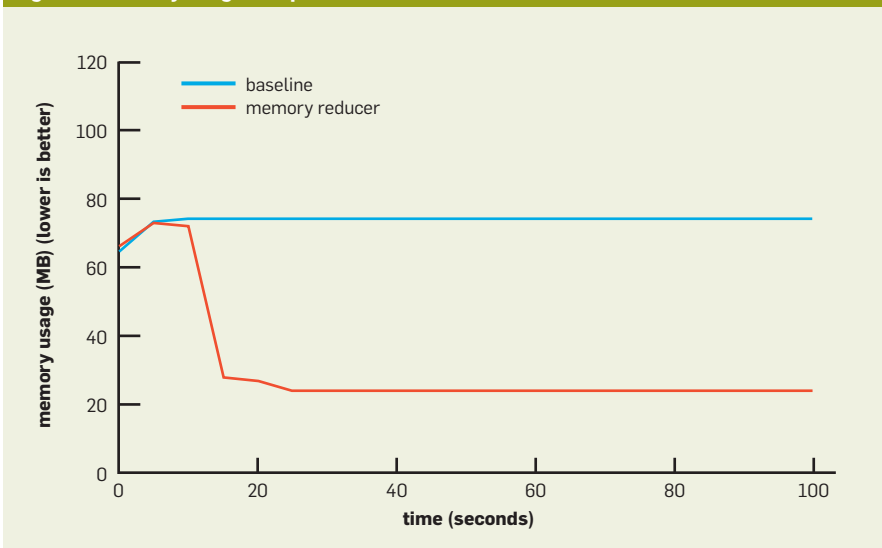


Figure 4. Memory usage comparison.



total garbage-collection time by 13% to 780ms. This is because scheduling garbage collection proactively and making faster incremental marking progress with idle tasks resulted in more garbage collections.

Idle-time garbage collection also improves regular Web browsing. While scrolling popular Web pages such as Facebook and Twitter, we observed that about 70% of the total garbage-collection work is performed during idle time.

The memory reducer kicks in when Web pages become inactive. Figure 4 shows an example run of Chrome with and without the memory reducer on the Google Web Search page. In the first few seconds both versions use the same amount of memory as the Web page loads and allocation rate is high. After a while the Web page becomes inactive since the page has loaded and there is no user interaction. Once the memory reducer detects that the page is inactive, it starts a major garbage collection. At that point the graphs for the baseline and the memory reducer diverge. After the Web page becomes inactive, the memory usage of Chrome with the memory reducer decreases to 34% of the baseline.

A detailed description of how to run the experiments presented here to reproduce these results can be found in the 2016 Programming Language Design and Implementation (PLDI) artifact evaluation document.²

Other Idle-Time Garbage-Collected Systems

A comprehensive overview of garbage collectors taking advantage of idle times is available in a previous article.⁴ The authors classify different approaches in three categories: slack-based systems where the garbage collector is run when no other task in the system is active; periodic systems where the garbage collector is run at predefined time intervals for a given duration; and hybrid systems taking advantage of both ideas. The authors found that, on average, hybrid systems provide the best performance, but some applications favor a slack-based or periodic system.

Our approach of idle-time garbage-collection scheduling is different. Its main contribution is that it profiles the application and garbage-collection components to predict how long garbage-collection operations will take and when

the next minor or major collection will occur as a result of application allocation throughput. That information allows efficient scheduling of garbage-collection operations during idle times to reduce jank while providing high throughput.

Concurrent, Parallel, Incremental Garbage Collection

An orthogonal approach to avoid garbage-collection pauses while executing an application is achieved by making garbage-collection operations concurrent, parallel, or incremental. Making the marking phase or the compaction phase concurrent or incremental typically requires read or write barriers to ensure a consistent heap state. Application throughput may degrade because of expensive barrier overhead and code complexity of the virtual machine.

Idle-time garbage-collection scheduling can be combined with concurrent, parallel, and incremental garbage-collection implementations. For example, V8 implements incremental marking and concurrent sweeping, which may also be performed during idle time to ensure fast progress. Most importantly, costly memory-compaction phases such as young-generation evacuation or old-generation compaction can be efficiently hidden during idle times without introducing costly read or write barrier overheads.

For a best-effort system, where hard realtime deadlines do not have to be met, idle-time garbage-collection scheduling may be a simple approach to provide both high throughput and low jank.

Beyond Garbage Collection and Conclusion

Idle-time garbage-collection scheduling focuses on the user's expectation that a system that renders at 60 frames per second appears silky smooth. As such, our definition of idleness is tightly coupled to on-screen rendering signals. Other applications can also benefit from idle-time garbage-collection scheduling when an appropriate definition of idle time is applied. For example, a node.js-based server that is built on V8 could forward idle-time periods to the V8 garbage collector while it waits for a network connection.

The use of idle time is not limited

to garbage collection. It has been exposed to the Web platform in the form of the requestIdleCallback API,⁵ enabling Web pages to schedule their own callbacks to be run during idle time. As future work, other management tasks of the JavaScript engine could be executed during idle time (for example, compiling code with the optimizing just-in-time compiler that would otherwise be performed during JavaScript execution). 

Related articles on queue.acm.org

Real-time Garbage Collection

David F. Bacon

<http://queue.acm.org/detail.cfm?id=1217268>

A Conversation with David Anderson

<http://queue.acm.org/detail.cfm?id=1080872>

Network Virtualization: Breaking the Performance Barrier

Scott Rixner

<http://queue.acm.org/detail.cfm?id=1348592>

References

1. Degenbaev, U., Eisinger, J., Ernst, M., McIlroy, R., Payer, H. Idle time garbage collection scheduling. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*, (2016).
2. Degenbaev, U., Eisinger, J., Ernst, M., McIlroy, R., Payer, H. PLDI'16 Artifact: Idle time garbage collection scheduling (Santa Barbara, CA, June 13-17, 2016) 570-583. ACM, 978-1-4503-4261-2/16/06; <https://goo.gl/AxvigS>.
3. Google Inc. The RAIL performance model; <http://developers.google.com/Web/tools/chrome-devtools/profile/evaluate-performance/rail>.
4. Kalibera, T., Pizlo, F., Hosking, A. L., Vitek, J. Scheduling real-time garbage collection on uniprocessors. *ACM Trans. Computer Systems* 29, 3 (2011), 8:1-8:29.
5. McIlroy, R. Cooperative scheduling of background tasks. W3C editor's draft, (2016); <https://w3c.github.io/requestidlecallback/>.
6. Ungar, D. 1984. Generation scavenging: a nondisruptive high-performance storage reclamation algorithm. In *Proceedings of the 1st ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments (SDE 1)*.

Ulan Degenbaev is a software engineer at Google, working on the garbage collector of the V8 JavaScript engine.

Jochen Eisinger is a software engineer at Google, working on the V8 JavaScript engine and Chrome security. Prior to that, he worked on various other parts of Chrome.

Manfred Ernst is a software engineer at Google, where he works on virtual reality. Prior to that, he integrated a GPU rasterization engine into the Chrome Web browser. Ernst was also research scientist at Intel Labs and a cofounder and the CEO of Bytes+Lights.

Ross McIlroy is a software engineer at Google and tech lead of V8's interpreter effort. He previously worked on Chrome's scheduling subsystem and mobile optimization efforts. Previously, McIlroy worked on various operating-system and virtual-machine research projects, including Singularity, Helios, Barrelfish, and HeraJVM.

Hannes Payer is a software engineer at Google, tech lead of the V8 JavaScript garbage collection effort, and a virtual-machine enthusiast. Prior to V8, Payer worked on Google's Dart virtual machine and various Java virtual machines.

Copyright held by owner/authors.

Just because you have been doing it the same way doesn't mean you are doing it the right way.

BY KATE MATSUDAIRA

Fresh Starts

I LOVE FRESH starts. Growing up, one of my favorite things was starting a new school year. From the fresh school supplies (I am still a sucker for pen and paper) to the promise of a new class of students, teachers, and lessons, I couldn't wait for summer to be over and to go back to school.

The same thing happens with new jobs (and to some extent, new teams and new projects). They reinvigorate you, excite you, and get you going.

The trouble is that starting anew isn't something you get to do all the time. For some people it might happen once a year, once every two years, or once every four years. Furthermore, learning something new isn't always in the best interest of your employer. Of course, great managers want you constantly to be learning and advancing your career, but if you are doing your job well, they also probably like the idea of keeping you in

that role where they can rely on you to get the work done. Putting you into a position where you will have to work hard to learn new skills isn't always best for your company—and so it probably doesn't happen often.

Wouldn't it be great if you frequently were in a position where you were pushed to grow outside of your comfort zone? Where you had to start new and fresh?

Well, the good news is you can. In fact, you can make your current position one that focuses on your growth and extends the boundaries of your knowledge—and that is all up to you.

In technology and computer science, almost more than any other field, a growth mind-set is mandatory for success. In this field the tools and best practices are constantly evolving—there is always something new to learn. For many people this high rate of change can be overwhelming, but for the right person this can mean opportunity. When you are willing to dive in and learn new skills, it puts you ahead of the game; and when you are strategic about what skills you learn, it can help you grow your career even faster.

No matter where you are in your career, there is more to learn. All of us can always use an excuse to get more invigorated and excited by our jobs. Here are three steps you can take to develop your current role and make tomorrow (or even the rest of today) a fresh start.

Create A Learning Plan

When you have been doing a job for a while, there isn't as much for you to learn in your day-to-day. Sure, there are always opportunities to improve little things, but your rate of knowledge acquisition slows down the longer you have been in a position. This makes it even more important to have a learning plan. You should have a list of things you plan to learn with some concrete tasks associated with each. If you need some inspiration on what should be on this list, here are some questions to ponder:

► To be promoted to the next level in your job, what do you need to accomplish? Are there any skills you need to acquire or improve?

► If you think 10 years into the future, what do you want to do? Do you know anyone doing that now? What do they know that you don't?

► Look back over your past performance reviews. Are there any areas where you could continue to develop and improve? If you ask others for feedback, what would they say and how can you do better?

Build Better Relationships

Most of us spend more time with our coworkers than our families. When you have great relationships with the people you work with every day, you tend to be happier—and you tend to be more productive and collaborative. Also, when people like you and want to help you, then you are more likely to get promoted and discover opportunities. Here are two ideas for improving your working relationships:

► *Improve your communication skills.* When you get better at writing email messages, or verbal presentation, you help share information, and this creates better decision making across your whole team.

► *Take someone to lunch.* If you work with someone you don't know very well, or haven't had the best working relationship with, make the first move and ask this person to lunch or coffee. This is a great way to get to know people and understand their points of view. Working relationships are usually strained because two sides are making incorrect assumptions, and the first step is opening the lines of communication. Be open, practice your listening skills, and offer to foot the bill—for the cost of a lunch you would be amazed at how much that gesture can improve your work life.

Make Better Use of Your Down Time

One of my favorite time-management tricks is using spare minutes to maximize your learning. When you can make

the most of the small moments and learn things that help advance your career, then you will be one step ahead. This can be as simple as nixing social-media checks and replacing them with 10–15 minutes of reading articles or websites that help increase your knowledge. Here are some other ideas to get more out of those little moments:

► *Be on time.* When you can start on time and end on time, you make the most of meetings (plus it is a sign of respect when you show up when you say you will), and you will have more freedom to do what you want to do.

► *Keep a reading queue.* Whether you use bookmarks, notes, or some other tool, keep a list of items you want to read. These can be articles, white papers, or books—but when you have a list it is much easier just to go there to fill 15 minutes with useful learning than to spend those 15 minutes surfing the web looking for something interesting.

► *Listen to audiobooks or smart podcasts.* Whether it is on your commute or when you are working out, if you can't sit and read, try listening to your lessons. There are so many great op-

tions here, and it is a great way to maximize time and knowledge.

Of course, there are lots of other great ways to make your old career new again, but these little ideas could give you inspiration so that when you come to work tomorrow you can be excited.

If you have any other thoughts or suggestions, feel free to send them to me. And if there is a topic you would like to see covered, let me know. 

Related articles on queue.acm.org

Lean Software Development - Building and Shipping Two Versions

Kate Matsudaira

<http://queue.acm.org/detail.cfm?id=2841311>

A Conversation with Matt Wells

<http://queue.acm.org/detail.cfm?id=988401>

Cherry-picking and the Scientific Method

Kode Vicious

<http://queue.acm.org/detail.cfm?id=2466488>

Kate Matsudaira (katemats.com) is the founder of her own company, Popforms. Previously she worked in engineering leadership roles at companies like Decide (acquired by eBay), Moz, Microsoft, and Amazon.

Copyright held by author.

Publication rights licensed to ACM. \$15.00.



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

Tame the dynamics of change by centralizing each concern in its own module.

BY ANDRE MEDEIROS

Dynamics of Change: Why Reactivity Matters

PROFESSIONAL PROGRAMMING IS about dealing with software at scale. Everything is trivial when the problem is small and contained: it can be elegantly solved with imperative programming or functional programming or any other paradigm. Real-world challenges arise when programmers have to deal with large amounts of data, network requests, or intertwined entities, as in user interface (UI) programming.

Of these different types of challenges, managing the dynamics of change in a code base is a common one that may be encountered in either UI programming or the back end. How to structure the flow of control and concurrency among multiple parties that need

to update one another with new information is referred to as *managing change*. In both UI programs and servers, concurrency is typically present and is responsible for most of the challenges and complexity.

Some complexity is accidental and can be removed. Managing concurrent complexity becomes difficult when the amount of essential complexity is large. In those cases, the interrelation between the entities is complex—and cannot be made less so. For example, the requirements themselves may already represent essential complexity. In an online text editor, the requirements alone may determine that a keyboard input needs to change the view, update text formatting, perhaps also change the table of contents, word count, paragraph count, request the document to be saved, and take other actions.

Because essential complexity cannot be eliminated, the alternative is to make it as understandable as possible, which leads to making it maintainable. When it comes to complexity of change around some entity Foo, you want to understand what Foo changes, what can change Foo, and which part is responsible for the change.

How Change Propagates from One Module to Another

Figure 1 is a data flow chart for a code base of e-commerce software, where rectangles represent modules and arrows represent communication. These modules are interconnected as requirements, not as architectural decisions. Each module may be an object, an object-oriented class, an actor, or perhaps a thread, depending on the programming language and framework used.

An arrow from the Cart module to the Invoice module (Figure 2a) means the cart changes or affects the state in the invoice in a meaningful way. A practical example of this situation is a feature that recalculates the total invoicing amount whenever a new product is added to the cart (Figure 2b).



The arrow starts in the `Cart` and ends in the `Invoice` because an operation internal to the `Cart` may cause the state of the `Invoice` to change. The arrow represents the dynamics of change between the `Cart` and the `Invoice`.

Assuming all code lives in some module, the arrow cannot live in the space between; it must live in a module, too. Is the arrow defined in the `Cart` or in the `Invoice`? It is up to the programmer to decide.

Passive Programming

It is common to place the arrow definition in the arrow tail: the `cart`. Code in the `Cart` that handles the addition of a new product is typically responsible for triggering the `Invoice` to update its invoicing data, as demonstrated in the chart and the Kotlin (<https://kotlin-lang.org/>) code snippet in Figure 3.

The `Cart` assumes a proactive role, and the `Invoice` takes a passive role. While the `Cart` is responsible for the change and keeping the `Invoice` state up to date, the `Invoice` has no code indicating the update is coming from the `Cart`. Instead, it must expose `updateInvoicing` as a public method. On the other hand, the `cart` has no ac-

cess restrictions; it is free to choose whether the `ProductAdded` event should be private or public.

Let's call this programming style *passive programming*, characterized by remote imperative changes and delegated responsibility over state management.

Reactive Programming

The other way of defining the arrow's ownership is *reactive programming*, where the arrow is defined at the arrow head: the `Invoice`, as shown in Figure 4. In this setting, the `Invoice` listens to a `ProductAdded` event happening in the `cart` and determines that it should change its own internal invoicing state.

The `Cart` now assumes a broadcasting role, and the `Invoice` takes a reactive role. The `Cart`'s responsibility is to carry out its management of purchased products, while providing notification that a product has been added or removed.

Therefore, the `Cart` has no code that explicitly indicates its events may affect the state in the `Invoice`. On the other hand, the `Invoice` is responsible for keeping its own invoicing state up to date and has the `Cart` as a dependency.

The responsibilities are now inverted, and the `Invoice` may choose to have its `updateInvoicing` method private or public, but the `Cart` must make the `ProductAdded` event public. Figure 5 illustrates this duality.

The term *reactive* was vaguely defined in 1989 by Gérard Berry.¹ The definition given here is broad enough to cover existing notions of reactive systems such as spreadsheets, the actor model, Reactive Extensions (Rx), event streams, and others.

Passive vs. Reactive for Managing Essential Complexity

In the network of modules and arrows for communication of change, where should the arrows be defined? When should reactive programming be used and when is the passive pattern more suitable?

There are usually two questions to ask when trying to understand a complex network of modules:

- ▶ Which modules does module *X* change?
- ▶ Which modules can change module *X*?

The answers depend on which approach is used: reactive or passive, or

Figure 1. Data flow for a codebase of e-commerce software.

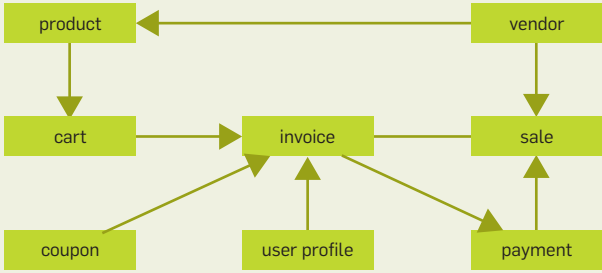


Figure 3. Passive programming with code in tail.



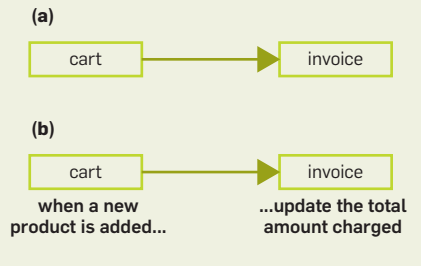
Figure 4. Reactive programming with code in head.



Figure 5. Public vs. private.

Programming	"Product added" event in the Cart	Update invoicing data method in the Invoice
Passive	private or public	public
Reactive	public	private or public

Figure 2. The Cart changes the Invoice.



both. Let's assume, for simplicity, that whichever approach is chosen, it is applied uniformly across the architecture. For example, consider the network of e-commerce modules shown in Figure 6, where the passive pattern is used everywhere. To answer the first question for the Invoice module (Which modules does the invoice change?), you need only to look at the code in the Invoice module, because it owns the arrows and defines how other modules are remotely changed from within the Invoice as a proactive component.

To discover which modules can change the state of the Invoice, however, you need to look for all the usages of public methods of the Invoice throughout the code base.

In practice, this becomes difficult to maintain when multiple other modules may change the Invoice, which is the case in essentially complex software. It may lead to situations where the programmer has to build a mental model of how multiple modules concurrently modify a piece of state in the module in question. The opposite alternative is to apply the reactive pattern everywhere, illustrated in Figure 7.

To discover which modules can change the state of the Invoice, you can just look at the code in the Invoice module, because it contains all "arrows" that define dependencies and dynamics of change. Building the mental model of concurrent changes is easier when all relevant entities are co-located.

On the other hand, the dual concern of discovering which other modules the Invoice affects can be answered only by searching for all usages of the Invoice module's public broadcast events.

When arranged in a table, as in Figure 8, these described properties for passive and reactive are dual to each other.

The pattern you choose depends on

which of these two questions is more commonly on a programmer's mind when dealing with a specific code base. Then you can pick the pattern whose answer to the most common question is, "look inside," because you want to be able to find the answer quickly. A centralized answer is better than a distributed one.

While both questions are important in an average code base, a more common need may be to understand how a particular module works. This is why reactivity matters: you usually need to know how a module works before looking at what the module affects.

Because a passive-only approach generates *irresponsible* modules (they delegate their state management to other modules), a reactive-only approach is a more sensible default choice. That said, the passive pattern is suitable for data structures and for creating a hierarchy of ownership. Any common data structure (such as a hash map) in object-oriented programming is a passive module, because it exposes methods that allow changing its internal state. Because it delegates the responsibility of answering the question "When does it change?" to whichever module contains the data-structure object, it creates a hierarchy: the containing module as the parent and the data structure as the child.

Managing Dependencies and Ownership

With the reactive-only approach, every module must statically define its dependencies to other modules. In the Cart and Invoice example, Invoice would need to statically import Cart. Because this applies everywhere, all modules would have to be singletons. In fact, Kotlin's object keyword is used (in Scala as well) to create singletons.

In the reactive example in Figure 9, there are two concerns regarding dependencies:

- What the dependency is: defined by the import statement.
- How to depend: defined by the event listener.

The problem with singletons as dependencies relates only to the what concern in the reactive pattern. You would still like to keep the reactive style of how dependencies are put together, because it appropriately answers the question,

Figure 6. Frequent passive pattern.

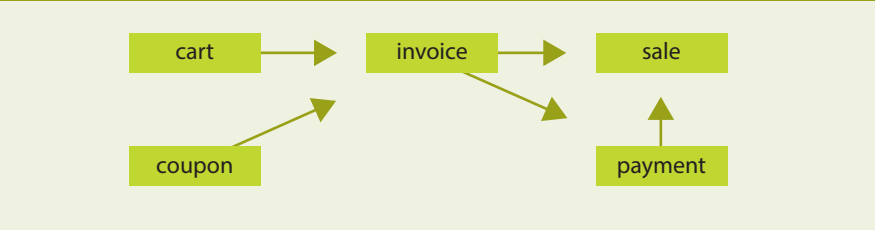
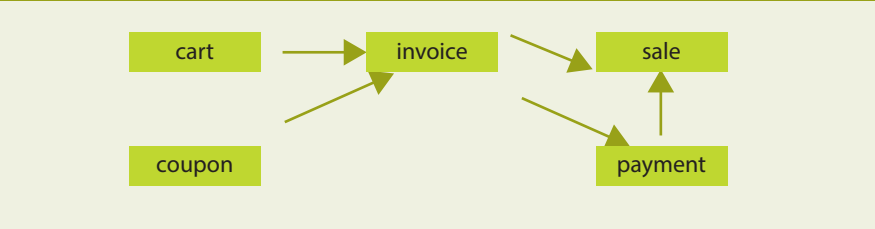


Figure 7. Frequent reactive pattern.



"How does the module work?"

While reactive, the module being changed is statically aware of its dependencies through imports; while passive, the module being changed is unaware of its dependencies.

So far, this article has analyzed the passive-only and reactive-only approaches, but in between lies the opportunity for mixing both paradigms: keeping only the *how* benefit from reactive, while using passive programming to implement the *what* concern.

The Invoice module can be made passive with regard to its dependencies: it exposes a public method to allow another module to set or inject a dependency. Simultaneously, Invoice can be made reactive with regard to how it works. This is shown in the example code in Figure 10, which yields a hybrid passively reactive solution:

- How does it work? Look inside (reactive).
- What does it depend on? Injected

via a public method (passive).

This would help make modules more reusable, because they are not singletons anymore. Let's look at another example where a typical passive setting is converted to a passively reactive one.

Example: Analytics Events

It is common to write the code for a UI program in passive-only style, where each different screen or page of the program uses the public methods of an Analytics module to send events to an Analytics back end, as illustrated in the example code in Figure 11.

Figure 8. Dual properties.

	Passive	Reactive
How does it work?	Find usages	Look inside
What does it affect?	Look inside	Find usages

Figure 9. Reactive-only approach.

```

package my.project

import my.project.Cart // This is a singleton

public object Invoice { // This is a singleton too
    fun updateInvoicing(product: Product) {
        // ...
    }

    fun setup() {
        Cart.onProductAdded { product ->
            this.updateInvoicing(product)
        }
    }
}
  
```

Figure 10. A hybrid passively reactive solution.

```
package my.project

public object Invoice {
    fun updateInvoicing(product: Product) {
        // ...
    }

    private var cart: Cart? = null

    public fun setCart(cart: Cart) {
        this.cart = cart
        cart.onProductAdded { product ->
            this.updateInvoicing(product)
        }
    }
}
```

Figure 11. Passive-only approach.

```
// In the LoginPage module
package my.project

import my.project.Analytics

val loginButton = //...
loginButton.addClickListener { clickEvent ->
    Analytics.sendEvent('User clicked the login button')
}
```

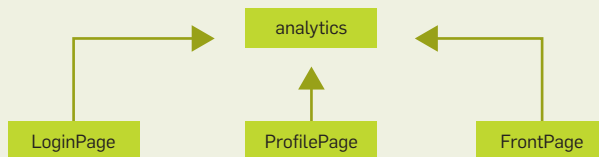
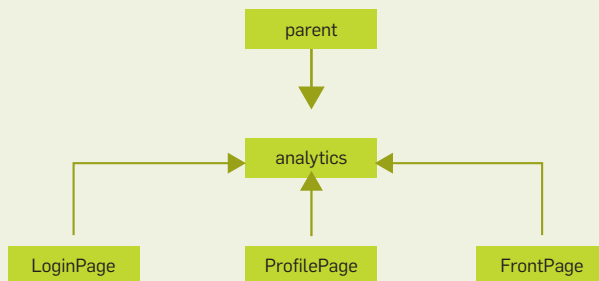


Figure 12. Public injection method.



```
// In the Analytics module
package my.project

public object Analytics {
    public fun inject(loginPage: Page) {
        loginPage.loginButton.addClickListener { clickEvent ->
            this.sendEvent('User clicked the login button')
        }
    }

    private fun sendEvent(eventMessage: String) {
        // ...
    }
}
```

The problem with building a passive-only solution for analytics events is that every single page must have code related to analytics. Also, to understand the behavior of analytics, you must study it scattered throughout the code. It is desirable to separate the analytics aspect from the core features and business logic concerning a page such as the LoginPage. Aspect-oriented programming² is one attempt at solving this, but it is also possible to separate aspects through reactive programming with events.

In order to make the code base reactive only, the Analytics module would need to statically depend on all the pages in the program. Instead, you can use the passively reactive solution to make the Analytics module receive its dependencies through a public injection method. This way, a parent module that controls routing of pages can also bootstrap the analytics with information on those pages (see Figure 12 for an example).

Mind the Arrows

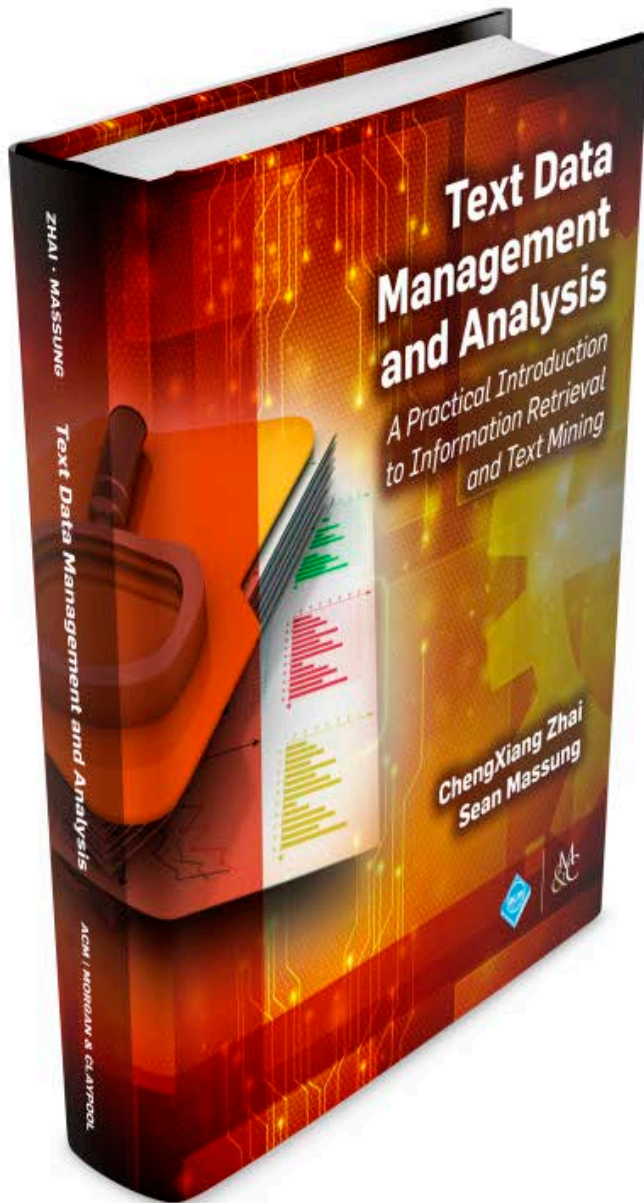
Introducing reactive patterns in an architecture can help better define which module owns a relationship of change between two modules. Software architectures for essential complex requirements are often about structuring the code in modules, but do not forget that the arrows between modules also live in modules. Some degree of reactivity matters because it creates separation of concerns. A particular module should be responsible for its own state. This is easily achievable in an event-driven architecture, where modules do not invasively change each other. Tame the dynamics of change by centralizing each concern in its own module. **C**

References

1. Berry, G. Real-time programming: Special-purpose or general purpose languages. RR-1065. INRIA, 1989; <https://hal.inria.fr/inria-00075494/document>.
2. Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J. M., Irwin, J. Aspect-oriented programming. In *Proceedings of the 11th European Conference on Object-Oriented Programming* (1997), 220–242.

Andre Medeiros is a Web and mobile developer at Futurice. His work focuses on reactive programming for user interfaces, particularly with the ReactiveX libraries. Medeiros has built JavaScript libraries and tools such as Cycle.js and RxMarbles.

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



The most useful and practical knowledge for building a variety of text data applications.

CS STUDENTS (Undergrad & Graduate)
LIBRARY & INFORMATION SCIENTISTS
TEXT DATA PRACTITIONERS

ChengXiang Zhai & Sean Massung (Authors)
University of Illinois at Urbana-Champaign

Text Data Management and Analysis covers the major concepts, techniques, and ideas in **information retrieval** and **text data mining**. It focuses on the practical viewpoint and **includes many hands-on exercises designed with a companion software toolkit** (i.e., MeTA) to help readers learn how to apply techniques of information retrieval and text mining to real-world text data. It also shows readers how to **experiment** with and **improve** some of the algorithms for interesting application tasks. The book can be used as a **text** for computer science undergraduates and graduates, library and information scientists, or as a **reference** for practitioners working on relevant problems in **managing and analyzing text data**.



<http://books.acm.org>
<http://www.morganclaypoolpublishers.com/text>

DOI:10.1145/2896817

Combine simple whitelisting technology, notably prefix filtering, in most BGP-speaking routers with weaker cryptographic protocols.

BY ROBERT LYCHEV, MICHAEL SCHAPIRA, AND SHARON GOLDBERG

Rethinking Security for Internet Routing

ON JUNE 12, 2015, an incident in the Asia-Pacific region caused network performance problems for hundreds of thousands of Internet destinations, including Facebook and Amazon.^{24,37} It was not the result of a natural disaster, a failed transatlantic cable, or a malicious attack. Instead, it resulted from a misconfiguration at a Malaysian ISP that inadvertently exploited the Internet's Border Gateway Protocol (BGP) to disrupt connectivity at networks in Malaysia and beyond. BGP establishes Internet connectivity by setting up routes between independently operated networks. Over the past two decades, several high-profile routing incidents (often resulting from misconfigurations^{4,8,28,30,37}) have regularly demonstrated that BGP is highly vulnerable to malicious attacks. BGP attacks cause a victim network Internet traffic to be rerouted to the attacker's own

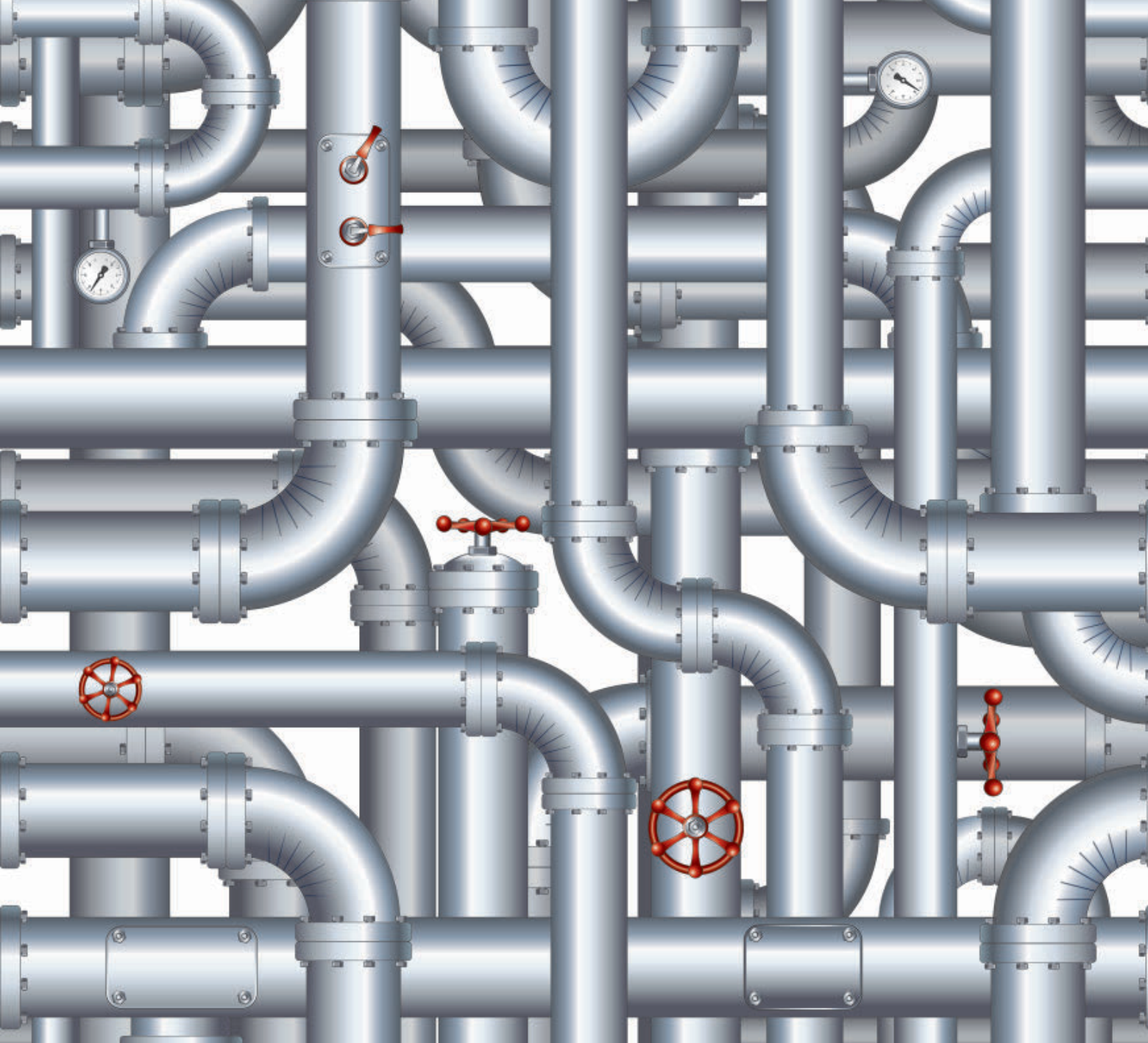
network. The rerouted traffic might then be dropped before it reaches its legitimate destination^{4,28,30,37} or, more deviously, be subject to eavesdropping,^{2,32} traffic analysis,³⁶ or tampering.^{15,21,34}

Barriers to securing BGP. To deal with these vulnerabilities, the Internet community has spent almost two decades considering a variety of protocols for securing BGP.⁵ Today, however, Internet routing remains largely unprotected by BGP security protocols. The sluggish deployment of BGP security is the result of economic, operational, and policy challenges. The root cause for this situation is that the Internet lacks a single authority that can mandate deployment of BGP security upgrades. Deployment decisions are instead made by independently operated networks according to their own local policy and business objectives. BGP security is adopted by a network only if its security benefits are thought to justify its deployment and operational costs. Moreover, the diversity of BGP security protocols has led to some controversy as to which protocol should actually be deployed. This issue is exacerbated by the fact that each protocol offers different security benefits and comes with different costs.

Our goal. Which BGP security protocol should be deployed throughout the Internet? To answer, we have developed a framework for quantifying the security benefits provided to the Internet as a whole by different BGP security protocols. We begin with a full deployment scenario, where a BGP security protocol is deployed by every network in the Internet. In practice, how-

» key insights

- **Prefix filtering, a simple whitelisting technology available in most routers today, can provide valuable security benefits.**
- **Operational issues during partial deployment can cause the strongest cryptographic protocols, including BGPSEC, to deliver fewer security benefits than might otherwise be expected.**
- **The results explored here support deploying a combination of prefix filtering with weaker cryptographic protocols like the RPKI.**



ever, full deployment remains elusive. Indeed, a standardized BGP security protocol, the Resource Public Key Infrastructure (RPKI),²⁰ has been in the process of deployment since the start of this decade but currently contains security information about only $\approx 5\%$ of the Internet's routes.²⁹ We thus also study partial deployment scenarios, where some networks deploy a BGP security protocol, but others do not.

Our results. We obtained our results via simulations of BGP routing on empirically measured graphs of the Internet's topology. This article focuses on the security benefits provided by a given protocol, on average, to the welfare of

the Internet as a whole. First, we find that valuable security benefits can be provided by simple whitelisting technologies—prefix filtering⁹—and they are comparable even to those provided by the strongest cryptographic protocols, notably BGPSEC.¹⁹ Next, we find that partial deployment has been a blind spot for the network routing community; operational issues that can arise during partial deployment cause the strongest cryptographic protocols to provide fewer security benefits than might initially have been expected. Taken together, our results call for rethinking the efforts to promote only the strongest cryptographic routing protocols (such as BGPSEC).

Our results point instead toward using a combination of simple whitelisting technology (prefix filtering) available in most BGP-speaking routers today with weaker cryptographic protocols (such as the RPKI).

BGP Threats and Defenses

How does routing with BGP work? Why is BGP vulnerable to attacks? And what BGP security protocols are available to patch these vulnerabilities? We explore the answers using Figure 1.

Routing with BGP. The Internet can be regarded as a network of autonomous systems (ASes). ASes are large networks operated by different organi-

zations, each with a different AS number. In Figure 1, AS 27781 is operated by an ISP in St. Maarten and AS 23520 by an ISP serving the Caribbean, AS 701 is Verizon’s backbone network, and AS 6939 is Hurricane Electric’s backbone network. Viewed at this resolution, the Internet can be described as a graph where nodes are ASes and edges are the links between them. Interconnections between ASes change on a very slow timescale (months to years), so we treat the edges in the AS graph as static.

Neighboring ASes remunerate each other for Internet service according to their business relationship. We show two key business relationships in Figure 1: customer-to-provider, where the customer AS purchases Internet connectivity from its provider AS (a directed edge from customer to provider) or settlement-free peering, where two ASes transit each other’s customer traffic for free (an undirected edge). Such free peering agreements are often established between ASes of equal size or between large content providers (such as Google and Microsoft) and other ASes. Figure 1 is a subgraph of the IPv4 AS topology inferred by Chi et al.⁶ using data from September 24, 2012, and contains 39,056 ASes, 73,442 customer-provider links, and 62,129 settlement-free peering links. All results in this article are based on this IP version 4 (IPv4) AS-level graph; we consider an IPv6 AS-level graph in the online appendix.

IP prefixes. Instead of maintaining routes to all possible Internet Protocol (IP) addresses, ASes use BGP to discover routes to a much smaller number of IP prefixes. An IPv4 is a 32-bit address (such as 72.252.8.8) where every number is a byte in decimal, and the dots are separators. An IP prefix is a set of IP ad-

dresses with a common prefix (such as 72.252.0.0/16) is the set of IP addresses $\{72.252.0.0, 72.252.0.1, \dots, 72.252.255.255\}$, where the notation /16 (“slash sixteen”) implies the first 16 bits (the “prefix”) are common to all addresses in the set (namely, those beginning with 72.252). IP prefixes have variable lengths, and the addresses in one IP prefix may contain the addresses in another; for example, IP prefix 72.252.0.0/16 contains IP prefix 72.252.8.0/21. Each AS is allocated a set of IP prefixes; for example, 72.252.0.0/16 is allocated to AS 23520, while 72.252.8.0/21 is allocated to AS 27781 in Figure 1.

Longest-prefix-match routing. The IP address 72.252.8.8 is contained in both IP prefix 72.252.0.0/16 and IP prefix 72.252.8.0/21. So how should routers forward IP packets with destination IP address 72.252.8.8? To avoid ambiguity, every Internet router identifies the longest IP prefix that covers the destination IP address in the packet and forwards the packet along the route to that IP prefix. In Figure 1, a packet with destination IP address 72.252.8.8 would be forwarded on a route to the longer 21-bit IP prefix 72.252.8.0/21 allocated to AS 27781, rather than the shorter 16-bit IP prefix 72.252.0.0/16 allocated to AS 23520.

Learning paths with BGP. How do ASes use BGP to learn the path to 72.252.8.0/21 in Figure 1? The process starts when AS 27781 originates the BGP announcement

72.252.8.0/21 : 27781

to each of its neighbors, including AS 23520. AS 23520 selects this route and sends the BGP announcement

72.252.8.0/21 : 27781, 23520

to its neighbor AS 2828. This process continues until AS m learns following two routes

72.252.8.0/21 :27781, 23520; 2828; 6939
72:252.8.0/21 :27781, 701, 16795

in BGP announcements from its neighbors AS 6939 and AS 16795, respectively. We say these routes are available to AS m , since every AS on each route has actually announced the route to its neighbor on the route. We next discuss why AS m does not have an available route through AS 5580.

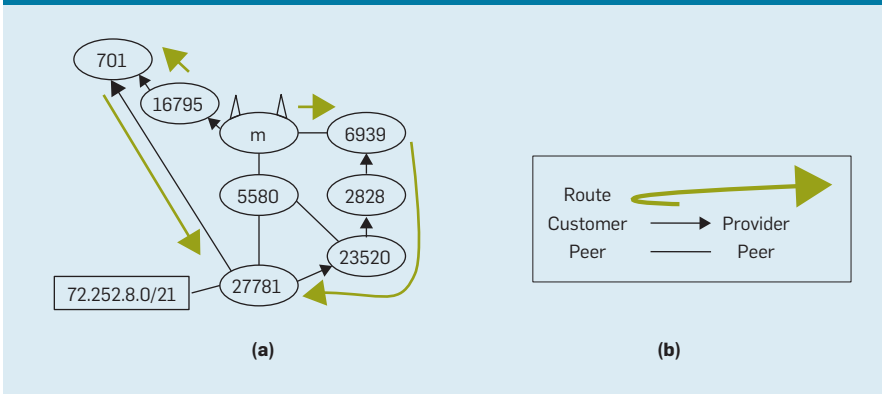
Each AS uses its local BGP routing policy to select a single best route from the set of routes it learns from its neighbors. In Figure 1, AS m selects the route through its peer AS 6939; m prefers the “peer route” through its peer AS 6939 (because this route comes at no monetary cost) over the “provider route” through its provider AS 16795 (which comes at monetary cost). Once an AS selects a route, it announces that route to a subset of its neighbors according to its local export policy. In Figure 1, AS 5580 chooses not to announce, to its neighbor AS m , its one-hop peer route to AS 27781. This is because AS 5580 will transit traffic from only one neighbor (AS m) to another (AS 27781) if at least one neighbor is a customer that pays for Internet service.

Why is BGP insecure? BGP is “insecure” because any AS can announce any path it wants to any subset of its neighbors. An attacker can exploit this to reroute traffic to the attacker’s own network.

Threat model. We consider a single attacking AS m that wants to attract, to m ’s own network, the traffic destined to an IP prefix that is legitimately allocated to a victim AS d and actually announced in BGP by d . Outside our scope are attacks (such as by spammers^{33,39}) on “unused” IP prefixes, or prefixes that are either not allocated or not announced in BGP.^a Attacker m can announce any path it wants to any neighbor it wants, even if that path is bogus, or if announcing that path vio-

a Because m ’s route is the only way to reach an unused IP prefix, m attracts traffic from all ASes.

Figure 1. Subgraph of Chi et al.’s empirical AS graph.⁶



lates m 's normal export policies. However, because m cannot lie to its neighbor about their business relationship or about m 's own AS number (because this information is programmed into its neighbor's routers), any path announced by m must include m 's own AS number as the first hop. Finally, the threat of multiple colluding ASes is out of scope, since the strongest proposals for securing BGP cannot withstand this threat,^b and most attacks in the wild involve only a single attacking AS.

We now illustrate the existence of threats to BGP by choosing examples from Figure 1 and simulating them on Chi et al.'s Internet topology,⁶ using a framework described later; the impact of these threats is also described later. (Experts might thus wish to read the section on quantifying security benefits first.) The following threats are commonly seen in the wild.

Threat. Subprefix hijack. One devastating attack on BGP is the subprefix hijack.^{4,21,28,31,34} If AS m wishes to launch a subprefix hijack on the victim's IP prefix 72.252.8.0/21 in Figure 1, it announces to each of its neighbors a route, such as

72.252.8.0/24 : m

Importantly, AS m is not actually allocated this subprefix. Nevertheless, longest-prefix match routing still ensures that any AS that learns the bogus route to the subprefix 72.252.8.0/24 through m will forward all IP packets destined for addresses in this subprefix to m . Notice, because of longest-prefix-match routing, the actual ASes on the attacker's route are irrelevant.

Threat. Prefix hijack. In a prefix hijack,^{8,37} the hijacker originates the exact same IP prefix that belongs to a victim. The attacker m in Figure 1 can launch this attack on a victim IP prefix 72.252.8.0/21 by announcing

72.252.8.0/21 : m

to its neighbors. Rather than attracting 100% of traffic, as in a subprefix hijack, in a prefix hijack, traffic will split, with ASes "closer" to the hijacker selecting the hijacked route, and

BGP is "insecure" because any AS can announce any path it wants to any subset of its neighbors.

ASes "closer" to the legitimate origin AS selecting the legitimate route. We simulated this attack by m on prefix 72.252.8.0/21 on Chi et al.'s Internet topology⁶ and found that 56% of ASes route through m and 44% route through AS 27781.

Threat. Route leak. In a route leak, an attacker violates its normal export policies by announcing a legitimate route to too many of its neighbors.²⁷ In Figure 1, AS 5580 could violate its normal export policies by leaking the route

72.252.8.0/21 : 27781, 5580

to all its neighbors, including its peers AS 23520 and AS m . This violates AS 5580's normal export policy, which requires AS 5580 to announce peer routes, or routes through a neighboring peer, to neighboring customers only. While route leaks might seem innocuous, several incidents observed in the wild have proved to be quite damaging.^{30,37} Simulations, as discussed later, show that the route leak allows AS 5580 to attract traffic from 45% of ASes in the graph instead of the 3.7% it would attract under normal conditions.

How can we secure BGP? The past two decades have seen several proposals for securing BGP.⁵ To simplify the landscape, we describe the most prominent proposals in terms of their security guarantees, as well the threats they can and cannot prevent.

Defense. Origin validation. Prefix and subprefix hijacks arise because BGP does not offer a way to validate the allocation of IP prefixes. To remedy this deficiency, origin validation^{5,20,26} provides a trusted database binding ASes to the IP prefixes allocated to them, and any BGP message that does not adhere to this binding is ignored. With origin validation, in Figure 1, m cannot hijack prefix 72.252.8.0/21, because the trusted database does not contain a binding from that prefix to m . The only proposal for origin validation that has seen some deployment²⁹ is a cryptographic certificate infrastructure called the Resource Public Key Infrastructure (RPKI).²⁰ The RPKI is relatively lightweight and requires neither online cryptographic computations nor changes to the BGP message structure. Instead, ASes download and cryptographically validate RPKI certificates to a local cache at infrequent intervals

^b Even fully deployed BGPSEC cannot guarantee path validation when multiple ASes collude.³

(daily), then check the BGP messages they receive against the already-cryptographically validated information in their local cache.

Threat. The one-hop hijack. While origin validation eliminates prefix and subprefix hijacks, it cannot prevent an attacker from announcing any path that ends at the AS that is legitimately allocated the victim's IP prefix. Origin validation does not stop AS m in Figure 1 from launching a one-hop hijack, where m announces the route

72.252.8.0/21 : 27781, m

to each of its neighbors. Because the 72.252.8.0/21 is legitimately allocated to AS 27781, this route will not be discarded by origin validation. However, this route is bogus because no edge exists between m and 27781. Simulations, as discussed later, show that this causes 31% of ASes to select bogus routes through m , instead of the legitimate AS 27781.

Defense. Topology validation. The one-hop hijack succeeded because origin validation fails to validate that the first edge (between m and AS 27781) in the BGP announcement actually exists in the AS graph. Topology validation validates that every edge in a BGP announcement exists in the AS graph. Secure Origin BGP (soBGP)⁴⁰ is a well-known proposal that provides topology validation. Like the RPKI, soBGP uses a cryptographic certificate infrastructure that ASes infrequently download to their local caches to provide origin validation and certify the presence of links between pairs of ASes.^c Like the RPKI, it requires neither changes to the BGP message structure nor to online cryptographic computations.

Threat. Announce an unavailable path. Topology validation does not prevent an attacker from announcing a path that exists in the AS graph, but is not available, or has not been announced by each AS on the route. In Figure 1, the attacker m can attract traffic from 17% of ASes, as discussed later, by announcing the short path

72.252.8.0/21 : 27781, 5580, m

to each of its neighbors, instead of its legitimate longer path through AS 6939 shown in Figure 1. While this short path exists in the topology, it is still bogus because it is not actually available to m ; this follows because AS 5580 has an export policy that forbids it from announcing the peer path through AS 27781 to its peer m .

Defense. Path validation. To prevent this attack, path validation forces the attacker to announce only available paths. Path validation is a “gold standard” for BGP security, and several proposals provide this security guarantee;^{5,18,19} of them, BGPSEC has the most traction and is being standardized. BGPSEC is relatively heavyweight, requiring deployment of the RPKI, each AS on a route to append its cryptographic digital signature to every BGP message, and each AS on a route to cryptographically validate, in real time, every signature on every BGP message it receives. The computational overhead involved in signing and validating routes for all 500,000 of the Internet's IP prefixes, in real time and even under router failure-recovery scenarios, could require routers to be upgraded with crypto to hardware accelerators.

Threat. Announce an available path. Path validation does not prevent an attacker from announcing a short available path to each of its neighbors, even if doing so violates the attacking AS's normal export policies, or if the attacker is not actually forwarding traffic along the announced path.^d In Figure 1, m attracts traffic from < 0.01% of ASes when m honestly announces its preferred five-hop peer path through AS 6939. Meanwhile, even with path validation, m can instead announce its shorter (but less-preferred) four-hop provider path

72.252.8.0/21 : 27781, 701; 16795, m

to all its neighbors, allowing m to attract traffic from 4.5% of ASes. Notice this is also a route leak.

Under the assumption there is only a single attacker occupying a single AS, it follows that path validation is a strictly stronger defense than topology

validation, and topology validation is strictly stronger than origin validation. Any attack that succeeds against the stronger defense also succeeds against the weaker one. Moreover, each defense provides a mechanism for validating the correctness of BGP announcements, but neither restricts the export policies an AS can use. All three defenses are thus still vulnerable to route leaks. Fortunately, there is an orthogonal defense that can sometimes prevent route leaks.

Defense. Prefix filtering. In this article, we assume that prefix filtering whitelists the BGP announcements made by stub ASes; a stub is an AS with no customers. As stubs are consumers (rather than providers) of Internet service, stubs should carry only incoming traffic destined for their own allocated IP prefixes. If each provider AS has a “prefix list” of the IP prefixes allocated to its customers that are stub ASes, when a stub announces any path to any IP prefix that is not allocated to the stub, the stub's provider could thus ignore that announcement. If all providers of a particular stub AS implement prefix filtering, prefix filtering completely eliminates every possible attack by that stub, including route leaks. Prefix filtering requires no changes to the BGP message structure, uses access control lists rather than cryptography, is available in most BGP speaking routers, and can be combined with any of the defenses we described earlier. However, it does require an ISP to maintain prefix lists for each customer, by collecting its own data or by using information in public databases (such as Internet Routing Registries³⁵). Because it can be challenging for ISPs to keep this information up to date,³⁵ we use a conservative definition of prefix filtering; indeed, in practice, many ISPs filter not only their stubs but also their customer cone, or their customers, their customer's customers, and their customer's customer's customers.⁹ In the next section we show that even this conservative definition provides tangible security benefits, even though it cannot prevent attacks by non-stub ASes.

Full Deployment

What defense should be deployed in the Internet? Should ISPs require the

^d The process of determining the AS-level path that packets take through the Internet is error prone³⁵ and can be biased by adversarial ASes.³²

^c soBGP also allows ASes to indicate a link's business relationship, though our analysis does not include this functionality.

“gold standard” defense of path validation, which comes at the cost of on-line cryptographic computations and a modifications to the BGP message structure? Or does the lighter offline cryptography used for origin validation suffice? Should ISPs forgo cryptography altogether and use prefix filtering instead? We aim to answer these questions quantitatively by comparing the efficacy of each defense discussed earlier at limiting the impact of routing attacks. Because we cannot just go out and launch BGP attacks on the Internet, we instead simulate attacks on the empirically measured AS-level topology⁶ described earlier. In this section, we assume a particular defense is fully deployed by every AS. We consider partial deployment scenarios later. We next present our quantitative framework, then describe our results.

Quantifying security benefits. We earlier illustrated the existence of threats to BGP using examples from Figure 1. But how representative are these examples, and what sort of damage can each attack cause? To answer, we simulate routing when attacker AS m performs an attack on victim AS d , and determine what source ASes in the AS graph are safe, or do not select a route that passes through m 's network, and which are deceived, or are not safe. Then, to get a measure of the global damage m caused to d by the attack, we count the number of deceived ASes when m attacks d . Finally, we measure the overall damage caused by the attack by averaging the number of deceived ASes over randomly selected pairs of attacker AS m and victim AS d . This measurement also allows us to avoid predicting what ASes will launch an attack or what ASes an attack might target. (Other approaches for measuring damage are discussed in the online appendix.)

Modeling routing policies and export policies. We need a concrete model of how ASes select routes during attacks. In practice, ASes' routing policies can differ between ASes and are often kept private, so we use the classic routing model of Gao and Rexford¹¹ and Huston,¹⁷ which was shown to capture the policies of many ASes.¹ The model assumes each AS executes the following steps (in order) when choosing a single

route from the set of routes to a given IP prefix:

Local pref (LP). Prefer “customer routes” (through a neighboring customer) that generate revenue over “peer routes” (through a neighboring peer) that are revenue neutral over “provider routes” (through a neighboring provider) that come at a cost;^e

AS paths (SP). Prefer shorter routes over longer routes; and

Tiebreak (TB). Use other criteria (such as geographic location) to break ties among remaining routes; we lack empirical information about how ASes implement their TB step, so, unless stated otherwise, we model this step as if it were done randomly.

After selecting a single route, an AS announces that route to a subset of its neighbors:

Export policy (Ex). A customer route is exported to all neighbors. Peer routes and provider routes are exported to customers only. This export policy captures ASes' willingness to transit traffic from one neighbor to another only if at least one neighbor is a paying customer.

LP implies that AS m in Figure 1 prefers the peer route through AS 6939 over the provider route through AS 16795. Moreover, Ex implies that AS 5580 in Figure 1 does not announce to its peer AS 23520 the direct peer route to the destination AS 27781.

^e We discuss the robustness of these results to other LP models in the online appendix.

Applying our threat model. How can we quantify the “security” provided by each of the four secure routing protocols described earlier? Ideally, we would measure the overall damage caused by the most damaging attack on each protocol. Intuition suggests the most damaging attack is as follows:

Naive attack. Announce to every neighbor the shortest possible path that is valid according to the secure routing protocol. When attacking BGP (respectively, origin validation, topology validation, path validation), the attacker thus announces to each of its neighbors a subprefix hijack (respectively, one-hop hijack, the shortest path that exists in the topology, the shortest path available to the attacker).

The naive attack is not, however, the most damaging. Indeed, in Goldberg et al.,¹⁴ we proved it is NP-hard to find an attacker's optimal traffic-attraction attack strategy. In fact, it is sometimes more effective to announce a longer path instead of a shorter path. We thus use the naive attack to lower bound the damage that might be done by an attacker; this lower bound suffices to allow us to compare secure routing protocols.

Comparing defenses. We simulate naive attacks for each secure routing protocol on the empirical AS-level topology⁶ for 292,000 randomly chosen (attacker AS, victim AS) pairs. This number of simulations was sufficient for our results to stabilize.

Figure 2. We present the average fraction of safe ASes for BGP, origin, to-

Figure 2. Comparing defenses. The average percentage of safe ASes during naive attack with a randomly chosen (attacker, victim) pair; error bars represent one standard deviation; and the horizontal line represents the effect of prefix filtering.

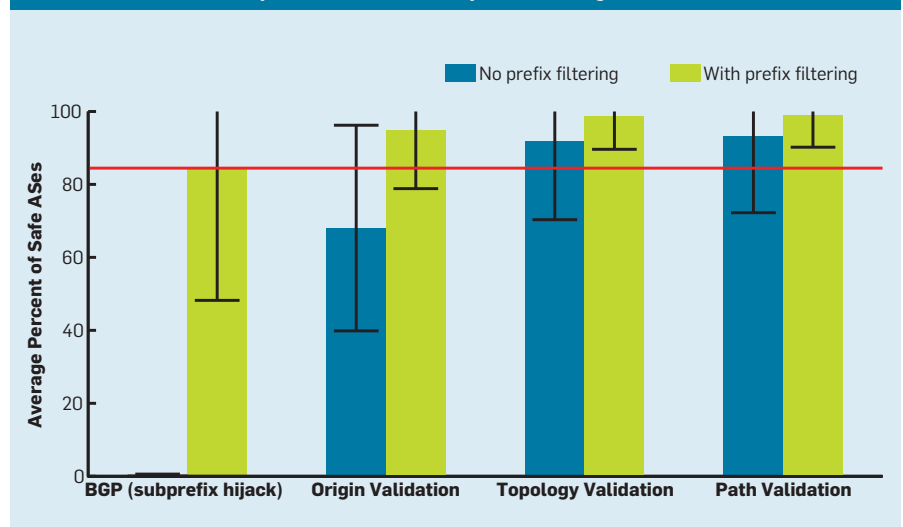
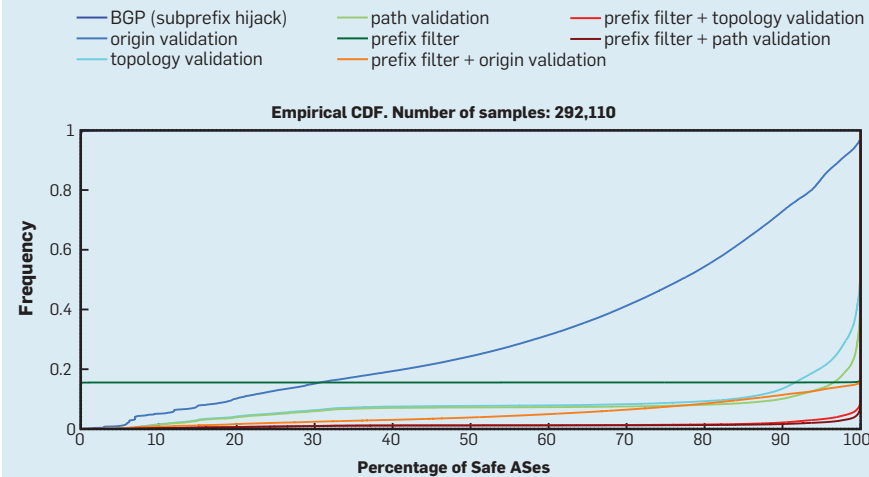


Figure 3. Cumulative distribution function of percentage of safe ASes during naive attack with a randomly chosen (attacker, victim) pair.



pology, and path validation. The right (yellow) and left (blue) bars represent results when prefix filtering is and is not used in combination with each defense, respectively. The horizontal line represents the percentage of attacks completely eliminated by prefix filtering. Since 85% of ASes in the AS graph are stubs, prefix filtering guarantees that only 15% of non-stub attackers can successfully launch a naive attack on any given victim.

Figure 3. Since Figure 2 presented only averages, we present the full picture in Figure 3. For each defense, we plot the empirical cumulative distribution function for the percentage of safe ASes. The BGP curve is essentially a step function at $x = 0$, since the attacker that launches a subprefix hijack attracts traffic from 100% of ASes in almost every simulation. Meanwhile, the prefix filtering curve is a step function at $x = 0$ with $y = 15\%$; for $\approx 15\%$ of simulations where the attacker is not a stub, the attacker launches a subprefix hijack and attracts traffic from 100% of ASes, while for the remaining simulations the attacker is a stub and is thus forced to behave honestly. Likewise, the combination of prefix filtering with origin validation causes the origin validation curve to shift into the bottom 15% of the plot; again, this happens because only the 15% of attackers that are non-stubs can launch a naive attack.

Despite the fact that we used suboptimal attack strategies for the attacker, we can still make several observations:

- With fully deployed prefix filtering, at most 15% of ASes can be deceived when we average over all (attacker, destination) pairs; this follows because 85% of ASes are stubs, and fully deployed prefix filtering eliminates all attacks by stubs;

- Even though the attacker runs a suboptimal attack strategy, and even when we assume path validation (without prefix filtering) is fully deployed, we still find that on average $100 - 93 = 7\%$ of ASes are deceived when path validation is fully deployed; path validation makes just $15 - 7 = 8\%$ more ASes safe, on average, over prefix filtering alone;

- The benefits provided by topology validation and path validation are almost identical, even though path validation is a stronger defense. In Goldberg et al.,¹⁴ we argued this is because path lengths in the Internet are fairly short. The paths the attacker announces with topology validation are only slightly shorter (on average) than those it can announce with path validation; and

- The combination of origin validation with prefix filtering provides benefits comparable to those provided by topology/path validation alone.

Takeaways. Our results suggest that origin/topology/path validation protocols are dealing with only one half of the problem; while they do restrict (a) the path the attacker can choose to announce, they fail to restrict (b) its export policies. Meanwhile, prefix filtering restricts both (a) and (b) but only for stub ASes. We conclude that ISPs should

strive to secure the routing system through a combination of prefix filtering and origin/topology/path validation.

Partial Deployment

Thus far we have completely overlooked one crucial fact—that, in practice, it may take decades before a given routing security solution is fully deployed by every AS in the Internet. We expect instead most routing protocols to exist for years in a state of partial deployment, where some ASes deploy the secure protocol but others do not. Indeed, prefix filtering has been partially deployed for several decades,^{9,35} and the origin validation with the RPKI has been partially deployed since the start of this decade. We now quantify the “security” of various defenses in partial deployment.

Prefix filtering in partial deployment. Why is prefix filtering not yet fully deployed? Prefix filtering is implemented solely at the discretion of each individual ISP, and there is no way for one AS to validate that another AS has properly implemented prefix filtering. This is in stark contrast to cryptographic protocols^{18–20,40} (for path/topology/origin validation) that allow any AS that deploys the protocol to validate routes announced by other ASes that have deployed the protocol. Moreover, an ISP deploying prefix filtering derives little local benefit for itself. Instead, it altruistically protects the rest of the Internet from attacks. This leads to lopsided incentives for deployment. We therefore analyze prefix filtering in partial deployment, where some providers filter BGP announcements from their stub customers, but others do not.

Attacks by a given stub are thwarted only if all its providers implement prefix filtering. What happens when only the k largest providers filter announcements from their stub customers? It follows that an attack by a stub will fail if and only if that stub’s smallest provider implements prefix filtering. Figure 4 is a pie chart that breaks up stubs by the size of their smallest provider. Figure 4 shows only 85% of the pie, since the other 15% of ASes are non-stubs. We see that if all providers with more than 500 customers were to implement prefix filtering, then attacks by 13.8% of ASes would be eliminated (the lightest gray slice of the pie). This translates to just 14 providers imple-

menting prefix filtering. If all providers with more than 25 customers filter (corresponding to approximately 422 of the 6,092 providers in our topology), then the fraction of ASes that can attack drops by almost half ($48.4\% = 13.8\% + 15.0\% + 19.6\%$).

Takeaways. We focused on filtering stubs. Our results could hence underestimate the efficacy of prefix filtering, because, in practice,⁹ some providers use even more rigorous prefix filters that filter their entire customer cone. We conclude that prefix filtering, even by just few large ISPs, effectively prevents attacks on BGP.

Topology/path validation in partial deployment. We found little difference between the efficacy of topology validation and path validation. So, from now on, we treat the two as interchangeable. We also found that the combination of prefix filtering with origin/topology/path validation provides the best protection against routing attacks, and that prefix filtering is useful even when partially deployed. This suggests that prefix filtering should be deployed in combination with any cryptographic BGP security protocol. However, deployment of origin validation (with, say, the RPKI²⁰) is already a significant challenge for ISPs,⁷ and any topology or path validation protocol necessarily incorporates one for origin validation as well. Is it really worthwhile for ISPs to deploy topology/path validation on top of origin validation with prefix filtering? To answer, we assume a future scenario where both prefix filtering and origin validation are fully deployed, and the remaining challenge is adoption of topology/path validation. We say an AS is an “adopter” if it deploys topology or path validation (on top of origin validation with prefix filtering); a “non-adopter” AS uses only origin validation with prefix filtering.

Our partial-deployment threat model. Our goal is to quantify the security benefits obtained from a set S of adopters of path/topology validation. As discussed earlier, we use simulations to measure the average percentage of ASes in the topology that are safe (do not select a route through attacker m) when m attacks a victim AS d . We then average over all pairs (m, d) of victim AS d and non-stub attacker m . (The attacker must be a non-stub, since we assume prefix filtering is

fully deployed.) But what does it mean for an attacker to launch an attack in our partial-deployment scenario? Since origin validation and prefix filtering are already fully deployed, we need consider only the naive attack that slips past the combination of these protocols—one-hop hijacks by non-stub attackers, as discussed earlier.

Attacking adopters? Naturally, non-adopter ASes (that have not deployed topology/path validation) can fall victim to one-hop hijacks launched by non-stub attackers. But what about adopters? To answer, we must first define the notion of a “secure route.” A secure route is a route for which every AS on the route is an adopter, whereas an insecure route contains at least one non-adopter AS. We use this definition because BGPSEC (respectively, soBGP) fails to achieve path (respectively, topology) validation for a path if at least one AS on the path is a non-adopter. Moreover, even an adopter must sometimes select an insecure route; if an adopter never selected an insecure route, it would lose connectivity to ASes that cannot be reached via secure routes. But can an adopter that does have a secure route to a destination still be affected by a one-hop hijack? Unfortunately, it can be so affected in the following way:

Figure 5 shows how AS 21740 could select a bogus route during a one-hop hijack by AS m of IP prefix 4.0.0.0/8. Under normal conditions, AS 21740 has a secure provider route directly to 4.0.0.0/8. Note that AS 21740 does not have a peer route via AS 174;

AS 174’s export policy **Ex** prevents it from announcing, to AS 21740, its peer route to AS 3356. During the attack, m announces it is directly connected to AS 3356, so AS 21740 sees a bogus insecure four-hop peer route via its peer AS 174. Importantly, AS 21740 has no idea this route is bogus, as it looks like any other route that might be announced with legacy BGP. What should AS 21740 do? It could select the expensive secure provider route, deciding the added security is more important than the added cost of the provider route. On the other hand, it could decide security is not worth the added cost, and thus fall victim to the attack by choosing the cheaper but insecure peer route.

Secure routing policy model. Why would an adopter prefer an insecure

Figure 4. Distribution of stubs, ordered according to the size of their smallest provider.

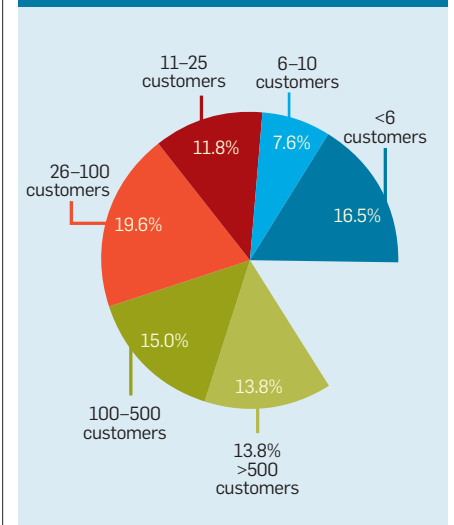
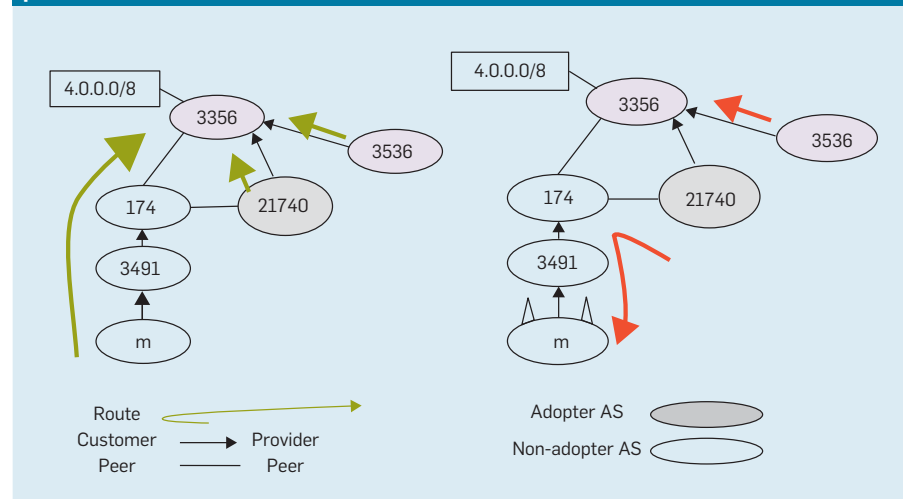


Figure 5. Attacking adopter ASes: (left) normal conditions; (right) when m launches a one-hop hijack and AS 21740 prefers insecure peer routes over (expensive) secure provider routes.



route over a secure route? This can happen because economics and performance often outweigh security concerns. During partial deployment, network operators are expected to cautiously incorporate security into routing policies, placing it after the **LP** and **SP** steps (see the routing policy model discussed earlier) to avoid disruptions due to changes in the traffic through their networks and revenue lost when expensive secure routes are chosen instead of revenue-generating customer routes. Security may be only the top priority once these disruptions are absent (such as in full deployment) or to protect highly sensitive IP prefixes. Our model thus assumes every adopter will add the following step to its routing policy between the **SP** and **TB** steps:

SecP. Prefer a secure path over an insecure path. Placing the **SecP** step after the **LP** and **SP** steps means both economics and performance supercede security. A survey of 100 network operators¹² found that the majority (over 57%) of those that opted to answer this question would rank security this way.

Which ASes should adopt? To quantify the security benefits of topology/path validation in each of these three routing policy models, we must first decide which set of ASes to consider as adopters. In Lychev et al.²³ we showed we can gain valuable insights even by completely sidestepping the question. Our framework is based on a key observation: For each attacker-destination pair (m, d) , it is possible to partition ASes into three distinct categories based on their position in the AS graph:

Doomed. Some ASes are doomed to route through the attacker regardless of which ASes are adopters. AS 174 in Figure 5 is doomed, as it always prefers the bogus customer route to the attacker m over a (possibly secure) peer path to 4.0.0.0/8 for every possible choice of set of adopters;

Immune. Other ASes are immune to the attack regardless of which ASes are adopters. AS 23520 in Figure 1 is immune to attacks, as its one-hop customer route to 72.252.8.0/21 is always more attractive than the one-hop hijack path offered by attacker m , regardless of which ASes are adopters; and

Protectable. Only the remaining ASes are protectable, or whether or not they route through the attacker depends on which specific ASes are adopters.

Prefix filtering should therefore be deployed in combination with any cryptographic BGP security protocol.

We leverage this observation as follows. Recall that we quantify security benefits by computing the average fraction of safe ASes over pairs of attacker m and victim d . We thus get a lower bound on the average fraction of safe ASes for all possible sets of adopters by averaging the fraction of immune ASes over (m, d) pairs. This follows because immune ASes are always safe, regardless of which ASes are adopters. Likewise, we get an upper bound by computing the average fraction of ASes that are not doomed.

The marginal benefits of path/topology validation. We now quantify the marginal benefits of topology/path validation over just origin validation with prefix filtering. We compute the average fraction of immune, protectable, and doomed source ASes, averaged over all possible pairs of destinations and non-stub attackers in the AS graph. We find that 53% of ASes are immune, 30% of ASes are doomed, and only 17% of ASes are protectable. This means the average fraction of safe ASes is already 53%, even if there are no adopters of topology/path validation. Furthermore, on average, only 17% of ASes can additionally be made safe if topology/path validation is deployed in addition to origin validation and prefix filtering. Thus 17% represents the maximum gain of topology/path validation; in realistic partial deployment scenarios, where some ASes are adopters, but others are not, we find this gain is even smaller. The online appendix has more discussion of these results, including an analysis of other routing-policy models.

Takeaways. Given the routing policies that operators favor most during partial deployment, in practice, it will be difficult to realize the security benefits of topology/path validation over those already provided by origin validation and prefix filtering.

Conclusion

The aggregate trends revealed through our quantitative analysis can be used to inform the debate about which BGP security protocols should be deployed in the Internet.^f

^f Because we work with inferred AS-level topologies and a model of routing policies, we caution against interpreting our results as hard numbers that measure the impact of an attack launched by a specific attacker in the wild.

First, we find that prefix filtering,⁹ a simple whitelisting technology available in most BGP-speaking routers today, provides a defense comparable to that provided by the cryptographic technologies that have been developed by standards bodies for the past two decades. Prefix filtering should therefore be deployed in combination with any cryptographic BGP security protocol. Prefix filtering is also useful even when it is partially deployed by only a few tens or hundreds of large ASes. Second, we find that partial deployment has been a blind spot in our discussions of routing security. In full deployment, robust cryptographic security guarantees like path validation (such as BGPSEC¹⁹) or topology validation (such as soBGP⁴⁰) unquestionably provide more protection against attacks than weaker guarantees like origin validation (such as the RPKI²⁰). These more robust security guarantees come at the cost of higher overheads. However, when path or topology validation technologies are partially deployed, our results indicate they provide limited security benefits over what is already provided by origin validation and prefix filtering. The routing community should therefore aggressively deploy prefix filtering and origin validation and focus its energy on the non-trivial operational³⁵ and policy issues, especially those related to trust and liability for information in RPKI certificates^{7,10} that must be addressed before these technologies can be fully deployed.

Acknowledgments

This article is a new synthesis of earlier research we published in Goldberg et al.¹⁴ and Lychev et al.²³ Our related research was supported, in part, by the National Science Foundation grants (1017907, 1350733), Microsoft Research, and Cisco. We thank Alison Kendler, Jeff Lupien, and Paul Oka for outstanding research assistance, as well as our other research collaborators on work that has shaped the discussion here: Kyle Brogle, Danny Cooper, Yossi Gilad, Phillipa Gill, Shai Halevi, Ethan Heilman, Pete Hummon, Aanchal Malhotra, Leonid Reyzin, Jennifer Rexford, and Tony Tauber.

References

- Anwar, R., Niaz, H., Choffnes, D., Cunha, I., Gill, P., and Bassett, E.-K. Investigating interdomain routing policies in the wild. In *Proceedings of the Internet Measurement Conference* (Tokyo, Japan, Oct. 28–30). ACM Press, New York, 2015.
- Arnbak, A. and Goldberg, S. Loopholes for circumventing the Constitution: Unrestrained bulk surveillance on Americans by collecting network traffic abroad. *Michigan Telecommunications and Technology Law Review* 317 (2015); <http://repository.law.umich.edu/mttlr/vol21/iss2/3>
- Boldyreva, A. and Lychev, R. Provable security of S-BGP and other path vector protocols: Model, analysis and extensions. In *Proceedings of the 19th ACM Conference on Computer and Communications Security* (Raleigh, NC, Oct. 16–18). ACM Press, New York, 2012, 541–552.
- Brown, M.A. *Pakistan hijacks YouTube*. Dyn Research blog, Feb. 2008; <http://research.dyn.com/2008/02/pakistan-hijacks-youtube-1/>
- Butler, K., Farley, T., McDaniel, P., and Rexford, J. A survey of BGP security issues and solutions. *Proceedings of the IEEE* 98, 1 (2010), 100–122.
- Chi, Y.-J., Oliveira, R., and Zhang, L. Cyclops: The Internet AS-level observatory. *ACM SIGCOMM Computer Communication Review* 38, 5 (2008), 5–16.
- Cooper, D., Heilman, E., Brogle, K., Reyzin, L., and Goldberg, S. On the risk of misbehaving RPKI authorities. In *Proceedings of HotNets XII, the 12th ACM Workshop on Hot Topics in Networks* (College Park, MD, Nov. 21–22). ACM Press, New York, 2013.
- Cowie, J. China's 18-minute mystery. Dyn Research blog, Nov. 2010; <http://research.dyn.com/2010/11/chinas-18-minute-mystery/>
- Durand, J., Pepelnjak, I., and Doering, G. *RFC 7454: BGP Operations and Security*. Internet Engineering Task Force, 2015; <http://tools.ietf.org/html/rfc7454>
- Gallo, A. *RPKI: BGP Security Hampered by a Legal Agreement*. Packetpushers blog, Dec. 2014; <http://packetpushers.net/rpki-bgp-security-hampered-legal-agreement/>
- Gao, L. and Rexford, J. Stable Internet routing without global coordination. *IEEE/ACM Transactions on Networking* 9, 6 (2001): 681–692.
- Gill, P., Schapira, M., and Goldberg, S. A survey of interdomain routing policies. *ACM SIGCOMM Computer Communication Review* 44, 1 (2013), 28–34.
- Giotas, V., Luckie, M., Huffaker, B., and Claffy, K. IPv6 AS relationships, cliques, and congruence. In *Proceedings of the International Conference on Passive and Active Network Measurement* (New York, Mar. 19–20). Springer International Publishing, 2015, 111–122.
- Goldberg, S., Schapira, M., Hummon, P., and Rexford, J. How secure are secure interdomain routing protocols? In *Proceedings of ACM SIGCOMM'10 Conference* (New Delhi, India, Aug. 30–Sept. 3). ACM Press, New York, 2010, 87–98.
- Goodin, D. Hacking team orchestrated brazen BGP hack to hijack IPs it didn't own. *Ars Technica* (July 12, 2015); <http://arstechnica.com/security/2015/07/hacking-team-orchestrated-brazen-bgp-hack-to-hijack-ips-it-didnt-own/>
- Griffin, T. and Huston, G. *RFC 4264: BGP Wedgies*. Internet Engineering Task Force, 2005; <http://tools.ietf.org/html/rfc4264>
- Huston, G. Peering and settlements - Part I, II. *The Internet Protocol Journal* 2, 1 (Mar. 1999).
- Kent, S., Lynn, C., and Seo, K. Secure Border Gateway Protocol (S-BGP). *IEEE Journal on Selected Areas in Communications* 18, 4 (Apr. 2000), 582–592.
- Lepinski, M. *draft-ietf-sidr-bgpsec-protocol-14: BGPSEC Protocol Specification*. Internet Engineering Task Force, 2015; <https://tools.ietf.org/html/draft-ietf-sidr-bgpsec-protocol-14>
- Lepinski, M. and Kent, S. *RFC 6480: An Infrastructure to Support Secure Internet Routing*. Internet Engineering Task Force, 2012; <http://tools.ietf.org/html/rfc6480>
- Litke, P. and Stewart, J. *BGP Hijacking for Cryptocurrency Profit*. Dell SecureWorks Counter Threat Unit, Aug. 7, 2014; <http://www.secureworks.com/cyber-threat-intelligence/threats/bgp-hijacking-for-cryptocurrency-profit/>
- Lychev, R. *Evaluating Security-Enhanced Interdomain Routing Protocols in Full and Partial Deployment*. Ph.D. thesis, Georgia Tech, Atlanta, GA, 2014; <https://smartech.gatech.edu/handle/1853/52325>
- Lychev, R., Goldberg, S., and Schapira, M. BGP security in partial deployment: Is the juice worth the squeeze? In *Proceedings of the SIGCOMM'13 Conference* (Hong Kong, China, Aug. 12–16). ACM Press, New York, 2013, 171–182.
- Madory, D. *Global Collateral Damage of Tmnet Leak*. Dyn Research blog, June 12, 2015; <http://research.dyn.com/2015/06/global-collateral-damage-of-tmnet-leak/>
- Mao, Z., Rexford, J., Wang, J., and Katz, R.H. Towards an accurate AS-level traceroute tool. In *Proceedings of the SIGCOMM'03 Conference* (Karlsruhe, Germany, Aug. 25–29). ACM Press, New York, 2003, 365–378.
- McDaniel, P., Aiello, W., Butler, K., and Ioannidis, J. Origin authentication in interdomain routing. *Computer Networks* 50, 16 (2006), 2953–2980.
- McPherson, D., Amante, S., Osterweil, E., and Mitchell, D., Eds. *Route-Leaks & MITM Attacks Against BGPSEC*. Internet Draft, ETF Network Working Group, Nov. 18, 2013; <http://tools.ietf.org/html/draft-ietf-grow-simple-leak-attack-bgpsec-no-help-03>
- Misel, S. Wov. AS7007! Merit NANOG Archive, Apr. 1997; https://www.nanog.org/maillinglist/mailarchives/old_archive/1997-04/msg00340.html
- National Institute of Standards and Technology. RPKI Deployment Monitor, Gaithersburg, MD; <http://www.x.antd.nist.gov/rpki-monitor/>
- Paseka, T. *Why Google Went Offline Today and a Bit about How the Internet Works*. Cloudfare blog, Nov. 6, 2012; <https://blog.cloudflare.com/why-google-went-offline-today-and-a-bit-about/>
- Peterson, A. Researchers say U.S. Internet traffic was re-routed through Belarus. *The Washington Post* (Nov. 20, 2013); <https://www.washingtonpost.com/news/the-switch/wp/2013/11/20/researchers-say-u-s-internet-traffic-was-re-routed-through-belarus-thats-a-problem/>
- Pilosov, A. and Kapela, T. Stealing the Internet: An Internet-scale man-in-the-middle attack. In *DEFCON* (Las Vegas, NV, Aug. 8–10, 2008).
- Ramachandran, A. and Feamster, N. Understanding the network-level behavior of spammers. *ACM SIGCOMM Computer Communication Review* 36, 4 (Sept. 2006), 291–302.
- Shaw, A. Spam? Not Spam? Tracking a hijacked Spamhaus IP. *Greenhost*, Mar. 21, 2013; <https://greenhost.nl/2013/03/21/spam-not-spam-tracking-hijacked-spamhaus-ip/>
- Steenbergen, R., Volk, R., Kumari, W., Blunk, L., and McPherson, D. ISP route filtering: Responsibilities & technical challenges. In *NANOG'43 North American Network Operators' Group Conference* (Brooklyn, NY, June 1–4, 2008).
- Sun, Y., Edmundson, A., Vanbever, L., Li, O., Rexford, J., Chiang, M., and Mittal, P. RAPTOR: Routing Attacks on Privacy in Tor. In *Proceedings of the 24th USENIX Security Symposium* (Washington, D.C., Aug. 12–14). USENIX Society, Berkeley, CA, 2015, 11–20.
- Toonk, A. Massive route leak causes Internet slowdown. *BGPmon* blog, June 12, 2015; <http://www.bgpmon.net/massive-route-leak-cause-internet-slowdown/>
- Underwood, T. Con-Ed Steals the 'Net. Dyn Research blog, Jan. 2006; <http://research.dyn.com/2006/01/coned-steals-the-net/>
- Vervier, P.-A., Thonnard, O., and Dacier, M. Mind your blocks: On the stealthiness of malicious BGP hijacks. In *Proceedings of the NDSS'15 Network and Distributed System Security Symposium* (San Diego, CA, Feb. 8–11). Internet Society, Reston, VA, 2015.
- White, R. *Deployment Considerations for Secure Origin BGP (soBGP)*. IETF Internet Draft, Network Working Group, June 25, 2003; <https://datatracker.ietf.org/doc/draft-white-sobgp-bgp-deployment/>

Robert Lychev (robert.lychev@mit.edu) is a technical staff member in the Cyber Analytics and Decision Systems Group at MIT Lincoln Laboratory, Cambridge, MA.

Michael Schapira (schapiram@cs.huji.ac.il) is an associate professor in the School of Computer Science and Engineering at the Hebrew University of Jerusalem, Israel, and the co-scientific leader of the Fraunhofer Cybersecurity Center at Hebrew University.

Sharon Goldberg (goldbe@cs.bu.edu) is an associate professor in the Computer Science Department of Boston University, Boston, MA, and a faculty fellow of Boston University's Hariri Institute for Computing.

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

DOI:10.1145/2896816

The most important consideration is how the collection of measurements may affect a person's well-being.

BY CRAIG PARTRIDGE AND MARK ALLMAN

Ethical Considerations in Network Measurement Papers

NETWORK MEASUREMENT—BECAUSE it is typically at arm's length from humans—does not comfortably fit into the usual human-centered models for evaluating ethical research practices. Nonetheless, the network measurement community increasingly finds its work potentially affects humans' well-being and itself poorly prepared to address the resulting ethical issues. Here, we discuss why ethical issues are different for network measurement versus traditional human-subject research and propose requiring measurement papers to include a section on ethical considerations. Some of the ideas will also prove applicable to other areas of

computing systems measurement, where a researcher's attempt to measure a system could indirectly or even directly affect humans' well-being.

A conference program committee (PC) is usually the first outside independent organization to evaluate research work that measures network systems. In recent years, questions about whether the work submitted follows sound ethical practices have become increasingly common within PC discussions. We have experience with this situation as researchers and as members and leaders of PCs struggling with ethical concerns.

The fundamental cause of this struggle is that the network measurement community lacks a set of shared ethical norms. Historically, measurements of computing and communications systems have not been viewed as affecting humans to a degree that they require ethical review. Indeed, ethics review boards often declare measurement research "exempt" from full review as not involving human subjects; Burnett and Feamster⁴ described such an experience in 2015. Beyond the need to protect the privacy of communications content, researchers lack consensus about how to ethically handle even the most basic situations. Authors often work from one set of ethical notions, while a PC applies one or more different sets of ethical notions as part of its review. This divergence leaves well-meaning researchers—in all roles—on fundamentally different pages. The situation is further exacerbated because the network measurement community lacks a culture of de-

» key insights

- The network measurement community increasingly finds itself ill-equipped to deal with the implications of the potential effects its experiments can have on human well-being.
- We aim to minimize the risk of inflicting harm.
- We advocate exposing ethical thinking through a published "ethical considerations" section included in all empirically based papers.



scribing the ethical reasoning behind a set of experiments. It also leaves PCs to infer the ethical foundations on which a paper is based, while precautions taken by careful researchers are not exposed to others who may leverage or build on previous techniques in subsequent work.

In this article, we advocate requiring an “ethical considerations” section in measurement papers as a first step toward addressing these issues. By requiring such a section—even if the result is a statement that there are no ethical issues—we provide the starting point for a discussion about ethics in which authors have a chance to justify the ethical foundations of their experimental methodologies and PC members can review the authors’ perspective and provide specific feedback, as necessary. Further, by including these sections in published papers, the entire research community would begin to develop a collective understanding of both what is ethically acceptable and how to think through ethics issues.^a

Our aim here is to present an initial straw man for discussing ethics. We do not attempt to prescribe what is and what is not ethical. We do not tackle all possible ethical questions that arise in our work as Internet empiricists. Rather, we advocate for a framework to help the measurement research community start an explicit conversation about the largest ethical issues involved in measuring networked systems (such as the Internet, cloud computing systems, and distributed transactions systems).

Background

Three strands of intellectual activity come together when examining ethics and network measurement.

Evolution of the field of ethics. The study of ethics in computing has evolved as the capabilities of computer systems have evolved.

Evolution of our ability to extract information from measurement data. Developing an empirical understanding of network behavior has been a pillar of network research since its earliest

days. The area has steadily improved, refining its tools to extract ever more information from measurements, such that longtime assumptions about what information can be extracted from a measurement often no longer hold.

The law. The legal issues surrounding network measurement are at best murky in any single jurisdiction,¹⁵ since there is little case law to establish how courts will interpret communication systems law within the context of modern data networks; such issues multiply when a measurement study crosses (many) jurisdictions. We encourage researchers to consult their local counsel when legal questions arise. However, for the purposes of this article our focus is on ethical issues; we mention legal issues only when they help illuminate the ethical issues.

Ethics. The study of ethics in information and communication science has broadly followed two (overlapping) lines of inquiry. The first focuses on human-centered values (such as life, health, security, and happiness). This thinking originated in the 1940s with Norbert Weiner and has been carried down to the present. A more recent expression is the 2012 *Menlo Report* by the U.S. Department of Homeland Security,⁷ focusing on potential harm to persons, either through revealing confidential information or altering their environment and ensuring the risks of harm from the experiment are recognized, moderated, and equitably distributed (such as in seeking to ensure that those persons whose environment is altered by the experiment are also persons who are likely to benefit from the experimental results).

The second line of ethical thinking has focused on the professional responsibility of the computing and information sciences professional. The focus has been on following good industry practices in the creation of software artifacts and codes of conduct that outline a professional’s responsibilities to society, employers, colleagues, and self. A detailed expression of this thinking is the joint *IEEE/ACM Software Engineering Code of Ethics and Professional Practice*, which identifies eight principles and presents more than 80 distinct admonitions.¹

Both approaches concern the effect

of one’s work on other humans and systems that directly interface with humans.

Network measurements, and many other types of system measurement, are usually at least one step removed from directly interfacing with humans. Intuitively, probing a network or counting hits in a cache does not affect humans. Nonetheless, measurement work can affect humans, and we focus here on measurements where the effect, however indirect, can be envisaged. This focus means we do not discuss ethical issues where the harm, to first order, might come to vendors, systems, or intellectual-property-rights owners.

Evolution of network measurement. The field of network measurement—broadly defined—is relatively old. As best we can tell, beginning with the electronic telegraph, all networks have been the subject of various forms of measurement. We briefly trace the evolution of the field both technically and in a legal and ethical context and finish with some observations.

Technical evolution of measurement. Measuring a communications network and analyzing the results has been a staple of (data) communications research from its inception. By 1911, AT&T had a statistical group that, among other functions, leveraged measurement to better engineer the telephone system and predict demand. When the ARPANET (forerunner of the Internet) was turned on in 1969, its first node was installed at UCLA for the use of Leonard Kleinrock’s measurement group.

Measurement can be passive or active. Passive measurement simply observes in-situ traffic. Active measurement injects new traffic to observe a system’s response. Given that networks are digital systems, built according to standards, one might imagine that examining network traffic (passively or actively) is largely an exercise in detecting bugs. In reality, the interactions of traffic in networks give rise to complex patterns. Furthermore, because the communications infrastructure is distributed, the interaction of delays between components and routine failures can lead to complex phenomena. Finally, variations in how specifications are implemented can lead to interesting interactions.

^a We recognize that limiting the public view of the ethics discussions between authors and PCs to published papers is imperfect, as it limits the ability to build on ethics failures, but it will provide a foundation of good ethics work to build upon.

Examples of important research results from passive monitoring include methods for ensuring sequence numbers are robust against device crashes,¹⁷ the discovery of self-similarity in network traffic,¹¹ and methods to avoid the self-synchronization of network traffic.⁸ Examples from active probing include measurements to develop the Network Time Protocol (which keeps distributed clocks synchronized)¹² and the study of network topology.²¹

Ethics and law of measurement. Much of our legal, social, and ethical dialog about network measurement uses legal terminology that was developed in the early days of measurement. Specifically, the ethics and legality of network measurements are often evaluated with the implicit assumption that the only parties allowed to capture data outside a workplace campus are communications companies providing service and government agencies given access to communications companies' data centers; see, for instance, the U.S. Code.¹⁹ Further, a typical formulation distinguishes between two classes of data, as follows.

The first class of data reveals when and for how long two parties communicated. U.S. law defines a device capable of capturing such data as a "pen register." More recently, the term "metadata" has been used to describe an expanded set of information, including packet headers. The U.S. government has suggested metadata is comparable to pen register data.²⁰

The second class of data reveals the contents of the conversation. To highlight the distinction, consider a phone call to a bank. A pen register records that a call took place at a specific time and for a specific duration. The contents of the conversation would reveal that the call was a balance inquiry. U.S. law has recognized, since 1967, that the content of a conversation is a distinct class of information that has a higher expectation of privacy,¹⁸ and this distinction between content and metadata is often carried over into ethical discussions.

Metadata is becoming content. A variety of factors has eroded the distinction between content and metadata. Specifically, researchers' ability to leverage metadata to infer—or even re-create—content is increasing rapidly.



Strictly speaking, active measurements have the potential to inflict direct and tangible harm.



Several examples illustrate this point:

Traffic tables. Measuring when devices in a network transmit is sufficient to derive traffic tables that can distinguish routers from end systems and identify what nodes are communicating with each other.⁵

Behavior of queues. The Queue Inference Engine takes information about transactions (such as pen register style data) and reverse engineers it to determine the behavior of queues.¹⁰ Researchers have made steady progress in using techniques like the Queue Inference Engine to characterize queues from metadata. For instance, a researcher can tell whether and for approximately how long a person likely waited in line at a bank ATM machine by tracking when transactions at the machine started and ended.²

Gaps between transmissions. Inter-packet gaps (a form of metadata) between encrypted transmissions can help infer where users' fingers were on the keyboard and thus give guidance about what letters are in their passwords.¹⁶

Packet headers. In some cases, it is possible to determine what words are being said in an encrypted voice conversation simply by looking at packet headers.²²

That is, with less data than a pen register would collect, a researcher is often able to determine that the call to a bank was, say, a balance inquiry. Furthermore, users and researchers alike should expect the distinction between metadata to data to continue to erode over time.

The Contours of Harm

While myriad ethical issues confront researchers conducting network measurements, our aim here is to address those causing tangible harm to people. We are not concerned with notions of potential harm to network resources (such as bandwidth) or equipment, except insofar as the effect on resources and equipment could cause tangible harm to a human. How a researcher's work affects individual human beings is the most important ethical issue.

Additionally, our goal, which mirrors the *Menlo Report*, is not to eliminate the possibility of harm within experiments. Rather, we aim to mini-

mize the risk of inflicting harm. In this context we make several observations bearing on how researchers should manage risk in their experiments:

A spectrum of harm. “Harm” is difficult to define. Rather than a precise definition we offer that a single probe packet sent to an IP address constitutes at most slight harm.^b Meanwhile, a persistent high-rate series of probes to a given IP address may well be viewed as both an attack and cause serious harm, as in unintentionally clogging a link precisely when the link is needed for an emergency. These ends of the spectrum are useful as touchstones when thinking about how to cope with the risk involved in specific experiments.

Indirect harm. We also recognize that the field of network measurement focuses on (for the most part) understanding systems and not directly assessing people. Any effect on people is thus a side effect of a researcher’s measurements. While researchers must grapple with the ethics of harm due to their measurements regardless of whether the harm is direct or indirect, the nature of the harm can sometimes dictate the manner in which researchers cope.

Potential harm. Note most often the research does not cause harm but rather only sets up the possibility of harm. That is, additional events or factors beyond the measurements must happen or exist for actual harm to be inflicted. Again, this does not absolve researchers from understanding the ethical implications of their experiments but does speak to how they may manage the risk involved in conducting a particular experiment.

While fuzzy, these aspects of “harm” present the broad contours of the issues with which researchers must grapple. Further, there is no one-size-fits-all way to manage harm, and we encourage honest disagreement among researchers about when potential and indirect harm rises to the level of making an experiment problematic. For instance, in the context of the example described earlier about probes causing slight vs. serious harm, we privately discussed whether periods of high-rate

^b We have experience with complaints about such probes, indicating some people do, in fact, view them as harmful.

Direct consent is not possible in most Internet measurements; the community of measurement researchers thus needs to cope with ethical challenges without relying on consent.

transmissions could be made short enough to reasonably be felt to avoid potential harm. We agreed it was possible but disagreed about when the experiment transitioned from slight harm to potentially serious harm.

Collecting Data

Strictly speaking, active measurements have the potential to inflict direct and tangible harm. Passive measurements, by their nature, are simply recordings of observations and in no way directly change—benignly or harmfully—the operation of the network. Likewise, downloading and (re)using a shared dataset does not alter a network’s operation—even if collecting it in the first place did. Previously collected data brings up thorny issues of the use of so-called “found data.” For instance, consider the Carna botnet,³ which leveraged customer devices with guessable passwords allowing illicit access and was used to take measurements that were publicly released. If a paper submission’s methodology section were to say, “We first compromised a set of customer devices,” the paper would likely be rejected as unethical, and probably illegal. However, if, instead, a researcher simply downloaded this data—causing no further harm to the customer devices or their users—would it be ethical to use it as part of one’s research?

On the one hand, a researcher can make the case that any harm done by collecting the data has already transpired and thus, by simply downloading and using it, the researcher is, in fact, causing no harm. Further, if the data can provide insights into the network, then perhaps the research community can view using the data as making the best of a bad situation. Alternatively, the research community could view the use of such data as a moral hazard.

The use of data whose collection was objectionable is an open one in the medical community, as in Mostow.¹³ The measurement community needs to find its own answers. There are likely different answers for different situations. For instance, a public dataset obtained through unethical means (such as the 2015 leak of the Ashley Madison dating website dataset) may be viewed differently from a non-public dataset that happens to have been leaked to a

researcher. The research community may view the first case as less problematic because of the reach of the data release, whereas in the latter case the community may decide the researcher is more culpable because, if not for the researcher's work, less would be known about the (potentially harmful) dataset. We encourage researchers to be thoughtful about the ethical issues related to the sources of their data.

Storing Data

The measurement community generally encourages the preservation of measurement data to facilitate revisiting it in response to questions or concerns during the initial work, to look at new research questions later, or to facilitate historical comparisons. Furthermore, the community encourages researchers to make their data public to better enable access to other researchers, as in CAIDA's Data Catalog (DatCat; <http://datcat.org/>), a National Science Foundation-sponsored repository of measurement data. Preserving and publishing measurement data raises a number of ethical issues; we highlight two in the following paragraphs.

First, how does a researcher determine if a dataset can ethically be made public? There are plenty of examples of successful data de-anonymization.¹⁴ As discussed earlier, a researcher's ability to extract information from seemingly innocuous data continues to improve. As an example, datasets published in the 1980s and early 1990s could likely be mined for passwords using packet-timing algorithms published in 2001.^{16,c}

Second, if the data cannot be made public, but is retained, what safeguards does the community expect the researcher to implement to avoid accidental disclosure? For instance, should the community expect all data stored on removable media to be encrypted? Should the data also be encrypted on non-removable disks? Should the rules vary according to the perceived sensitivity of the data?

It is not reasonable to expect researchers to anticipate all future analysis advances. However, it is reasonable to expect researchers to

understand how current techniques could exploit their measurement data and expect them to provide appropriate safeguards.

On the Limitations of Consent

One traditional way of dealing with ethical issues in experiments is to require (informed) consent from participants. This approach allows the people who could potentially be harmed by an experiment to weigh those potential harms against the potential benefits and directly decide whether or not to participate. In some cases, Internet measurement can (and does) leverage consent. For instance, the Netylizr measurement platform⁹ aims to assess a user's Internet connection by providing a webpage the user must purposefully access. Further, the webpage spells out what will happen and requires the user to explicitly start the measurements—hence consenting.

The Netylizr situation is akin to the consent model in more traditional areas (such as medicine) and works well. However, in other settings, obtaining informed consent for large-scale Internet measurements is significantly more difficult. Consider a study of end-user networks that uses a different methodology from the one in Netylizr. Dischinger et al.⁶ used various tests to probe IP addresses they believed represented home networks unbeknownst to the users. These tests provided a large-scale dataset that Netylizr cannot match but without the consent of the potentially affected people.^d

Consent in Internet-scale measurements is difficult for two reasons. First, unlike, say, medical experiments, it is often unclear who is being measured and affected by Internet measurements. Further, even if the affected human participants could be identified, the logistics of obtaining consent range from significantly difficult to impossible.

In more traditional areas of experimentation involving humans, proxy consent is generally not allowed, but in network measurements we lean on this mechanism. For instance, network measurements taken on a university

campus typically seek consent from the university. However, probes sent off-campus might affect third parties with no connection to the university. While proxy consent can thus foster useful review to help identify and mitigate ethical issues, some potentially affected users are not covered directly or represented by an advocate.

There are thus cases where Internet measurements can leverage consent, and we encourage researchers to do so in these situations. However, direct consent is not possible in most Internet measurements; the community of measurement researchers thus needs to cope with ethical challenges without relying on consent.

Proposal: An Ethics Section

Measurement researchers lack norms or examples to guide them. Our position is twofold: as a community we are not able to prescribe ethical norms for researchers to follow, and the best starting approach is to expose ethical thinking through a published “ethical considerations” section in all empirically based papers. This approach serves three main goals:

Recognize ethical implications. While some researchers are careful to understand the ethical implications of their work, such care is not universal; the first goal of an “ethical considerations” section is thus to force all authors to publicly examine the ethical implications of their own work.

Give explicit ethical voice. Rather than counting on PCs and editors to impute the ethical foundations of a piece of work, an ethical considerations section will give explicit voice to these issues; reviewers will be able to directly evaluate the stated ethical implications of the work and give concrete feedback to the authors, grounded in the authors' own approach.

Create public examples of good ethics. Ethics sections are not usually required by conferences and, if they are, are typically addenda to the paper seen by the PC and not published.

Public ethics sections in papers will foster a conversation among measurement researchers based on published exemplars, leading the community toward norms.

Here, we outline four strawman questions authors should answer in

c One risk is that users from the 1980s and 1990s who are still active today may still pick passwords in similar ways.

d Note our point is about the difficulty of getting consent, not a criticism of the paper; Dischinger et al.⁶ properly sought to minimize the possible harm to users, describing their efforts in their paper.

such an ethical considerations section. We aim for a short list of questions, believing that capturing 80% of the ethics issues is better than a longer list that is still not exhaustive:

For datasets directly collected by the author(s), could the collection of the data in the study be reasonably expected to cause tangible harm to any person's well being? If so, discuss measures taken to mitigate the risk of harm.

For datasets not directly collected by the author(s), is there an ethical discussion of the collection elsewhere? If so, provide a citation. If not, the paper should include a discussion of the ethics involved in both collecting and using the data—beyond simply noting that no additional data-collection harm would occur in reusing the data. This is especially important for non-public datasets.

Using current techniques, can the data used in the study reveal private or confidential information about individuals? If so, discuss measures taken to keep the data protected from inappropriate disclosure or misuse.

Please discuss additional ethical issues specific to the work that are not explicitly covered by one of these questions.

These questions do not intentionally address two important items:

Institutional review board. There is no suggestion of when it might be appropriate to consult an institutional review board or similar body. Furthermore, the involvement of such a body (or its non-involvement) is not a substitute for the measurement community's own ethical review.

Research results. We do not attempt to assess the ethics of the research result. Researchers are committed to advancing knowledge, which, in our view, includes publishing results and techniques that may, if used unethically, cause tangible harm.

Moreover, making ethics a core part of measurement papers will create new challenges for reviewers and PCs alike, including:

Review practices. Review forms likely will have to be updated to ask reviewers to discuss the strengths and weaknesses of the ethics section.

Mechanisms. Various mechanisms will be needed to help reviewers evaluate ethics. Possible mechanisms include ethics guidelines from the pro-

gram chair, ethics training, or simply an ethics teleconference at the start of the reviewing period. Over time, we hope example published papers will help this process.

Clear philosophy. PCs will need a clear philosophy on when papers are rejected based on ethical considerations and when papers with ethical gaps can be accepted, subject to revision. The questions concerning collection, as discussed earlier, will also come up, and PCs will need to find the measurement research community's answer(s).

Finally, what does it mean to reject a research paper on ethical grounds? While some papers may be resurrected by revising its analysis to not use an objectionable dataset, the rejection often means the measurements used to support the paper's research results may have caused harm. That determination may raise questions about how to mitigate the harm and prevent such harm in the future.

Conclusion

We have presented a strawman suggestion that authors of measurement papers include a (short) ethics section in their published papers. Doing so would help identify ethics issues around individual measurement studies in a way that allows PCs to evaluate the ethics of a measurement experiment and the broader community to move toward a common ethical foundation.

Acknowledgments

This article has benefited from many conversations with too many colleagues to name individually. Our thanks to all of them. Bendert Zevenbergen has organized multiple ethics discussions since 2014 in which this work has been discussed. This work is funded in part by National Science Foundation grant CNS-1237265. 

References

1. Association for Computing Machinery and IEEE Computer Society. *Software Engineering Code of Ethics and Professional Practice*, Version 5.2, 1999; <http://www.acm.org/about/se-code>
2. Bertsimas, D. and Servi, L.D. Deducing queueing from transactional data: The Queue Inference Engine revisited. *Operations Research* 40, 3, Supplement 2 (June 1992), 217–228.
3. Botnet, C. *Internet Census 2012*; <http://internetcensus2012.bitbucket.org/paper.html>
4. Burnett, S. and Feamster, N. Encore: Lightweight measurement of Web censorship with cross-origin requests. In *Proceedings of ACM SIGCOMM* (London, U.K., Aug. 17–21). ACM Press, New York, 2015, 653–667.

5. Cousins, D., Partridge, C., Bongiovanni, K., Jackson, A.W., Krishnan, R., Saxena, T., and Strayer, W.T. Understanding encrypted networks through signal and systems analysis of traffic timing. In *Proceedings of the 2003 IEEE Aerospace Conference* (Big Sky, MT, Mar. 8–15). IEEE Press, Piscataway, NJ, 2003, 2997–3003.
6. Dischinger, M., Haeberlen, A., Gummadi, K.P., and Saroiu, S. Characterizing residential broadband networks. In *Proceedings of the Seventh ACM SIGCOMM Conference on Internet Measurement* (San Diego, CA, Oct.). ACM Press, New York, 2007, 43–56.
7. Dittrich, D. and Kenneally, E. *The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research*. U.S. Department of Homeland Security, Washington, D.C., Aug. 2012; https://www.caida.org/publications/papers/2012/menlo_report_actual_formatted/
8. Floyd, S. and Jacobson, V. The synchronization of periodic routing messages. In *Proceedings of SIGCOMM Communications Architectures, Protocols and Applications* (San Francisco, CA, Sept. 13–17). ACM Press, New York, 1993, 33–44.
9. Kreibich, C., Weaver, N., Nechaev, B., and Paxson, V. Netalyr: Illuminating the edge network. In *Proceedings of the 10th ACM SIGCOMM Internet Measurement Conference* (Melbourne, Australia, Nov. 1–3). ACM Press, New York, 2010, 246–259.
10. Larson, R.C. The Queue Inference Engine: Deducing queue statistics from transactional data. *Management Science* 36, 5 (May 1990), 586–601.
11. Leland, W.E., Taqqu, M.S., Willinger, W., and Wilson, D.V. On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Transactions on Networks* 2, 1 (Feb. 1994), 1–15.
12. Mills, D.L. Internet time synchronization: The Network Time Protocol. *IEEE Transactions on Communications* 39, 10 (Oct. 1991), 1482–1493.
13. Mostow, P. Like building on top of Auschwitz: On the symbolic meaning of using data from the Nazi experiments, and on non-use as a form of memorial. *Journal of Law and Religion* 10, 2 (June 1993), 403–431.
14. Schneier, B. Why 'anonymous' data sometimes isn't. *Wired* (Dec. 2007); https://www.schneier.com/essays/archives/2007/12/why_anonymous_data_s.html
15. Sicker, D.C., Ohm, P., and Grunwald, D. Legal issues surround monitoring during network research. In *Proceedings of the Seventh ACM SIGCOMM Conference on Internet Measurement* (San Diego, CA, Oct. 24–26). ACM Press, New York, 2007, 141–148.
16. Song, D.X., Wagner, D., and Tian, Z. Timing analysis of keystrokes and timing attacks on SSH. In *Proceedings of the 10th USENIX Security Symposium* (Washington, D.C.). Usenix Association, Berkeley, CA, 2001.
17. Tomlinson, R. Selecting sequence numbers. In *Proceedings of the ACM SIGCOMM/SIGOPS Interprocess Communications Workshop* (Santa Monica, CA, Mar. 24). ACM Press, New York, 1975, 11–23.
18. United States Supreme Court. *Katz v. United States*. Washington, D.C., 1967; <https://supreme.justia.com/cases/federal/us/389/347/case.html>
19. U.S. Code, 18 Section 3121: *General Prohibition on Pen Register and Trap and Trace Device Use*; <https://www.law.cornell.edu/uscode/text/18/3121>
20. White House. Press gaggle by Deputy Principal Press Secretary Josh Earnest and Secretary of Education Arne Duncan en route Mooresville, NC, June 6, 2013; <https://www.whitehouse.gov/the-press-office/2013/06/06/press-gaggle-deputy-principal-press-secretary-josh-earnest-and-secretary>
21. Willinger, W. and Roughan, M. Internet topology research redux. In *ACM SIGCOMM eBook: Recent Advances in Networking*, ACM Press, New York, 2013; http://sigcomm.org/education/ebook/SIGCOMMBook2013v1_chapter1.pdf
22. Wright, C.V., Ballard, L., Coull, S.E., Monross, F., and Masson, G.M. Uncovering spoken phrases in encrypted Voice Over IP conversations. *ACM Transactions on Information Systems Security* 13, 4 (Dec. 2010), 35:1–35:30.

Craig Partridge (craig@bbn.com) is a chief scientist at Raytheon BBN Technologies, Cambridge, MA, and current chair of the ACM Fellows Committee.

Mark Allman (mallman@icir.org) is a computer scientist in the Network and Security Group of the International Computer Science Institute.

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

Come join us at CSCW 2017!



CSCW
2017
PORTLAND, OR

DoubleTree by Hilton Hotel
February 25 - March 1, 2017

For more information
about the conference and
to register, please visit:
cscw.acm.org/2017/

Facebook: www.facebook.com/acmCSCW/
Twitter: @ACM_CSCW
Sina Weibo: ACM_CSCW (in Chinese)

The ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW) is the premier venue for presenting research in the design and use of technologies that affect groups, organizations, communities, and networks. Bringing together top researchers and practitioners from academia and industry, CSCW explores the technical, social, material, and theoretical challenges of designing technology to support collaborative work and life activities.

The scope of CSCW spans socio-technical domains including work, home, education, healthcare, the arts, leisure, and entertainment. The conference seeks novel research results or new ways of thinking about, studying, or supporting shared activities in these and related areas.

Upcoming Deadlines

Workshop Proposals: *Oct. 17, 2016*

Interactive Posters: *Nov. 4, 2016*

Panels: *Nov. 4, 2016*

Doctoral Colloquium: *Nov. 4, 2016*

Demonstrations: *Nov. 4, 2016*

With the implantation of software-driven devices comes unique privacy and security threats to the human body.

BY A.J. BURNS, M. ERIC JOHNSON, AND PETER HONEYMAN

A Brief Chronology of Medical Device Security

THE CAPABILITIES OF modern medical devices continue to radically transform the treatment of acute conditions as well as the management of chronic long-term disease. As these technologies evolve, so also do the threats to the security and reliability of these devices. Over the past decade, there has been no shortage of headlines warning of pacemaker turned peacemaker, or insulin assassinations. Although these taglines are fictional (but not unimaginable), they capture the tenor of much of the medical device security reportage. While we strongly affirm the necessity of public awareness of these issues, we believe that hyperbole and/or mischaracterizations may lead to panic, desensitization, or perhaps worse, exploitation.

Today, attention is turning to the dangers posed by the omnipresent cyber threat, as signaled with the long-awaited release on Oct. 2, 2014 of Food and Drug Administration (FDA) guidance on the management of cybersecurity in medical devices,⁷ and the more recent draft guidance of Postmarket Management of Cybersecurity in Medical Devices.⁸ Therefore, as the human body joins the illustrious Internet of Things, it is constructive to take pause and see how we got here. We hope this brief chronology of medical device and health IT security helps provide context for the current state of medical device security.

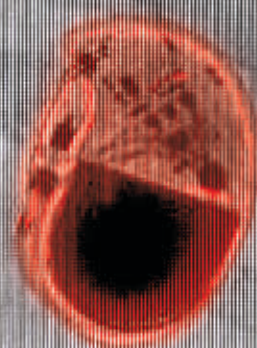
Though not clearly defined, it appears to us there have been several inflection points in the relatively brief history of medical devices. The first period is essentially a spillover from the broader systems engineering field involving concern over complex systems and accidental disasters. The second period begins with the advent of implantable medical devices, and the third with the threat of unauthorized access to these devices that could cause harm. Finally, the fourth and most recent period is the era of the cyber threat to medical device security. Tying all of these together are the implications of software-controlled systems and the threats to device and system security and consequently, patient health and privacy. We also spot-

» key insights

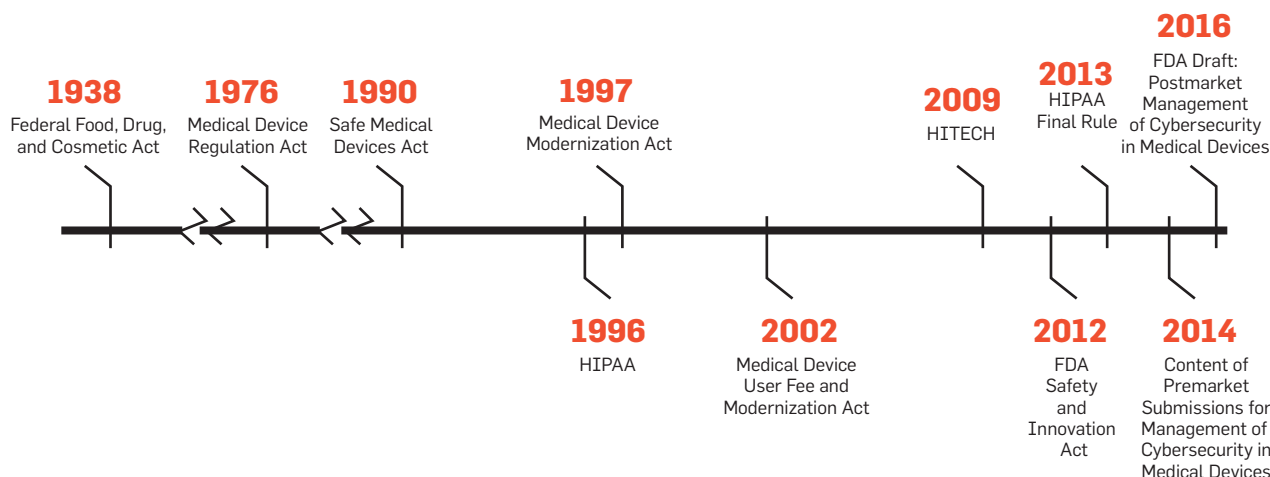
- The achievements of modern engineering and computer science are producing medical technologies that not only extend the lives of many patients, but also enhance the quality of life for many more managing chronic illness.
- Though medical devices are unique, the cybersecurity threats to medical device security are not unlike those that threaten other software-controlled, network-enabled devices.
- All security-focused decisions involve trade-offs. To fully understand the security trade-offs involved in designing, deploying, and maintaining medical devices, we believe it is critical to pause and take stock of what is at stake.



UPRIGHT



Timeline of Important Legislation



On Oct. 2, 2014, the FDA released its guidance on the management of cybersecurity in medical devices. This guidance represents the most recent in a long line of federal/legislative initiatives aimed at regulating and/or enhancing security and privacy in the highly sensitive health sector. We outline some of the important federal initiatives below, beginning with the passage of the Federal Food, Drug, and Cosmetic Act of 1938 and ending with the FDA Guidance on medical device cybersecurity in 2014.

1938 Federal Food, Drug, and Cosmetic Act. In the wake of a medicinal disaster known as Elixir Sulfanilamide in 1937, Congress passed the Federal Food, Drug, and Cosmetic Act of 1938. Today, this act along with its many amendments, has become one of the most influential in the history of U.S. medicine. In addition to extending the purview of the FDA over medical devices and cosmetics, the FD&C Act of 1938 also first mandated the FDA pre-market approval of pharmaceuticals. The overwhelming need for such regulation is expressed by a doctor's regrets over the Elixir Sulfanilamide incident:

"... six human beings, all of them my patients, one of them my best friend, are dead because they took medicine that I prescribed for them innocently, and to realize that that medicine which I had used for years in such cases suddenly had become a deadly poison in its newest and most modern form, as recommended by a great and reputable pharmaceutical firm in Tennessee: well, that realization has given me such days and nights of mental and spiritual agony as I did not believe a human being could undergo and survive. I have known hours when death for me would be a welcome relief from this agony." (Letter by Dr. A.S. Calhoun, Oct. 22, 1937)^{a,b}

22, 1937)^{a,b}

1976 Medical Device Regulation Act "passed to ensure safety and effectiveness of medical devices, including diagnostic products. The amendments require manufacturers to register with FDA and follow quality control procedures. Some products must have pre-market approval by FDA; others must meet performance standards before marketing."^c

1990 Safe Medical Devices Act "requires nursing homes, hospitals, and other facilities that use medical devices to report to FDA incidents that suggest that a medical device probably caused or contributed to the death, serious illness, or serious injury of a patient. Manufacturers are required to conduct post-market surveillance on permanently implanted devices whose failure might cause serious harm or death, and to establish methods for tracing and locating patients depending on such devices. The act authorizes FDA to order device product recalls and other actions."^d

1996 HIPAA. Regulated by the U.S. Department of Health and Human Services, the Health Information Portability and Accountability Act of 1996 resulted in the establishment of two important patient safeguards, the HIPAA Privacy Rule, and the HIPAA Security Rule.

a <http://www.fda.gov/AboutFDA/WhatWeDo/History/Origin/ucm054826.htm>

b Ballentine, Carol, "Taste of Raspberries, Taste of Death: The 1937 Elixir Sulfanilamide Incident," FDA Consumer magazine, 1981; Available at www.fda.gov/downloads/AboutFDA/WhatWeDo/History/Origin/ucm125604.doc.

c <http://www.fda.gov/AboutFDA/WhatWeDo/History/Origin/ucm054826.htm>

d <http://www.fda.gov/AboutFDA/WhatWeDo/History/Milestones/ucm128305.htm>

The Privacy Rule established "national standards for the protection of certain health information," and the Security Rule established "a national set of security standards for protecting certain health information that is held or transferred in electronic form."^e

⁴1997 FDA Modernization Act.

"Provisions include measures to accelerate review of devices, regulate advertising of unapproved uses of approved drugs and devices, and regulate health claims for foods."^e

2002 Medical Device User Fee and Modernization Act—"fees are assessed sponsors of medical device applications for evaluation, provisions are established for device establishment inspections by accredited third parties, and new requirements emerge for reprocessed single-use devices."^e

2009 HITECH Act. Enacted under the American Recovery and Reinvestment Act of 2009, the Health Information Technology for Economic and Clinical Health (HITECH) Act established for the provision of "business associate liability; new limitations on the sale of protected health information, marketing, and fundraising communications; and stronger individual rights to access electronic medical records and restrict the disclosure of certain information." Additionally, new rules were established for breach notifications with penalties applied for failure to notify individuals affected by breaches meeting certain criteria discovered after February 10, 2010.^e

2012 FDA Safety and Innovation Act (FDASIA) "expands FDA authorities to collect user fees from industry to fund reviews of innovator drugs, medical devices, generic

e <http://www.hhs.gov/ocr/privacy/hipaa/understanding/srsummary.html>

drugs and biosimilar biological products; promotes innovation to speed patient access to safe and effective products; increases stakeholder involvement in FDA processes, and enhances the safety of the drug supply chain.”^c

2013 HIPAA Final Rule. “[A] final rule that implements a number of provisions of the Health Information Technology for Economic and Clinical Health (HITECH) Act, enacted as part of the American Recovery and Reinvestment Act of 2009, to strengthen the privacy and security protections for health information established under the Health Insurance Portability and Accountability Act of 1996 (HIPAA).”^f

2014 FDA Guidance on Content of Premarket Submissions for Management of Cybersecurity in Medical Devices. Guidance issued by the FDA on the security of medical devices. It recommends that manufacturers “consider cybersecurity risks as part of the design and development of a medical device, and submit documentation to the FDA about the risks identified and controls in place to mitigate those risks. The guidance also recommends that manufacturers submit their plans for providing patches and updates to operating systems and medical software.”^g

2016 FDA Draft Guidance on Post-Market Management of Cybersecurity in Medical Devices. “The draft guidance details the agency’s recommendations for monitoring, identifying and addressing cybersecurity vulnerabilities in medical devices once they have entered the market.”^h

^f <http://www.hhs.gov/ocr/privacy/hipaa/administrative/omnibus/>

^g <http://www.fda.gov/NewsEvents/Newsroom/PressAnnouncements/ucm416809.htm>

^h <http://www.fda.gov/NewsEvents/Newsroom/PressAnnouncements/ucm481968.htm>

light the legislative timeline and the evolving threats to information security in healthcare.

Period 1. Complex Systems and Accidental Failures (1980s–Present)

“Welcome to the world of high-risk technologies,” begins the introduction to Charles Perrow’s treatise on Normal Accidents.²¹ From nuclear power plants to avionics to medical technologies, systems engineering feats in the second half of the 20th century forever altered human capabilities and the management of complex processes. However, these advances were accompanied by novel threats to the safety and security of these devices, and constituencies.

1985–1987: Therac-25. From June 1985 to January 1987, six patients received harmful levels of radiation due to defective Therac-25 accelerators. Still studied as a case of complex failure, the Therac-25 disaster was a deadly concoction of user error, faulty software engineering, and insufficient training/support. For example, in one instance, a software glitch caused the device to indicate a malfunction had occurred, causing a radiation therapist to erroneously readminister radiation several times. As it turned out, these glitches had become a part of the daily use of Therac-25, and the manufacturer’s support provided little help in the way of troubleshooting or interpreting error codes.¹⁶

2002: BIDMC network failure. On November 13, 2002, a researcher inadvertently flooded the network of the Beth Israel Deaconess Medical Center (BIDMC) with data, causing harmful delays in access to critical information and information systems. Unfortunately, the network diagnostics were only available through the network itself. When unable to sort out the issues, the hospital pulled the network offline for four days and reverted to paper-based processes. As noted, “[t]he principal point of failure was a software program for directing traffic on the network. The program was overwhelmed by a combination of data volume and network complexity that exceeded the software’s specifications.”¹⁴

Period 2. Implantable Medical Devices (2000–Present)

The first decade of the 21st century brought about significant changes in

the medical device landscape. By 2001, the number of implantable medical devices (IMDs) in use in the U.S. was greater than 25 million.²² The advent of IMDs raised the stakes considerably for the security and reliability of medical devices. Previously, the context of device failure was largely constrained to external devices housed in hospitals, clinics, and patient homes. With IMDs, the context of operation expanded symmetrically with the range of activity of the patient. Additionally, the integration of devices into the human body complicated the data communication process between device and physician.

2000s: Implantable Cardiac Defibrillator failure. From 1990 to 2000 the FDA issued recalls affecting 114,645 implantable cardiac defibrillators (ICD).¹⁸ In 2005, the death of a 21-year-old cardiac patient garnered greater attention than many of the previous ICD failures when an ICD short-circuited while initiating what might have been a life-saving electrical shock.¹³ In the aftermath of this high-profile tragedy, the health and safety risks associated with ICD malfunctions became a matter of public concern.^{19,23}

2005: HCMSS. In June 2005, a workshop on High Confidence Medical Device Software and Systems (HCMSS) was held in Philadelphia, PA. Sponsored by FDA, NIST, NSF, NSA, and NITRD, the workshop had the goal of developing a roadmap for overcoming crucial issues and challenges facing the design, manufacture, certification, and use of medical device software and systems.

Period 3. Unauthorized Parties and Medical Devices (2006–Present)

By 2006, medical device software had reached a tipping point in the U.S. as 50% of the medical devices on the market were either standalone software packages or other device-types with some software-driven functionality.^{6,22} The increasing complexity of these devices enabled by software led many researchers and some high-profile patients to begin questioning the vulnerability of medical devices (particularly IMDs) to unauthorized parties. It was during this time that the concept of ‘medical device hacking’ became a mainstream concern.

2006: Software updates for embed-

ded devices. In 2006, researchers demonstrated the challenges of securely updating the software of embedded devices.² Embedded devices lack interfaces that allow a client to acknowledge and install updates. Further, the nature of these devices necessitates that they are both nomadic and, in terms of network connectivity, sporadic. These attributes make embedded devices particularly susceptible to man-in-the-middle attacks.

2008: Implantable Cardiac Defibrillator. In 2008, researchers exposed vulnerabilities in an FDA-approved ICD that allowed modified-off-the-shelf devices to be configured to eavesdrop on information generated by the device and even control the defibrillator's dispensation of electric shock.¹²

2008: *Reigel vs. Medtronic*. In the midst of the revelations of novel security threats posed by implantable devices, the U.S. Supreme Court ruled in a high-profile case limiting liability for medical-device manufacturers for harms caused by devices approved by the FDA.⁵

2011: The year of the insulin pump. In 2011, several high-profile events involving the security of implantable insulin pumps caught the attention of the academics, practitioners, and the public at large. That same year a review of the state of trustworthy medical device software recommended the following to increase the trustworthiness of medical device software:⁹

- ▶ regulatory policies that specify outcome measures rather than technology,
- ▶ collection of statistics on the role of software in medical devices,
- ▶ establishment of open-research platforms for innovation,
- ▶ clearer roles and responsibility for the shared burden of software, clarification of the meaning of substantial equivalence for software, and
- ▶ an increase in Food and Drug Administration (FDA) access to outside experts in software.

2011: Peer-reviewed insulin pump vulnerability. In 2011, vulnerabilities of insulin pumps to unauthorized parties were disclosed.¹⁷ Using off-the-shelf hardware, successful passive attacks (for example, eavesdropping of the wireless communication) and active attacks (for example, impersonation and control of the medical devices to alter

the intended therapy) were achieved.¹⁷ These findings exposed a vulnerability in certain insulin pumps that could allow an unauthorized party to “emulate the full functions of a remote control: wake up the insulin pump, stop/resume the insulin injection, or immediately inject a bolus dose of insulin into the human body.”¹⁷

2011: Peer-reviewed defenses against unauthorized access to IMDs.

In response to emerging radio frequency (RF) vulnerabilities, a novel defense against unauthorized access proposed an RF shield to act as a proxy for communications with implantable medical devices (IMD).¹⁰ The shield actively prevents any device other than itself from communicating directly with the IMD by jamming all other communications.

Extending the shield concept, a similar defense emerged that passively monitors an individual's personal health system and interferes in the case of a detected anomaly, eliminating the need for protocol changes to interact with the “shield.”²⁴

2011: Jerome Radcliffe and Barnaby Jack. On August 4, 2011, Jerome Radcliffe, a diabetic patient, presented a talk at Black Hat 2011 in Las Vegas, NV, in which he announced he had partially reverse engineered the communication protocols for his own insulin pump. His presentation exposed a vulnerability in some insulin pumps allowing unauthorized access and control through the wireless channel.³ This presentation got the attention of many mainstream media outlets and brought the health and safety risks attributable to unauthorized access and control of medical devices previously identified by researchers and exploited in laboratories into the consciousness of the general public.

In October 2011, in Miami, FL, under the auspices of McAfee (now a division of Intel), famed late hacker Barnaby Jack made a presentation at the Hacker Halted conference exposing security vulnerabilities that allowed an insulin pump to be commandeered remotely via radio frequency.³

Period 4. Cybersecurity of Medical Devices (2012–Present)

Most recently, attention has turned to the cybersecurity of medical devices.

Harnessing the capabilities of ubiquitous networks, medical device manufacturers are increasingly enabling the connectivity of devices through the Internet or over networks, which also carry the Internet (for example, LANs). There are many advantages to connected devices, including real-time monitoring and software management such as remote installation of software updates. However, medical devices are not immune to the kinds of cybersecurity threats that have become prevalent in this network age. In fact, in terms of the potential consequences, the protection of medical devices is often more critical than that of other device types.

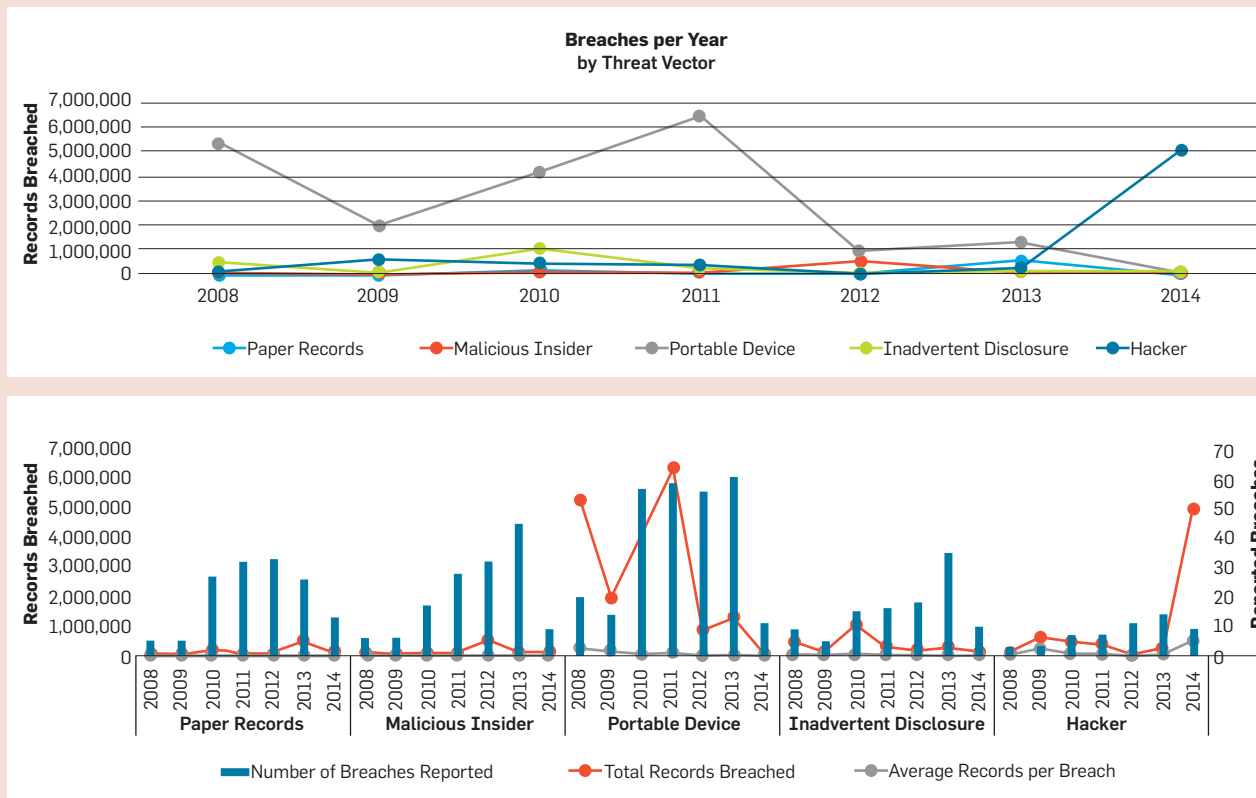
2012: ISPAB Board meeting. In February 2012, the Information Security and Privacy Advisory Board (ISPAB) held its annual board meeting in Washington, D.C. Great concern was expressed regarding emerging issues related to cybersecurity and the associated economic incentives of medical devices to increase medical device cybersecurity, and the coordination of agencies in the regulation of medical device cybersecurity.⁴

Specifically, software-controlled medical devices are increasingly available through and exposed to cybersecurity risks on the Internet. Further complicating this picture, the economics of medical device cybersecurity involves a complex system of payments between multiple stakeholders—including manufacturers, providers, and patients. At the same time, no one agency has primary responsibility from Congress to ensure the cybersecurity of medical devices deployed across this spectrum.⁴

2012: Barnaby Jack pacemaker hack. On October 17, 2012, at the Ruxcon Breakpoint Security Conference in Melbourne, Australia, Barnaby Jack exhibited a video presentation in which he demonstrated the ability to deliver an unwarranted shock through a pacemaker via wireless transmission. Jack found that certain devices could be accessed using a serial and model number. Exposing an important vulnerability, Jack disclosed that the devices would give up these credentials (that is, serial number and model number) when wirelessly contacted with a specific command, giving an unauthorized party the power to control the device.¹

2013–2014: FDA guidance on medi-

Evolving Threat Vectors of Infosec



On Aug. 18, 2014, Community Health Systems (CHS), one of the largest publicly traded hospital system in the U.S., reported that it had experienced the largest-ever breach of patient health information with the exposure of personal information of 4.5 million individuals. This hacking case, along with other high-profile instances, such as the highly publicized breach of a test server of the new Healthcare.gov site, highlight the evolving cyber-threat to information security in the health sector.

Health IT. The health industry has long been a laggard in terms of IT adoption. Today, spurred on by legislative initiatives such as HITECH, the rate of electronic health record (EHR) adoption is accelerating in the U.S. Increased opportunities for health information exchange, standardized data collections for use in medical research, and more effective treatment of patients are among the many potential benefits of the aggregation of patient health information into EHR systems. However, centralized EHR systems also create an economic incentive for malicious actors seeking access to the greatest number of records at the lowest cost. Previously, individual patient records were segmented

in large part by storing various versions of an individual's record, often in the form of paper records, in separate systems—creating less efficient targets (that is, information silos).

Breach trends. The magnitude and nature of the threat vectors to health information security have evolved over just the past few years. Assessing the breach information provided by the Privacy Rights Clearinghouse (privacyrights.org), two inflection points emerge:

- The increase in the number of breaches reported in 2010
- The emerging impact of cyber-threats in 2014

We believe the spike in reported breaches in 2010 is likely attributable to the passing of HITECH in 2009 and the accompanying stringent reporting standards and meaningful use requirements. Meanwhile, the cyber-threat to information security in 2014 was amplified by the CHS breach of 4.5 million records. Interestingly, it appears that the industry has improved its ability to limit the exposure of lost or stolen portable devices. In fact, despite a fairly consistent level of breaches reported, the total records breached from stolen or lost portable devices appears to be stabilizing at a lower level.

cal device cybersecurity. In June 2013, the FDA released draft guidance for the management of cybersecurity in medical devices, with the final guidance being released in October 2014.⁷ Drawing on much of the experiences and associated research presented here, the FDA guidance places emphasis on the need to consider device security during the design and development stages of med-

ical devices. Specifically the guidance recommends the following:

- Identification of assets, threats, and vulnerabilities;
- Assessment of the impact of threats and vulnerabilities on device functionality and end users/patients;
- Assessment of the likelihood of a threat and of a vulnerability being exploited;

- Determination of risk levels and suitable mitigation strategies; and
- Assessment of residual risk and risk acceptance criteria.

The guidance also identifies “core functions” of cybersecurity activities from the National Institute of Standards and Technology (NIST) cybersecurity framework.²⁰

2013–2016: State of medical device

security. Recently, security experts have begun advocating for a more holistic approach to securing increasingly complex and connected medical devices.²⁵ Noted trust challenges include: hardware failures/software errors, radio attacks, malware and vulnerability exploits, and side-channel attacks.²⁵

A 2014 survey of medical device security research found that the majority of the work in security and privacy has been centered on threats to the telemetry interface.²² That is, much prior research has examined threats to, and defenses of, medical device radio-based communication channels. The survey highlights five important areas of research in wireless telemetry: biometric authentication, distance-bounding authentication, out-of-band authentication, external devices, and anomaly detection.²²

There are inherent challenges in examining the software threats to medical device security. Not the least of these challenges is the reality that medical devices operate within a “closed source” paradigm, presenting challenges to performing static analyses or obtaining device firmware.²² Despite these challenges, the importance of ongoing security evaluation is clear, and the FDA’s 2016 draft guidance on post-market management of cybersecurity of medical devices seeks to provide recommendations for ensuring cybersecurity in devices that are already in circulation.⁸

The Future of Medical Device Security

The steps we take today will largely define the future of medical device security. Security is a game of trade-offs and the stakes are never higher than in healthcare. However, we must resist the temptation to sensationalize the issues related to cybersecurity in the health sector, and instead apply sober, rational, systematic approaches to understanding and mitigating security risks. Fortunately this approach is taking hold across the industry with the FDA recommending NIST’s cybersecurity framework prescribing that firms:

- **Identify.** Identify processes and assets needing protection;
- **Protect.** Define available safeguards;
- **Detect.** Devise incident detection techniques;

► **Respond.** Formulate a response plan; and

► **Recover.** Formalize a recovery plan.²⁰

In closing, the threats to the cybersecurity are emergent, and the inevitability of so-called ‘zero-day vulnerabilities’ must be addressed over the entire useful life of medical devices. This reality is a necessary outworking of innovation and should be embraced by healthcare providers, device manufacturers, software/app developers, security engineers, and even patients.

The medical field has long recognized the fiduciary responsibility of physicians with regard to patients’ well-being,¹¹ and it is safe to say that patients’ reluctance to accept medically indicated devices due to concerns about security poses a greater threat to their health than any threat stemming from medical device security. That said, in this world of high-risk medical technologies, it is incumbent on our field to continue to prioritize the security of medical devices as a part of our fiduciary responsibility to act in the interests of those who rely on these life-saving devices.

Acknowledgment

This work was supported by the National Science Foundation (NSF) project on Trustworthy Health and Wellness (THaW.org)–CNS-1329686 and CNS-1330142. The views expressed are those of the authors and should not be interpreted as representing the views, either expressed or implied, of NSF. We also thank Kevin Fu for his guidance. ■

References

1. Applegate, S.D. The dawn of kinetic cyber. In *Proceedings of the 5th International Conference on Cyber Conflict*. IEEE, 2013, 1–15.
2. Bellissimo, A. et al. Secure software updates: Disappointments and new challenges. In *Proceedings of the USENIX Summit on Hot Topics in Security*, 2006.
3. Burleson, W. et al. Design challenges for secure implantable medical devices. In *Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012, 12–17.
4. Chenok, D.J. ISPAD Letter to U.S. Office of Management and Budget (2012); http://csrc.nist.gov/groups/SMA/ispad/documents/correspondence/ispad-ltr-to-omb_med_device.pdf.
5. Curfman, G.D. et al. The medical device safety act of 2009. *New Eng. J. Med.* 360, 15 (2009), 1550–1551.
6. Faris, T.H. *Safe and Sound Software: Creating an Efficient and Effective Quality System for Software Medical Device Organizations*. ASQ Quality Press, 2006.
7. Food and Drug Administration. Content of Premarket Submissions for Management of Cybersecurity in Medical Devices; Guidance for Industry and Food and Drug Administration Staff (2014); <http://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/UCM356190.pdf>.
8. Food and Drug Administration. Postmarket

Management of Cybersecurity in Medical Devices; Draft Guidance for Industry and Food and Drug Administration Staff (2016); <http://www.fda.gov/downloads/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/ucm482022.pdf>.

9. Fu, K. Trustworthy medical device software. *Workshop Report on Public Health Effectiveness of the FDA 510 (k) Clearance Process: Measuring Postmarket Performance and Other Select Topics*. National Academies Press, Washington, D.C. (2011), 102.
10. Gollakota, S. et al. They can hear your heartbeats: Non-invasive security for implantable medical devices. *ACM SIGCOMM Computer Communication Review* 41, 4 (2011), 2–13.
11. Hafemeister, T.L. and Spinos, S. Lean on me: A physician’s fiduciary duty to disclose an emergent medical risk to the patient. *Washington University Law Review* 86, 5 (2009).
12. Halperin, D. et al. Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In *Proceedings of IEEE Symposium on Security and Privacy*. IEEE, 2008, 129–142.
13. Hauser, R.G. and Maron, B.J. Lessons from the failure and recall of an implantable cardioverter-defibrillator. *Circulation* 112, 13 (2005), 2040–2042.
14. Kilbridge, P. Computer crash-lessons from a system failure. *New Eng. J. Medicine* 348, 10 (2003), 881–882.
15. Lee, I. et al. High-confidence medical device software and systems. *Computer* 39, 4 (2006), 33–38.
16. Leveson, N.G. and Turner, C.S. An investigation of the Therac-25 accidents. *Computer* 26, 7 (1993), 18–41.
17. Li, C. et al. Hijacking an insulin pump: Security attacks and defenses for a diabetes therapy system. In *Proceedings of the 13th IEEE International Conference on e-Health Networking Applications and Services*. IEEE, 2011, 150–156.
18. Maisel, W.H. et al. Recalls and safety alerts involving pacemakers and implantable cardioverter-defibrillator generators. *JAMA* 286, 7 (2001), 793–799.
19. Meier, B. Maker of heart device kept flaw from doctors. *New York Times*, 2005.
20. National Institute of Standards and Technology (NIST). Framework for Improving Critical Infrastructure Cybersecurity (Ver. 1.0) Feb. 12, 2014; <http://www.nist.gov/cyberframework/upload/cybersecurity-framework-021214-final.pdf>.
21. Perrow, C. *Normal Accidents: Living with High Risk Technologies*. Princeton University Press, 2011.
22. Rushanan, M. et al. SoK: Security and privacy in implantable medical devices and body area networks. In *Proceedings of the 2014 IEEE Symposium on Security and Privacy*. IEEE CS, 524–539.
23. Vladeck, D.C. Medical Device Safety Act of 2009: Hearing before the Subcomm. on Health of the Comm. on Energy and Commerce (111th Cong., May 12, 2009); <http://scholarship.law.georgetown.edu/cong/45>.
24. Zhang, M. et al. MedMon: Securing medical devices through wireless monitoring and anomaly detection. *IEEE Trans. Biomedical Circuits and Systems* 7, 6 (2013), 871–881; DOI 10.1109/TBCAS.2013.2245664.
25. Zhang, M. et al. Towards trustworthy medical devices and body area networks. In *Proceedings of the 50th Annual Design Automation Conference*. ACM, 2013, 1–6.

A.J. Burns (aburns@uttyler.edu) is an assistant professor of computer science at the University of Texas, Tyler.

M. Eric Johnson (Eric.Johnson@owen.vanderbilt.edu) is the Ralph Owen Dean and Bruce D. Henderson Professor of Strategy at Vanderbilt University, Nashville, TN.

Peter Honeyman (honey@umich.edu) is a research professor of computer science and engineering at the University of Michigan, Ann Arbor.

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



Watch the authors discuss their work in this exclusive *Communications* video. <http://cacm.acm.org/videos/a-brief-chronology-of-medical-device-security>

research highlights

P. 74

Technical Perspective Naiad

By Johannes Gehrke

P. 75

Incremental, Iterative Data Processing with Timely Dataflow

By Derek G. Murray, Frank McSherry, Michael Isard,
Rebecca Isaacs, Paul Barham, and Martín Abadi

P. 84

Technical Perspective The Power of Parallelizing Computations

By James Larus

P. 85

Efficient Parallelization Using Rank Convergence in Dynamic Programming Algorithms

By Saeed Maleki, Madanlal Musuvathi, and Todd Mytkowicz

Technical Perspective

Naiad

By Johannes Gehrke

THE NAIADS in Greek mythology are the nymphs of fresh water. They are unpredictable and a bit scary, like big data, whose size has been exploding and continues to double every two years. Novel systems that process this data tsunami have been the focus of much research and development over the last decade. Many such big data processing systems are programmed through a workflow, where smaller programs with local state (nodes) are composed into bigger workflows through well-defined interfaces (edges). The resulting dataflows are then scaled to huge inputs through data parallelism (the execution of one node in the dataflow is scaled out across many servers), task parallelism (independent nodes in the dataflow are executed at the same time), and pipelining (a node later in the dataflow can already start its work based on partial output from its predecessors).

The most well-known class of such dataflow systems is based on the map-reduce pattern, enabling large-scale batch processing. These systems can process terabytes of data for preprocessing and cleaning, data transformation, model training and evaluation, and report generation, achieving high throughput while making the computation fault tolerant across hundreds of machines.

The second class of big data processing systems is stream-processing systems, which define dataflows that are optimized to react quickly to incoming data. They maintain state in their nodes for running aggregates or recent windows of data to watch out for patterns in the data stream; the occurrence of such patterns then triggers output for the next node. Stream processing systems are designed for low latency responses to new data while scaling with the arrival rate of records in the input streams. Example applications include monitoring data streams from the Internet of Things, high-performance trading, and maintenance of statistics for service analytics.

A third class of systems are graph processing systems optimized for dataflows with loops where it is important to efficiently handle state that is iteratively processed until a fixpoint for a computation is reached. Developers program graph algorithms by “thinking like a node” and writing the logic for a single node, and the platform then scales this to billions of nodes. Example applications include pagerank computation over large graphs and iterative solvers of linear systems such as Jacobi or Gaussian Belief Propagation.

The following paper describes the Naiad Dataflow System, which combines all these three classes in a single system, supporting high-throughput batch processing queries, low-latency data stream queries, and iterative programs in a single framework. The beauty of this work is that it does not create a Swiss Army Knife with different components for each capability, but that it unites them throughout the concept of *timely dataflow*, a simple, but expressive computational model that allows users to easily express all three of these concepts in the same platform. In timely dataflows, a record is associated with a structured timestamp that indicates where in the dataflow the record belongs. Thus, even though a huge number of records may flow concurrently through the system at any point in time with data and task parallelism and pipelining between nodes, it is easy to understand for a node which data it should process and when it should generate output. The result is a system that can have millisecond response times for low-latency queries, but also scales linearly for high-throughput applications.

In the stories about the Naiads, they are both the seduced and the seducers. May the following paper equally enchant you about the beauty of processing big data! 

Johannes Gehrke (johannes@acm.org) is a Distinguished Engineer in Office 365 at Microsoft, Bellevue, WA.

Copyright held by author.

COMMUNICATIONS APPS

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



Available for iPad, iPhone, and Android



Available for iOS, Android, and Windows

<http://cacm.acm.org/about-communications/mobile-apps>



Association for Computing Machinery

Incremental, Iterative Data Processing with Timely Dataflow

By Derek G. Murray, Frank McSherry, Michael Isard, Rebecca Isaacs, Paul Barham, and Martín Abadi

Abstract

We describe the timely dataflow model for distributed computation and its implementation in the Naiad system. The model supports stateful iterative and incremental computations. It enables both low-latency stream processing and high-throughput batch processing, using a new approach to coordination that combines asynchronous and fine-grained synchronous execution. We describe two of the programming frameworks built on Naiad: GraphLINQ for parallel graph processing, and differential dataflow for nested iterative and incremental computations. We show that a general-purpose system can achieve performance that matches, and sometimes exceeds, that of specialized systems.

1. INTRODUCTION

This paper describes the *timely dataflow* model for iterative and incremental distributed computation, and the Naiad system that we built to demonstrate it. We set out to design a system that could simultaneously satisfy a diverse set of requirements: we wanted efficient high-throughput processing for bulk data-parallel workloads; stateful computations supporting queries and updates with low latency (on the order of milliseconds); and a simple yet expressive programming model with general features like iteration. Systems already exist for batch bulk-data processing,^{6, 13, 27} stream processing,³ graph algorithms,¹¹ machine learning,¹⁵ and interactive ad hoc queries¹⁸; but they are all deeply specialized for their respective domains. Our goal was to find a common low-level abstraction and system design that could be re-used for all of these computational workloads. We were motivated both by the research question of whether such a low-level model could be found, and also by the pragmatic desire to reduce the engineering cost of domain-specific distributed systems by allowing them to share a single highly optimized core codebase.

To understand the difficulty of supporting low-latency, high-throughput, and iterative computations in the same system, we must first think about scheduling and coordination. An easy way to achieve low latency in a distributed system is to use fully decentralized scheduling with no global coordination: workers eagerly process messages sent by other workers and respond based on purely local information. One can write highly complex computations this way—for example using a trigger mechanism²¹—but it is challenging to achieve consistency across the system. Instead we sought a high-level programming model with the abstraction of computing over collections of data using constructs with well-understood semantics, including loops; however, it is hard to translate such a high-level program description into an uncoordinated mass of triggers. At the other extreme,

the easiest way to implement a high-throughput batch system with strong consistency is to use heavyweight central coordination, which has acceptable cost when processing large amounts of data, because each step of the distributed computation may take seconds or even minutes. In such systems it may make sense to insert synchronization barriers between computational steps,⁶ and manually unroll loops and other control flow into explicitly acyclic computation graphs.^{20, 27} The overhead of these mechanisms precludes low-latency responses in cases where only a small amount of data needs to be processed.

Timely dataflow is a computational model that attaches virtual timestamps to events in structured cyclic dataflow graphs. Its key contribution is a new coordination mechanism that allows low-latency asynchronous message processing while efficiently tracking global progress and synchronizing only where necessary to enforce consistency. Our implementation of Naiad demonstrates that a timely dataflow system can achieve performance that matches—and in many cases exceeds—that of specialized systems.

A major theme of recent high-throughput data processing systems^{6, 13, 27} has been their support for transparent fault tolerance when run on large clusters of unreliable computers. Naiad falls back on an older idea and simply checkpoints its state periodically, restoring the entire system state to the most recent checkpoint on failure. While this is not the most sophisticated design, we chose it in part for its low overhead. Faster common-case processing allows more computation to take place in the intervals between checkpointing, and thus often decreases the total time to job completion. Streaming systems are, however, often designed to be highly available³; users of such systems would rightly argue that periodic checkpoints are not sufficient, and that (setting aside the fact that streaming systems generally do not support iteration) a system like MillWheel³ could achieve much higher throughput if it simply dispensed with the complexity and overhead of fault tolerance. In keeping with the philosophy of timely dataflow we believe there is a way to accommodate both lazy batch-oriented and eager high-availability fault tolerance within a single design, and interpolate between them as appropriate within a single system. We have developed a theoretical design for timely dataflow fault tolerance² and are in the process of implementing it.

In the remainder of this paper we first introduce timely dataflow and describe how its distributed implementation

The original version of this paper was entitled “Naiad: A Timely Dataflow System” and was published in the *Proceedings of the 24th ACM Symposium on Operating Systems Principles* (Farmington, PA, Nov. 3-6, 2013), 439–455.

achieves our desiderata (Section 2). We then discuss some applications that we have built on Naiad, including graph computation (Section 3) and differential dataflow (Section 4). Finally we discuss lessons learned and open questions (Section 5). Some of the material in this article was previously published at SOSOP 2013 in a paper that describes Naiad in more detail.¹⁹

2. SYSTEM DESIGN AND IMPLEMENTATION

Figure 1 illustrates one type of application that motivated timely dataflow, since it mixes high-throughput iterative processing on large volumes of data with fine-grained, low-latency reads and updates of distributed state. Updates continually arrive at the left, reflecting activity in a social network. The dashed rectangle surrounds an iterative clustering algorithm that incrementally maintains a view of conversation topics, aggregated by the dynamic community structure that the recent activity implies. At the top, incoming queries request topic recommendations that are tailored to particular users and their community interests: these queries are joined with the freshest available clustering to provide high quality and up-to-date results. Before Naiad, no existing system could implement all of these features with acceptable performance. A standard solution might be to write the clustering algorithm in the language of a batch system like MapReduce⁶ or Spark²⁷ and re-run it from scratch every few hours, storing the output in a distributed datastore like Bigtable.⁵ A separate program might target a low-latency streaming system like MillWheel³ and perform a simpler non-iterative categorization of recent updates, saving fresh but approximate recommendations to another table of the distributed store. A third program would accept user queries, perform lookups against the batch and fresh data tables, combine them and return results. While this kind of hybrid approach has been widely deployed, a single program on a single system would be simpler to write and maintain, and it would be much easier to reason about the consistency of its outputs.

Combining these disparate requirements in a high-performance system is challenging, and a crucial first step was to design suitable abstractions to structure the necessary computation. This section starts by explaining the

computational model we arrived at, the abstractions we chose, and the reasoning behind them.

2.1. Dataflow

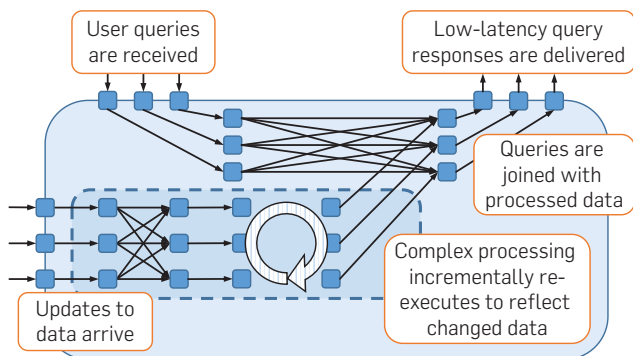
Our first choice was to represent every program as a dataflow graph. Dataflow is a common approach for distributed data processing^{6, 13, 27} because it explicitly encapsulates the boundaries between computations: the nodes of a dataflow graph represent subcomputations, and the directed edges represent the paths along which data is communicated between them. As a result, a system that represents its programs using dataflow can automatically determine subcomputations that can be executed in parallel. It then has a large degree of flexibility in scheduling them, and it can—at least in principle—place, move, and restart nodes independently without changing the semantics of the overall computation.

We based our design on *stateful dataflow*, in which every node can maintain mutable state, and edges carry a potentially unbounded stream of messages. Although statefulness complicates fault tolerance, we believe that it is essential for low-latency computation. Incremental or iterative computations may hold very large indexed data structures in memory and it is essential that an application be able to rapidly query and update these data structures in response to dataflow messages, without the overhead of saving and restoring state between invocations. We chose to require state to be private to a node to simplify distributed placement and parallel execution. One consequence of adopting stateful dataflow is that loops can be implemented efficiently using cycles in the dataflow graph (with messages returning around a loop to the node that stores the state). In contrast, stateless systems^{20, 27} implement iteration using acyclic dataflow graphs by dynamically unrolling loops and other control flow as they execute.

Having settled on stateful dataflow we attempted to minimize the number of execution mechanisms, in order to make timely dataflow systems easier to reason about and optimize. For example, we adopted the convention that all computation in nodes occurs in single-threaded event handlers, which the runtime invokes explicitly. With this convention all scheduling decisions are centralized in a common runtime, making CPU usage more predictable and allowing the system builder to aggressively optimize performance and control latency. It also simplifies the implementation of individual nodes: because the system guarantees that all event handlers will run in a single thread, the application programmer can ignore the complexities of concurrent programming. By encouraging single-threaded node implementations we push programmers to obtain parallelism by adding nodes to the dataflow graph, and force the system builder to ensure low overhead when scheduling a node's computation. The resulting system should be able to interleave many short-lived invocations of different nodes, and be well-suited to performing fine-grained updates with low latency.

Data-parallelism is a standard approach for constructing parallel dataflow graphs from operators whose inputs and outputs are collections of records. A data-parallel operator

Figure 1. An application that supports real-time queries on continually updated data. The dashed rectangle represents iterative processing that incrementally updates as new data arrive.



includes a *key function* that maps each input record to a key, such that records with different keys can be processed independently. As DeWitt and Gray showed more than 20 years ago, such an operator can be implemented in a dataflow graph by splitting it into multiple nodes, each of which takes responsibility for a disjoint subset of the key space.⁷ Data-parallelism is attractive because the results are identical regardless of how one partitions the key space, so the programmer need only specify an appropriate key function, and the system can automatically choose the degree of parallelism. Our framework libraries provide standard data-parallel operators that can be customized for specific applications.

2.2. Timely dataflow

Applications should produce consistent results, and consistency requires coordination, both across dataflow nodes and around loops. We called new model “timely” dataflow because it depends on logical timestamps to provide this coordination. We started with the goal of supporting general-purpose incremental and iterative computations with good performance, and then tried to construct the narrowest possible programming interface between system and application writer that satisfied our requirements. Our desire for a narrow interface, like our desire for few mechanisms, stems from the belief that it makes systems simpler to understand and engineer.

Asynchronous messages. All dataflow models require some means for one node to send a message along an outgoing edge to another node. In a timely dataflow system, each node implements an `OnRecv` event handler that the system can call when a message arrives on an incoming edge, and the system provides a `Send` method that a node can invoke from any of its event handlers to send a message on an outgoing edge. Messages are delivered asynchronously, which gives the system has great latitude in how the messages are delivered. For example, it can buffer messages between a pair of nodes to maximize throughput. At the other extreme the system may deliver messages via “cut-through,” whereby the `OnRecv` handler for the destination node runs on the same callstack as the source’s `Send` call. Cut-through eliminates buffering altogether, which improves cache performance and enables optimizations such as eager data reduction²⁵ that can drastically reduce memory consumption. Unlike the Synchronous Data Flow model,¹⁴ a timely dataflow node may call `Send` a variable number of times in response to an incoming message; as a result, timely dataflow can represent more programs, but it requires dynamic scheduling of the individual nodes.

Each message in a timely dataflow graph is labeled with a logical timestamp. A timestamp can be as simple as an integer attached to an input message to indicate the batch in which it arrived. Timestamps are propagated through computations and, for example, enable an application programmer to associate input and output data. Timely dataflow also supports more complex multi-dimensional timestamps, which can be used to enforce consistency when dataflow graphs contain cycles, as outlined below.

Consistency. Many computations include subroutines that must accumulate all of their input before generating an

output: consider for example reduction functions like `Count` or `Average`. At the same time, distributed applications commonly split input into small asynchronous messages to reduce latency and buffering as described above. For timely dataflow to support incremental computations on unbounded streams of input as well as iteration, it a mechanism to signal when a node (or data-parallel set of nodes) has seen a consistent subset of the input for which to produce a result.

A *notification* is an event that fires when all messages at or before a particular logical timestamp have been delivered to a particular node. Since a logical timestamp t identifies a batch of records, a notification event for a node at t indicates that all records in that batch have been delivered to the node, and a result can be produced for that logical timestamp. We exposed notifications in the programming model by adding a system method, `NotifyAt( $t$ )`, that a node can call from an event handler to request a notification. When the system can guarantee that no more messages with that timestamp will ever be delivered to the node, it will call the node’s `OnNotify( $t$ )` handler. This guarantee is a global property of the state of the system and relies on a distributed protocol we describe below. Nodes typically use an `OnNotify` handler to send a message containing the result of a computation on a batch of inputs, and to release any temporary state associated with that batch.

Iteration with cyclic graphs. Support for iteration complicates the delivery of notifications, because in a cyclic dataflow graph the input to a node can depend on its output.^a As a result, we had to invent suitable restrictions on the structure of timely dataflow graphs, and on the timestamps that can be affixed to messages, to make the notification guarantee hold. The general model is described in detail elsewhere^{1,19} but the restrictions that we adopted in the Naiad system are easy to explain informally. A Naiad dataflow graph is acyclic apart from structurally nested cycles that correspond to loops in the program. The logical timestamp associated with each event at a node is a tuple of one or more integers, in which the first integer indicates the batch of input that the event is associated with, and each subsequent integer gives the iteration count of any (nested) loops that contain the node. Every path around a cycle includes a special node that increments the innermost coordinate of the timestamp. Finally, the system enforces the rule that no event handler may send a message with a time earlier than the timestamp for the event it is handling. These conditions ensure that there is a *partial order* on all of the pending events (undelivered messages and notifications) in the system, which enables efficient progress tracking.

2.3. Tracking progress

The ability to deliver notifications promptly and safely is critical to a timely dataflow system’s ability to support low-latency incremental and iterative computation with consistent results. For example, the system can use a global

^a MillWheel has a notification—or “Timer”—interface that is similar to the timely dataflow design,³ but since it does not support iteration, the timestamps are simply integers and the graph is acyclic, greatly simplifying progress tracking.

progress tracker to establish the guarantee that no more messages with a particular timestamp can be sent to a node. By maintaining an aggregated view of the pending events in the system, the progress tracker can use the partial order on these events to determine (for each node) the earliest logical time of any subsequent event; this earliest time is monotonic (i.e., it never goes backwards). Moreover, there is an efficient way—sketched below—to compute this earliest time so that notifications are delivered promptly when they come due.

The progress tracker is an “out-of-band” mechanism for delivering notifications. Previous systems have implemented the equivalent of notifications using “in-band” control messages along dataflow edges: for example by requiring nodes to forward a special “punctuation” message on their outgoing edges to indicate that a batch is complete.²⁴ While in-band punctuations might appear to fit better with our philosophy of keeping things simple, the performance benefits of the out-of-band progress tracker design outweighed the cost of the extra complexity. Punctuations are unattractive for data-parallel dataflow graphs because the number of messages that must be sent to indicate the end of a batch is proportional to the number of edges in the graph rather than the number of nodes (as in the out-of-band design). The simplicity of punctuations breaks down when the dataflow can be cyclic, because (i) a node cannot produce a punctuation until it receives punctuations on all of its inputs, and (ii) in a cyclic graph at least one node must have an input that depends on its output. Although punctuations support a limited class of iterative computations,⁴ they do not generalize to nested iteration or non-monotonic operators, and so do not meet our requirements.

Having established the need for out-of-band coordination, we could still have adopted a simpler centralized scheduling discipline, for example triggering nodes to process events in each iteration after the previous was complete. A subtle but powerful property of incrementally updated iterative computation convinced us to pursue superior performance. Consider for example the problem of computing the connected components of a large graph: it might require 200 iterations and be partitioned over 100 worker computers. Now imagine re-running the computation after deleting a single edge from the graph. It would not be surprising if the work done in the second run were identical to that in the first except for, say, eight distinct loop iterations; and if those iterations differed only at two or three workers each. When incrementally updating the computation, a sophisticated implementation can actually be made to perform work only at those 20 or so times and workers, and this is only possible because the out-of-band notification mechanism can “skip over” workers and iterations where there is nothing to do; a design that required the system to step each node around the loop at every iteration would be much less efficient. This example also illustrates a case in which event handlers send messages and request notifications for a variety of times in the future of the events being processed; again, we could have chosen a simpler design that restricted this generality, but we would have lost substantial performance for useful applications. Space does not permit a full treatment of

the node logic needed for such applications, but Section 4 sketches an explanation and provides further references.

2.4. Implementation

Naiad is our high-performance distributed implementation of timely dataflow. It is written in C#, and runs on Windows, Linux, and Mac OS X.^b A Naiad application developer can use all of the features of C#, including classes, structs, and lambda functions, to build a timely dataflow graph from a system-provided library of generic `Stream` objects. Naiad uses deferred execution: at runtime executing a method like `Max` on a `Stream` actually adds a node to an internal dataflow graph representation. The dataflow computation only starts once the graph has been completely built and data are presented to the input nodes. The same program can run on a single computer with one or more worker threads, or—with a simple change in configuration—as a process that communicates with other instances of the same program in a distributed computation. Workers exchange messages locally using shared memory, and remotely using persistent TCP connections between processes. Each dataflow edge transmits a sequence of objects of a particular C# type, and generics are used extensively so that operators and the edges connecting them can be strongly typed.

Performance considerations. To achieve performance that is competitive with more specialized systems, we heavily optimized Naiad’s few primitive mechanisms. In particular, we found it necessary to reduce overheads from serialization for .NET types by adding run-time code generation, and from garbage collection by using value types extensively in the runtime and standard operators. In order to get low-latency responses to small incremental updates and fast loop iterations, we needed to ensure that progress tracking is efficient: notifications are delivered to a node as soon as possible once it cannot be sent any more messages with a given timestamp.

Naiad’s progress tracking protocol is essentially equivalent to distributed reference counting for termination detection or garbage collection.²³ Each event is associated with a graph location (edge or node): a message with the edge it is sent on, and a notification with the node that requests and receives it. Each worker maintains a count for its local view of the number of outstanding events for each pair of location and timestamp. Whenever an event is delivered the progress tracker decrements the corresponding location’s count at the event’s timestamp and increments any counts for messages sent or notifications requested by the event handler, then broadcasts this information to all other workers.

As stated, this protocol would be wildly inefficient, but we made several optimizations that allow workers to accumulate updates and delay sending them without stalling the global computation. As a simple example, if a worker has a pending notification at a node n and timestamp t , it can safely accumulate updates to later timestamps until that notification is delivered. Accumulated updates frequently cancel each other out, so the global broadcast traffic is much less than a naive analysis would suggest.

^b The full source code is available from <https://github.com/TimelyDataflow/Naiad>.

The delivery of notifications defines the critical path for a Naiad computation, and the protocol as implemented can dispatch notifications across a cluster in a single network round-trip. Figure 2 shows that, using the protocol, a simple microbenchmark of notifications in a tight loop performs a global barrier across 64 servers (connected by Gigabit Ethernet) with a median latency of just 750 μ s.

Layering programming abstractions. We wanted to ensure that Naiad would be easy to use for beginners, while still flexible enough to allow experienced programmers to customize performance-critical node implementations. We therefore adopted a layered model for writing Naiad programs. The lowest layer exposes the raw timely dataflow interfaces for completely custom nodes. Higher layers are structured as framework libraries that hide node implementations behind sets of data-parallel operators with related functionality whose inputs and outputs are distributed collections of C# objects.

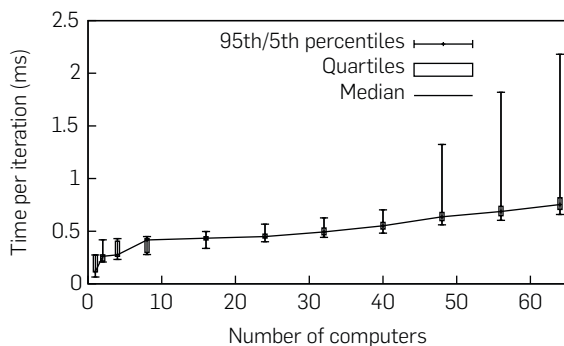
We modeled many of our libraries on the distributed query libraries in DryadLINQ,²⁶ with the added support for graph processing and incremental computation that we discuss in the following sections. Within libraries we can often re-use common implementations; for example most of the LINQ operators in Naiad build on unary and binary forms of a generic buffering operator with an `OnRecv` callback that adds records to a list indexed by timestamp, and an `OnNotify(t)` method that applies the appropriate transformation to the list or lists for time t . In many cases we were able to specialize the implementation of operators that require less coordination: for example `Concat` immediately forwards records from either of its inputs, `Select` transforms and outputs data without buffering, and `Distinct` outputs a record as soon as it is seen for the first time.

The ease of implementing new frameworks as libraries on Naiad enabled us to experiment with various distributed processing patterns. In the following sections, we elaborate on the frameworks that we built for graph processing (Section 3) and differential dataflow (Section 4).

3. GRAPH PROCESSING ON NAIAD

It is challenging to implement high-performance graph algorithms on many data processing systems. Distributed graph

Figure 2. The median latency of a global barrier implemented using notifications in a cycle is just 750 μ s on 64 machines. Error bars show the 95th percentile latencies in each configuration.

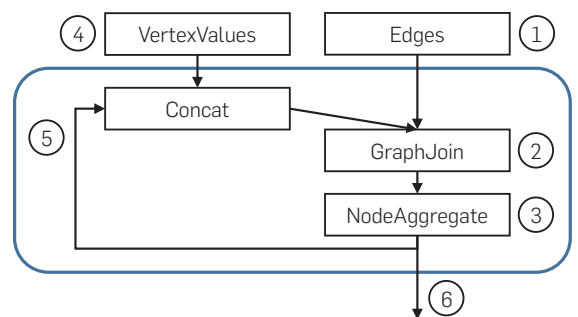


algorithms typically require efficient communication, coordination at fine granularity, and the ability to express iterative algorithms. These challenges have spurred research into specialized distributed graph-processing systems¹¹ and—more recently—attempts to adapt dataflow systems for graph processing.¹² We used a variety of graph algorithms to evaluate both the expressiveness of the timely dataflow programming model and the performance of our Naiad implementation. To avoid confusion in this section we use the term “operator” for dataflow nodes, and “graph,” “node,” and “edge” refer to elements of the graph that is being analyzed by a program running on Naiad unless otherwise qualified.

To understand how we implement graph algorithms on Naiad, it is instructive to consider the Gather-Apply-Scatter (GAS) abstraction of Gonzalez et al.¹¹ In the GAS abstraction, a graph algorithm is expressed as the computation at a node in the graph that (i) gathers values from its neighbors, (ii) applies an update to the node’s state, and (iii) scatters the new value to its neighbors. Figure 3 shows how we express this abstraction as a timely dataflow graph. The first step is to load and partition the edges of the graph (1). This step might use a simple hash of the node ID, or a more advanced partitioning scheme that attempts to reduce the number of edges that cross partition boundaries. The core of the computation is a set of stateful *graph-join* operators (2), which store the graph in an efficient in-memory data structure that is optimized for random node lookup. The graph-join effectively computes the inner join of its two inputs—the static (*src, dst*) edge relation, and the iteratively updating (*src, val*) state relation—and has the effect of scattering the updated state values along the edges of the graph. A set of stateful *node-aggregate* operators (3) perform the gather and apply steps: they store the current state of each node in the graph, gather incoming updates from the neighbors (i.e., the output of the graph-join), apply the final value to each node’s state, and produce it as output. To perform an iterative computation, the node-aggregate operators take the initial value for each node in the first iteration (4), feed updated state values around the back-edge of the loop (5), and produce the final value for each node after the algorithm reaches a fixed point (6).

Depending on the nature of the algorithm, it may be possible to run completely asynchronously, or synchronize after each iteration. In our experience, the most efficient implementation of graph algorithms like PageRank or weakly

Figure 3. Illustration of a graph algorithm as a timely dataflow graph.



connected components uses `OnRecv` to aggregate incoming values to the node-aggregate operator asynchronously, and `OnNotify` to produce new aggregated states for the nodes synchronously in each iteration. Because it is possible to coordinate at timescales as short as a millisecond, more complex graph algorithms benefit from dividing iterations into synchronous sub-iterations, using the prioritization technique that we briefly describe in Section 4.

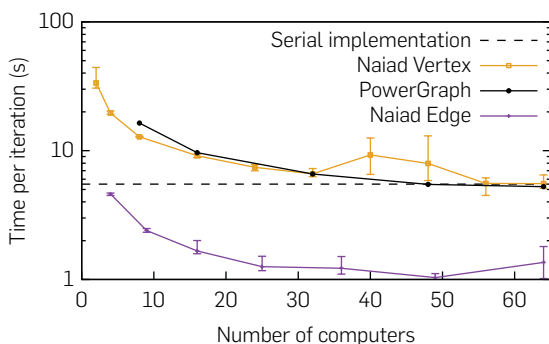
Motivated by the dataflow in Figure 3, we implemented the GraphLINQ framework on Naiad. GraphLINQ extends the LINQ programming model—with its higher-order declarative operators over collections, such as `Select`, `Where`, and `GroupBy`—with `GraphJoin`, `NodeAggregate`, and `Iterate` operators that implement the specialized dataflow nodes depicted in Figure 3. GraphLINQ allows the programmer to use standard LINQ operators to define the dataflow computation that loads, parses, and partitions the input data as a graph, and then specify a graph algorithm declaratively. A simple implementation of PageRank is just nine lines of GraphLINQ code.

When implementing graph algorithms on a dataflow system, a common concern is that the generality of the system will impose a performance penalty over a specialized system. To evaluate this overhead, we measured the performance of several implementations of PageRank on a publicly available crawl of the Twitter follower graph, with 42 million nodes and 1.5 billion edges.^c Figure 4 compares two Naiad implementations of PageRank to the published results for PowerGraph,¹¹ which were measured on comparable hardware.^d We present two different implementations of PageRank on Naiad. The first (“Naiad Vertex”) uses a simple hash function to partition the nodes of the Twitter graph between the workers, and performs all processing for each node on a single worker; this implementation performs similarly to the best PowerGraph implementation, taking approximately 5.55 s per iteration on 64 machines. The more advanced (“Naiad Edge”) implementation uses

^c <http://an.kaist.ac.kr/traces/WWW2010.html>.

^d The Naiad results were computed using two racks of 32 servers, each with two quad-core 2.1 GHz AMD Opteron processors, 16 GB of RAM, and an Nvidia NForce Gigabit Ethernet NIC. The PowerGraph results were computed using 64 Amazon EC2 `cc1.4xlarge` instances, each with two quad-core Intel Xeon X5570 processors, 23 GB of RAM, and 10Gbit/s networking.¹¹

Figure 4. Time per iteration for PageRank on the Twitter follower graph, as the number of machines is varied.



an edge-based partitioning in the spirit of PowerGraph’s edge partitioning with a vertex cut objective, but based on a space-filling curve¹⁶; it outperforms PowerGraph by a factor of 5, taking just 1.03 s per iteration on 49 machines. Figure 4 plots a single-threaded baseline for the PageRank operation, using a late-2014 MacBook Pro with 16 GB of RAM: using a similar data layout to the advanced Naiad implementation, this implementation takes 5.25 s per iteration.

4. DIFFERENTIAL DATAFLOW

Differential dataflow is a computational framework that we developed to efficiently execute and incrementally update iterative data-parallel computations. The framework comprises algorithms, data structures, and dataflow graph constructs layered atop a timely dataflow system.¹⁷

4.1. Incremental view maintenance

Differential dataflow is a generalization of *incremental view maintenance*, a useful technique from database systems. Incremental view maintenance can be implemented as a dataflow graph of data-parallel nodes. Each node continually receives records and maintains the correct output for their accumulation. Because the node implementations are data-parallel, they only need to revisit previously received input records with the same keys as newly arriving inputs. Looking at only these records, the node can determine how the output must be corrected (if at all) to reflect the new input. By producing and communicating only changed output records, the node informs downstream nodes of the relatively few keys they must reconsider. The system as a whole performs work only when and where actual changes occur.

Incremental view maintenance is the basis for many successful stream processing systems³ and graph processing systems.⁸ In a stream processing system, a small per-record update time means that the system can execute with very low latency compared to batch systems. In an incremental graph processing system, the time to perform a round of message exchanges depends only on the number of messages exchanged rather than the total number of nodes or edges. Despite its value for both stream and graph processing systems, incremental view maintenance is not suitable for combining the two.

4.2. From incremental to differential dataflow

Differential dataflow provides the ability to combine incremental and iterative updates by removing the implicit assumption that time is totally ordered; instead it indexes and accumulates records according to partially ordered timestamps. Consider a graph processing system that accepts incremental updates to its node and edge sets, and correctly updates the output of an iterative computation. This system must deal with multiple types of updates, due to both iterations progressing and inputs changing; differential dataflow distinguishes these types of updates using multi-dimensional logical timestamps. When a new record arrives, the implementation constructs the accumulation needed to determine the new output from all records with timestamps less than or equal to that of the new record. Concretely, consider the example

of timestamps (epoch, iteration) for multiple rounds of an iterative computation that receives multiple epochs of updated input. Using the partial order $(a, b) \leq (x, y)$ iff $a \leq x \wedge b \leq y$ we can get both the standard streaming and graph processing behavior at once: a timestamp (epoch, 0) collects all updates $(i, 0)$ with $i \leq \text{epoch}$, and a timestamp (0, round) collects all updates $(0, j)$ with $j \leq \text{round}$. Further, a timestamp (epoch, round) can take advantage of exactly those records that are useful for it: those at timestamp (i, j) where $i \leq \text{epoch}$ and $j \leq \text{round}$. Records at later epochs or rounds can be ignored.

Figure 5 shows, for different implementation strategies, the execution time for each iteration of a graph processing computation: namely, weakly connected components (via label propagation) on a graph derived from a 24-h window of Twitter mentions. Each vertex represents a user, and it repeatedly exchanges the smallest user ID it has seen so far (including its own) with its neighbors. As the computation proceeds, labels eventually stop changing and converge to the smallest user ID in each connected component. The implementation strategies are as follows:

- **Stateless** batch execution (not shown) repeatedly recomputes all labels in each iteration, and does a constant number of updates as the computation progresses. This is the baseline version that could be implemented on top of MapReduce.
- **Incremental** dataflow uses incremental view maintenance to improve on the stateless version. The amount of work decreases as the computation starts to converge and unchanged labels are neither re-communicated nor re-computed.
- **Prioritized** differential dataflow improves on this further by incrementally introducing the labels to propagate, starting with the smallest values (those most likely to be retained at each vertex) and adding larger values only once the small labels have fully propagated. The advantage of introducing small labels earlier is that many vertices (that eventually receive small labels) will no longer propagate the larger labels that they possess during the early iterations, which reduces

the amount of unproductive communication and computation.

The **1s change** series shows that the amount of work required to update the edge set by sliding the window forward one second—incrementally updating the connectivity structures as well—is vanishingly small by comparison.

Since differential dataflow uses the same representation for incremental and iterative changes to collections, the techniques are composable. Figure 7 shows an implementation of an algorithm for finding the strongly connected components (SCC) of a directed graph. The classic algorithm for SCC is based on depth-first search, which is not easily parallelizable. However, by nesting two connected components queries (Figure 6) inside an outer FixedPoint, we can write a data-parallel version using differential dataflow (Figure 7). Strictly speaking the connected components query computes directed reachability, and the SCC algorithm repeatedly removes edges whose endpoints reach different components and must therefore be in different SCCs. Iteratively trimming the graph in alternating directions—by

Figure 6. A connected components algorithm in differential dataflow that uses FixedPoint to perform iterative aggregation over node neighborhoods.

```
// produces a (src, label) pair for each node in the graph
Collection<Node> ConnectedComponents(Collection<Edge> edges)
{
    // start each node with its own label, then iterate
    return edges.Select(x => new Node(x.src, x.src))
        .FixedPoint(x => LocalMin(x, edges));
}

// improves an input labeling of nodes by considering the
// labels available on neighbors of each node as well
Collection<Node> LocalMin(Collection<Node> nodes,
    Collection<Edge> edges)
{
    return nodes.Join(edges, n => n.src, e => e.src,
        (n, e) => new Node(e.dst, n.label))
        .Concat(nodes)
        .Min(node => node.src, node => node.label);
}
```

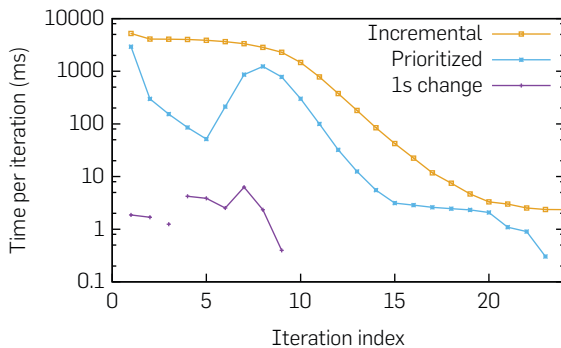
Figure 7. A function to compute strongly connected components in differential dataflow that uses connected components (Figure 6) as a nested iterative subroutine.

```
// returns edges between nodes within a SCC
Collection<Edge> SCC(Collection<Edge> edges)
{
    return edges.FixedPoint(y => TrimAndReverse(
        TrimAndReverse(y)));
}

// returns edges whose endpoints reach the same node, flipped
Collection<Edge> TrimAndReverse(Collection<Edge> edges)
{
    // establish labels based on reachability
    var labels = ConnectedComponents(edges);

    // struct LabeledEdge(a,b,c,d): edge (a,b); labels c, d;
    return edges.Join(labels, x => x.src, y => y.src,
        (x, y) => x.AddLabel1(y))
        .Join(labels, x => x.dst, y => y.src,
        (x, y) => x.AddLabel2(y))
        .Where(x => x.label1 == x.label2)
        .Select(x => new Edge(x.dst, x.src));
}
```

Figure 5. The execution time for each iteration of the connected components algorithm, for a graph built from a Twitter conversation dataset. The “1s change” curve shows a sliding window update that requires no work for many of the iterations.



reversing the edges in each iteration—eventually converges to the graph containing only those edges whose endpoints are in the same SCC.

4.3. Implementation

Our implementation of differential dataflow comprises several standard nodes, including `Select`, `Where`, `GroupBy`, and `Join`, as well as a higher-order `FixedPoint` node that iteratively applies an arbitrary differential dataflow expression until it converges to a fixed point. The records exchanged are of the form (data, time, difference), where data is an arbitrary user-defined type, time is a timestamp, and difference is a (possibly negative) integer.

The standard nodes have somewhat subtle implementations that nonetheless mostly follow from the mathematical definition of differential dataflow¹⁷ and the indexing needed to respond quickly to individual time-indexed updates. The `FixedPoint` node introduces a new coordinate to the timestamps of enclosed nodes, and extends “less or equal” and “least upper bound” for the timestamps according to the product order described above (one timestamp is less than or equal to another if all of its coordinates are). An important aspect of the implementation is that all differential dataflow nodes are generic with respect to the type of timestamp as long as it implements “less or equal” and “least upper bound” methods, and this means that they can be placed within arbitrarily nested fixed-point loops. When the fixed point of an expression is computed, the expression’s dataflow subgraph is constructed as normal, but with an additional connection from the output of the subgraph back to its input, via a node that advances the innermost coordinate by one (informally, this advances the iteration count).

5. LESSONS LEARNED AND OPEN QUESTIONS

Timely dataflow demonstrates that it is possible to combine asynchronous messaging with distributed coordination to generate consistent results from complex, cyclic dataflow programs. Naiad further demonstrates that we can build a system that combines the flexibility of a general-purpose dataflow system with the performance of a specialized system.

Our original Naiad implementation used C# as the implementation language. C#’s support for generic types and first-class functions makes it simple to build a library of reusable data-parallel operators like LINQ. The fact that a running C# program has access to its typed intermediate-language representation means that reflection can be used to generate efficient serialization code automatically. The advantage of automatic serialization when writing distributed applications should not be underestimated, since it allows programmers to use native language mechanisms like classes to represent intermediate values without paying the penalty of writing and maintaining serializers for every class.

Some of C#’s productivity benefits come at a cost to performance and we had to work to minimize that cost. The .NET runtime uses a mark-and-sweep garbage collector (GC) to reclaim memory, which simplifies user programs but presents challenges for building an

efficient system based on maintaining a large amount of state in memory. While we were able to use C# value types to reduce the number of pointers on the heap—and hence the amount of GC work required—it was not possible to eliminate GC-related pauses completely. Since building the original version of Naiad we have investigated alternative designs that would reduce the impact of garbage collection: the Broom project shows encouraging improvements in the throughput of Naiad programs using region-based memory allocation,⁹ and a reimplement of timely dataflow in the Rust language eliminates the garbage collector altogether.⁶

Many distributed dataflow systems exploit deterministic execution to provide automatic fault tolerance,^{13, 20, 27} but Naiad embraces non-determinism and asynchrony to produce results sooner. Furthermore, Naiad vertices can maintain arbitrary state, which makes it non-trivial to generate code that produces a checkpoint of a vertex. As explained in the introduction our current implementation of fault tolerance is based on restoring from a global checkpoint, which requires code in each stateful vertex to produce and consume a checkpoint of its state. Global checkpointing introduces a large amount of skew into the distribution of epoch and iteration execution times, and forces non-failing processes to roll back in the event of a failure. We have developed a model that permits different vertices to implement different checkpointing policies,² and are working on a Naiad implementation of the model, which exposes a range of performance tradeoffs that in many cases allow high-throughput, low-latency, and fault-tolerant execution.

Finally we note that, while Naiad supports the composition of many different models of computation in the same program, it lacks a high-level programming language—such as SQL—and an optimizer that chooses the most appropriate models for a particular task. Other authors have applied program analysis and query optimization techniques to Naiad. Sousa et al.²² achieved speedups over Naiad’s built-in operators by analyzing user-defined functions and generating new operators. Gog et al.¹⁰ achieved encouraging results with Musketeer, which transforms possibly iterative programs written in a high-level language into code that uses a variety of systems including Naiad. Still, we believe that there is scope for a more powerful compiler that can target Naiad’s different libraries, including differential dataflow, and generate optimized vertex code.

Acknowledgments

We worked on Naiad at the Microsoft Research Silicon Valley Lab between 2011 and the lab’s closure in September 2014. We thank Dave Andersen, Amer Diwan, and Matt Dwyer for suggestions on improving this paper. We are grateful to all of our former colleagues who commented on previous versions of the work, and especially to Roy Levin and Mike Schroeder, who created a unique environment in which this kind of research was encouraged and nurtured. 

⁶ <https://github.com/frankmcsherry/timely-dataflow>.

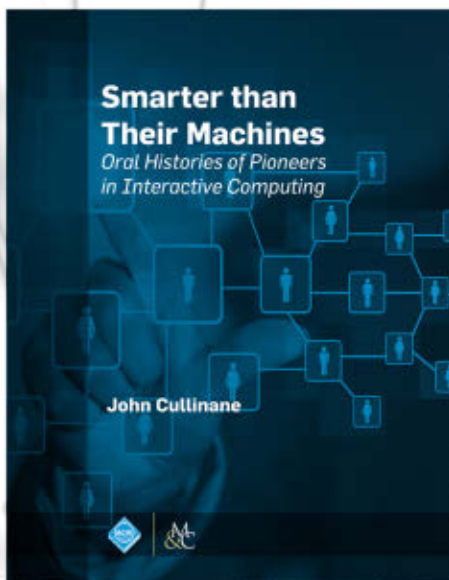
References

1. Abadi, M., Isard, M. Timely dataflow: A model. In *Proc. FORTE* (2015), 131–145.
2. Abadi, M., Isard, M. Timely rollback: Specification and verification. In *Proc. NASA Formal Methods* (April 2015), 19–34.
3. Akidau, T., Balikov, A., Bekiroğlu, K., Chernyak, S., Haberman, J., Lax, R., McVeety, S., Mills, D., Nordstrom, P., Whittle, S. MillWheel: Fault-tolerant stream processing at internet scale. *Proc. VLDB Endow.* 6, 11 (Aug. 2013), 1033–1044.
4. Chandramouli, B., Goldstein, J., Maier, D. On-the-fly progress detection in iterative stream queries. *Proc. VLDB Endow.* 2, 1 (Aug. 2009), 241–252.
5. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E. Bigtable: A distributed storage system for structured data. In *Proc. OSDI* (Nov. 2006), 205–218.
6. Dean, J., Ghemawat, S. Mapreduce: Simplified data processing on large clusters. *Commun. ACM* 51, 1 (Jan. 2008), 107–113.
7. DeWitt, D., Gray, J. Parallel database systems: The future of high performance database systems. *Commun. ACM* 35, 6 (June 1992), 85–98.
8. Ewen, S., Tzoumas, K., Kaufmann, M., Markl, V. Spinning fast iterative data flows. *Proc. VLDB Endow.* 5, 11 (July 2012), 1268–1279.
9. Gog, I., Giceva, J., Schwarzkopf, M., Vaswani, K., Vytiniotis, D., Ramalingam, G., Costa, M., Murray, D.G., Hand, S., Isard, M. Broom: Sweeping out garbage collection from big data systems. In *Proc. HotOS* (May 2015).
10. Gog, I., Schwarzkopf, M., Crooks, N., Grosvenor, M.P., Clement, A., Hand, S. Musketeer: All for one, one for all in data processing systems. In *Proc. EuroSys* (Apr. 2015).
11. Gonzalez, J.E., Low, Y., Gu, H., Bickson, D., Guestrin, C. PowerGraph: Distributed graph-parallel computation on natural graphs. In *Proc. OSDI* (Oct. 2012), 17–30.
12. Gonzalez, J.E., Xin, R.S., Dave, A., Crankshaw, D., Franklin, M.J., Stoica, I. GraphX: Graph processing in a distributed dataflow framework. In *Proc. OSDI* (Oct. 2014), 599–613.
13. Isard, M., Budi, M., Yu, Y., Birrell, A., Fetterly, D. Dryad: Distributed data-parallel programs from sequential building blocks. In *Proc. EuroSys* (Mar. 2007), 59–72.
14. Lee, E., Messerschmitt, D.G. Synchronous data flow. *Proc. IEEE* 75, 9 (1987), 1235–1245.
15. Li, M., Andersen, D.G., Park, J.W., Smola, A.J., Ahmed, A., Josifovski, V., Long, J., Shekita, E.J., Su, B.-Y. Scaling distributed machine learning with the parameter server. In *Proc. OSDI* (Oct. 2014), 583–598.
16. McSherry, F., Isard, M., Murray, D.G. Scalability! But at what COST? In *Proc. HotOS* (May 2015).
17. McSherry, F., Murray, D.G., Isaacs, R., Isard, M. Differential dataflow. In *Proc. CIDR* (Jan. 2013).
18. Melnik, S., Gubarev, A., Long, J.J., Romer, G., Shivakumar, S., Tolton, M., Vassilakis, T. Dremel: Interactive analysis of web-scale datasets. *Proc. VLDB Endow. Proc. VLDB Endow.* 3, 1–2 (Sep. 2010), 330–339.
19. Murray, D.G., McSherry, F., Isaacs, R., Isard, M., Barham, P., Abadi, M. Naiad: A timely dataflow system. In *Proc. SOSP* (Nov. 2013), 439–455.
20. Murray, D.G., Schwarzkopf, M., Smowton, C., Smith, S., Madhavapeddy, A., Hand, S. CIEL: A universal execution engine for distributed data-flow computing. In *Proc. NSDI* (Mar. 2011), 113–126.
21. Peng, D., Dabek, F. Large-scale incremental processing using distributed transactions and notifications. In *Proc. OSDI* (Oct. 2010), 251–264.
22. Sousa, M., Dillig, I., Vytiniotis, D., Dillig, T., Gkantsidis, C. Consolidation of queries with user-defined functions. In *Proc. PLDI* (June 2014), 554–564.
23. Tel, G., Mattern, F. The derivation of distributed termination detection algorithms from garbage collection schemes. *ACM Trans. Program. Lang. Syst.* 15, 1 (Jan. 1993), 1–35.
24. Tucker, P.A., Maier, D., Sheard, T., Fegaras, L. Exploiting punctuation semantics in continuous data streams. *IEEE Trans. Knowledge Data Eng.* 15, 3 (2003), 555–568.
25. Yu, Y., Gunda, P.K., Isard, M. Distributed aggregation for data-parallel computing: Interfaces and implementations. In *Proc. SOSP* (Oct. 2009), 247–260.
26. Yu, Y., Isard, M., Fetterly, D., Budi, M., Erlingsson, U., Gunda, P.K., Currey, J. DryadLINQ: A system for general-purpose distributed data-parallel computing using a high-level language. In *Proc. OSDI* (Dec. 2008), 1–14.
27. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M., Shenker, S., Stoica, I. Resilient Distributed Datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proc. NSDI* (Apr. 2012).

Derek G. Murray, Michael Isard, Rebecca Isaacs, Paul Barham, and Martín Abadi (dmurray, misard, risaacs, pbar, abadi@google.com) Google, Mountain View, CA.

Frank McSherry (fmcsherry@me.com) is still at large.

Copyright held by owners/authors.



A personal walk down the computer industry road. BY AN EYEWITNESS.

Smarter Than Their Machines: Oral Histories of the Pioneers of Interactive Computing is based on oral histories archived at the Charles Babbage Institute, University of Minnesota. These oral histories contain important messages for our leaders of today, at all levels, including that government, industry, and academia can accomplish great things when working together in an effective way.



ISBN: 978-1-62705-550-5 DOI: 110.1145/2663015
<http://books.acm.org>
<http://www.morganclaypoolpublishers.com/acm>

Technical Perspective

The Power of Parallelizing Computations

By James Larus

AS COMPUTERS BECOME parallel, performance-challenged programs must also become parallel. For some algorithms, this is not a great challenge as the underlying problem divides naturally into independent pieces that can be computed concurrently. Other problems are not so lucky. Their constituent computations are tightly interdependent, and a parallel implementation requires considerable coordination and synchronization and may still perform poorly.

Recursive algorithms fall into this category. The recursive call is a dependence between the calculations in successive function invocations, which makes it difficult to overlap their executions significantly. There is no general formula for transforming a recursive function for parallel execution. It is necessary to understand the intrinsic structure of the underlying computation and to find a way to preserve the essential relationships while running in parallel.

The following paper shows how some instances of dynamic programming, an important recursive algorithmic paradigm, can be effectively parallelized by taking advantage of the algebraic properties of the underlying computations. Dynamic programming uses a table to store the values of subcomputations and consults these intermediate results to compute new values. Depending on which values are accessed, it is possible to execute some calculations concurrently along a column or diagonal of the table. However, each of these calculations is typically small. The overhead of communication and synchronization limits which computations can profitably execute in parallel. In addition, the amount of parallelism is limited by the size of the table.

The authors demonstrate another way to parallelize these computations by dividing the table into independent chunks, each of which can be

computed independently, and then these intermediate results “patched” to correctly account for missing dependencies from calculations that should have been performed earlier. The benefits of this approach are clear—large independent computations execute well on multicore processors and incur less overhead than fine-grained computation.

A dynamic programming algorithm is a sequence of recursive calls, where each such call is a subcomputation on a prefix of the input that produces a result for the computation on the current input. The table memorizes subcomputations so they need be computed only once.

Suppose we want to execute this sequence of calls concurrently on P processors. If we divide the sequence into P chunks and run each independently, all computations except the first one are likely to produce a wrong answer, since they lack the results that should have been put in the table earlier. This paper shows how to fix these incorrect answers for an important class of problems.

The method applies to dynamic programming problems in which the computation can be described by a tropical semiring, an algebraic struc-

ture with two operators and a zero element. For dynamic programming, the semiring is defined over matrices, with the standard matrix product redefined by replacing multiplication with addition and addition with max. The key insight of this paper is that sequentially applying this matrix product operation never increases the rank of the result matrix and, in practice, a sequence of these operations often converges to a rank-1 matrix. At this point, the final result of the sequence of matrix products is parallel to the rank-1 intermediate results, differing only in magnitude.

This insight leads to an efficient coarse-grained parallelization. Break this sequence into P independent computations, each starting on a contiguous block of product computations. Each computation, except the first one, may be wrong since it ignores the earlier computations. However, they can be fixed by sequentially propagating correct results from the prior computation and redoing the product calculation until it produces a rank-1 matrix, at which point the rest of the calculations can be skipped since the final result differs only by an easily calculable offset.

In practice, for many problems, convergence to rank-1 is very quick; in others, it is slower or never occurs. But, in the cases where convergence is rapid (for example, Viterbi and Smith-Waterman) and the input is large, the resulting algorithm performs very well, even producing near-linear speedup on the latter problem for greater than 100 cores.

This paper is a nice reminder of the value of looking beyond the natural formulation of a computation to its underlying structure when a program does not naturally parallelize. 

This paper is a nice reminder of the value of looking beyond the natural formulation of a computation to its underlying structure.

James Larus (james.larus@epfl.ch) is a professor and Dean of Computer and Communications Sciences at EPFL, Lausanne, Switzerland.

Copyright held by author.

Efficient Parallelization Using Rank Convergence in Dynamic Programming Algorithms

By Saeed Maleki, Madanlal Musuvathi, and Todd Mytkowicz

Abstract

This paper proposes an efficient parallel algorithm for an important class of dynamic programming problems that includes Viterbi, Needleman–Wunsch, Smith–Waterman, and Longest Common Subsequence. In dynamic programming, the subproblems that do not depend on each other, and thus can be computed in parallel, form stages, or *wavefronts*. The algorithm presented in this paper provides additional parallelism allowing multiple stages to be computed in parallel despite dependences among them. The correctness and the performance of the algorithm relies on rank convergence properties of matrix multiplication in the tropical semiring, formed with plus as the multiplicative operation and max as the additive operation.

This paper demonstrates the efficiency of the parallel algorithm by showing significant speedups on a variety of important dynamic programming problems. In particular, the parallel Viterbi decoder is up to 24× faster (with 64 processors) than a highly optimized commercial baseline.

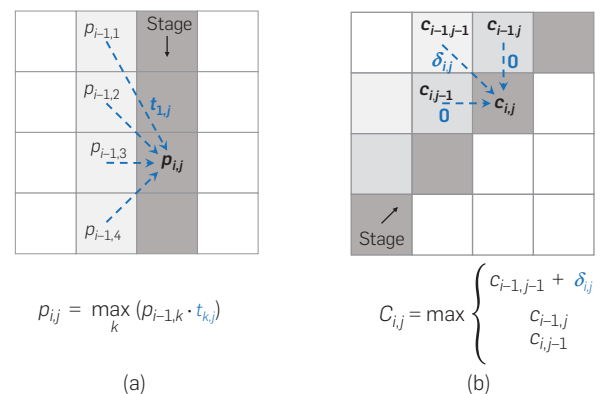
1. INTRODUCTION

Dynamic programming² is a method to solve a variety of important optimization problems in computer science, economics, genomics, and finance. Figure 1 describes two such examples: Viterbi,²³ which finds the most-likely path through a hidden-Markov model for a sequence of observations, and LCS,¹⁰ which finds the longest common subsequence between two input strings. Dynamic programming algorithms proceed by recursively solving a series of sub-problems, usually represented as cells in a table as shown in the figure. The solution to a subproblem is constructed from solutions to an appropriate set of subproblems, as shown by the respective recurrence relation in the figure.

These data-dependences naturally group subproblems into *stages* whose solutions do not depend on each other. For example, all subproblems in a column form a stage in Viterbi and all subproblems in an anti-diagonal form a stage in LCS. A predominant method for parallelizing dynamic programming is *wavefront* parallelization,¹⁵ which computes all subproblems within a stage in parallel.

In contrast, this paper breaks data-dependences *across* stages and fixes up incorrect values later in the algorithm. Therefore, this approach exposes parallelism for a class of dynamic programming algorithms we call *linear-tropical dynamic programming* (LTDP). A LTDP computation can be viewed as performing a sequence of matrix multiplications in the tropical semiring where the semiring is formed

Figure 1. Dynamic programming examples with dependences between stages. (a) the Viterbi algorithm and (b) the LCS algorithm.



with addition as the multiplicative operator and max as the additive operator. This paper demonstrates that several important optimization problems such as Viterbi, LCS, Smith–Waterman, and Needleman–Wunsch (the latter two are used in bioinformatics for sequence alignment) belong to LTDP. To efficiently break data-dependences across stages, the algorithm uses *rank convergence*, a property by which the rank of a sequence of matrix products in the tropical semiring is likely to converge to 1.

A key advantage of our parallel algorithm is its ability to simultaneously use both the coarse-grained parallelism across stages and the fine-grained wavefront parallelism within a stage^a. Moreover, the algorithm can reuse existing highly optimized implementations that exploit wavefront parallelism with little modification. As a consequence, our implementation achieves multiplicative speedups over existing implementations. For instance, the parallel Viterbi decoder is up to 24× faster with 64 cores than a state-of-the-art commercial baseline.¹⁸ This paper demonstrates similar speedups for other LTDP instances.

^a The definition of wavefront parallelism used here is more general and includes the common usage where a wavefront performs computations across logical iterations as in the LCS example in Figure 1a.

The original version of this paper is entitled “Parallelizing Dynamic Programming through Rank Convergence” and was published in ACM SIGPLAN Notices – PPOPP ‘14, August 2014, ACM.

2. BACKGROUND

2.1. Tropical semiring

An important set of dynamic programming algorithms can be expressed in an algebra known as the *tropical semiring*. The tropical semiring has two binary operators: $\oplus$ where $x \oplus y = \max(x, y)$, and $\otimes$ where $x \otimes y = x + y$ for all x and y in the domain. The domain of the tropical semiring is $\{\mathbb{R} \cup \{-\infty\}\}$, the set of real numbers extended with $-\infty$, which serves as the 0 of the semiring, meaning that $0 \oplus x = x \oplus 0 = x$ and $0 \otimes x = x \otimes 0 = 0$. Most properties of ordinary algebra also hold in the tropical semiring, allowing it to support an algebra of matrices over elements of the semiring. For a more detailed discussion of the tropical semiring refer to Section 2 in Ref.¹³

2.2. Matrix multiplication

Let $A_{n \times m}$ denote a matrix with n rows and m columns with elements from the domain of the tropical semiring. Let $A[i, j]$ denote the element of A at the i th row and j th column. The matrix product of $A_{l \times m}$ and $B_{m \times n}$ is $A \odot B$, an $l \times n$ matrix defined such that

$$\begin{aligned} (A \odot B)[i, j] &= \bigoplus_{1 \leq k \leq m} (A[i, k] \otimes B[k, j]) \\ &= \max_{1 \leq k \leq m} (A[i, k] + B[k, j]) \end{aligned}$$

Note, this is the standard matrix product with multiplication replaced by $+$ and addition replaced by $\max$.

The transpose of $A_{n \times m}$ is the matrix $A_{m \times n}^T$ such that $\forall i, j: A^T[i, j] = A[j, i]$. Using standard terminology, we will denote a $v_{n \times 1}$ matrix as the column vector $\vec{v}$, a $v_{1 \times n}$ matrix as the row vector $\vec{v}^T$, and an $x_{1 \times 1}$ matrix simply as the scalar x (in which case, by a conventional use of notation, we identify x as $x_{1,1}$). This terminology allows us to extend the definition of matrix-matrix multiplication above to matrix-vector and vector-scalar multiplication appropriately. Also, $\vec{v}[i]$ is the i th element of a vector $\vec{v}$. It is easy to check that matrix multiplication is associative in the tropical semiring: $(A \odot B) \odot C = A \odot (B \odot C)$.

2.3. Parallel vectors

Two vectors $\vec{u}$ and $\vec{v}$ are parallel in the tropical semiring, denoted as $\vec{u} \parallel \vec{v}$, if there exist non- 0 scalars x and y such that $\vec{u} \odot x = \vec{v} \odot y$. Intuitively, parallel vectors $\vec{u}$ and $\vec{v}$ in the tropical semiring differ by a constant offset. For instance, $[1 \ 0 \ 2]^T$ and $[3 \ 2 \ 4]^T$ are parallel vectors differing by an offset 2.

2.4. Matrix rank

The rank of a matrix $M_{m \times n}$, denoted by $\text{rank}(M)$, is the smallest number r such that there exist matrices $C_{m \times r}$ and $R_{r \times n}$ whose product is M . In particular, a rank-1 matrix is a product of a column vector and a row vector. (There are alternate ways to define the rank of a matrix in semirings, such as the number of linearly independent rows or columns in a matrix. While such definitions coincide in ordinary linear algebra, they are not equivalent in arbitrary semirings.⁴)

LEMMA 1. *For any vectors $\vec{u}$ and $\vec{v}$, and a matrix A of rank 1, $A \odot \vec{u} \parallel A \odot \vec{v}$*

Intuitively, this lemma states that a rank-1 matrix maps all vectors to a line. If $\text{rank}(A) = 1$ then it is the product of a column

vector $\vec{c}$ and a row vector $\vec{r}^T$. For any vectors $\vec{u}$ and $\vec{v}$:

$$A \odot \vec{u} = (\vec{c} \odot \vec{r}^T) \odot \vec{u} = \vec{c} \odot (\vec{r}^T \odot \vec{u}) = \vec{c} \odot x_u$$

$$A \odot \vec{v} = (\vec{c} \odot \vec{r}^T) \odot \vec{v} = \vec{c} \odot (\vec{r}^T \odot \vec{v}) = \vec{c} \odot x_v$$

for appropriate scalars x_u and x_v . As an example, consider

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{bmatrix} \quad \vec{u} = \begin{bmatrix} 1 \\ -\infty \\ 3 \end{bmatrix} \quad \vec{v} = \begin{bmatrix} -\infty \\ 2 \\ 0 \end{bmatrix}$$

$A = [1 \ 2 \ 3]^T \odot [0 \ 1 \ 2]$ is rank-1. $A \odot \vec{u} = [6 \ 7 \ 8]^T$ and $A \odot \vec{v} = [4 \ 5 \ 6]^T$ which are parallel with a constant offset 2. Also note that all rows in a rank-1 matrix are parallel to each other.

3. LINEAR-TROPICAL DYNAMIC PROGRAMMING

Dynamic programming is a method for solving problems that have optimal substructure—the solution to a problem can be obtained from the solutions to a set of its overlapping subproblems. This dependence between subproblems is captured by a recurrence equation. Classic dynamic programming implementations solve the subproblems iteratively applying the recurrence equation in an order that respects the dependence between subproblems.

3.1. LTDP definition

A dynamic programming problem is LTDP if (a) its solution and the solutions of its subproblems are in the domain of the tropical semiring, (b) the subproblems can be grouped into a sequence of stages such that the solution to a subproblem in a stage depends on only the solutions in the previous stage, and (c) this dependence is linear in the tropical semiring. In other words, $s_i[j]$, the solution to subproblem j in stage i of LTDP, is given by the recurrence equation

$$s_i[j] = \max_k (s_{i-1}[k] + A_i[j, k]) \quad (1)$$

for appropriate constants $A_i[j, k]$. This linear dependence allows us to view LTDP as computing, from an initial solution vector $\vec{s}_0$ (obtained from the base case for the recurrence equation), a sequence of vectors $\vec{s}_1, \vec{s}_2, \dots, \vec{s}_n$, where the vectors need not have the same length, and

$$\vec{s}_i = A_i \odot \vec{s}_{i-1} \quad (2)$$

for appropriate matrices of constants A_i derived from the recurrence equation. We will call $\vec{s}_i$ the *solution vector* at stage i and call A_i the *transformation matrix* at stage i .

3.2. Backward phase

Once all the subproblems are solved, finding the solution to the underlying LTDP optimization problem usually involves tracing the *predecessors* of subproblems backward. A predecessor of a subproblem is the subproblem for which the maximum in Equation (1) is reached. For ease of exposition, we define the *predecessor product* of a matrix A and a vector $\vec{s}$ as the vector $A \star \vec{s}$ such that

$$(A \star \vec{s})[j] = \arg \max_k (\vec{s}[k] + A[j, k])$$

Note the similarity between this definition and Equation (1).

We assume that ties in $\arg \max$ are broken deterministically. The following lemma shows that predecessor products do not distinguish between parallel vectors, a property that will be useful later.

LEMMA 2. $\vec{u} \parallel \vec{v} \Rightarrow \forall A: A \star \vec{u} = A \star \vec{v}$

This follows from the fact that parallel vectors in the tropical semiring differ by a constant and that $\arg \max$ is invariant when a constant is added to all its arguments.

3.3. Sequential LTDP

The *sequential* algorithm for LTDP can be phrased in terms of matrix multiplications using Equations 2. Assume that the LTDP problem has $n+1$ stages $\vec{s}_0, \vec{s}_1, \dots, \vec{s}_n$ where $\vec{s}_0$ is the initial solution vector given as an input argument and $\forall i: 1 \leq i \leq n, \vec{s}_i$ are the desired output solutions. The output solutions can be computed using the transformation matrices $A_1, \dots, A_n$ which capture the inductive case of the LTDP recurrence as shown in Equation (2). The sequential algorithm consists of a forward phase and a backward phase.

The forward phase computes the solution iteratively in n iterations. In iteration i , it computes $\vec{s}_i = A_i \odot \vec{s}_{i-1}$ and $\vec{p}_i = A_i \star \vec{s}_{i-1}$. This algorithm is deemed sequential because it computes the stages one after the other due to the data-dependence across stages.

The backward phase iteratively follows the predecessors of solutions computed in the forward phase. Depending on the LTDP problem, one of the subproblems in the last stage, say $s_n[opt]$, contains the optimal solution. Then the solutions along the optimal path for the LTDP problem are

$$\begin{aligned} & s_n[opt], \\ & s_{n-1}[p_n[opt]], \\ & s_{n-2}[p_{n-1}[p_n[opt]]], \\ & \dots, \\ & s_1[p_2[p_3[\dots[p_{n-1}[p_n[opt]]\dots]]], \\ & s_0[p_1[p_2[p_3[\dots[p_{n-1}[p_n[opt]]\dots]]]] \end{aligned}$$

These are easily computed by in linear time by tracing the path backwards from stage n to stage 0.

The exposition above consciously hides a lot of details in the $\odot$ and $\star$ operators. An implementation does not need to represent the solutions in a stage as a (dense) vector and perform (dense) matrix-vector operations. It might statically know that the current solution depends on only some of the subproblems in the previous stage (a sparse matrix) and only accesses those. Moreover, as mentioned above, an implementation might use wavefront parallelism to compute the solutions in a stage in parallel. Finally, implementations can use techniques such as tiling to improve the cache-efficiency of the sequential computation. All these implementation details are orthogonal to how the parallel algorithm described in this paper parallelizes across stages.

4. PARALLEL LTDP ALGORITHM

This section describes an efficient parallel algorithm for the sequential algorithm described in Section 4 across stages.

4.1. Breaking data-dependences across stages

Viewing LTDP computation as matrix multiplication in the tropical semiring provides a way to break data-dependences among stages. Consider the solution vector at the last stage $\vec{s}_n$. From Equation (2), we have

$$\vec{s}_n = A_n \odot A_{n-1} \odot \dots \odot A_2 \odot A_1 \odot \vec{s}_0$$

Standard techniques^{9, 12} can parallelize this computation using the associativity of matrix multiplication. For instance, two processors can compute the partial products $A_{hi} = A_n \odot \dots \odot A_{\lceil n/2 \rceil + 1}$ and $A_{lo} = A_{\lceil n/2 \rceil} \odot \dots \odot A_1$ in parallel, and then compute $A_{hi} \odot A_{lo} \odot \vec{s}_0$ to obtain $\vec{s}_n$.

However, doing so converts a sequential computation that performs matrix-vector multiplications (working from right to left) to a parallel computation that performs matrix-matrix multiplications. This results in a parallelization overhead linear in the size of the stages and thus requires a linear number of processors to observe a constant speedup. In practice, the size of each stage can easily be hundreds or larger and thus is not practical on real problems and hardware.

The key contribution of this paper is a parallel algorithm that avoids the overhead of matrix-matrix multiplications. This algorithm relies on the convergence of matrix rank in the tropical semiring as discussed below. Its exposition requires the following definition: For a given LTDP instance, the *partial product* $M_{i \rightarrow j}$, defined for stages $i \leq j$, is given by

$$M_{i \rightarrow j} = A_j \odot A_{j-1} \odot \dots \odot A_{i+1} \odot A_i$$

Note that $M_{i \rightarrow j}$ describes how stage j depends on stage i , because $\vec{s}_j = M_{i \rightarrow j} \odot \vec{s}_i$.

4.2. Rank convergence

The rank of the product of two matrices is not greater than the rank of either input matrix.

$$\text{rank}(A \odot B) \leq \min(\text{rank}(A), \text{rank}(B)) \quad (3)$$

This is because, if $\text{rank}(A) = r$, then $A = C \odot R$ for some matrix C with r columns. Thus, $A \odot B = (C \odot R) \odot B = C \odot (R \odot B)$ implying that $\text{rank}(A \odot B) \leq \text{rank}(A)$. A similar argument shows that $\text{rank}(A \odot B) \leq \text{rank}(B)$.

Equation 3 implies that for stages $k \geq j \geq i$

$$\text{rank}(M_{i \rightarrow k}) \leq \text{rank}(M_{i \rightarrow j}) \leq \text{rank}(A_i)$$

In effect, as the LTDP computation proceeds, the rank of the partial products will never increase. Theoretically, there is a possibility that the ranks do not decrease. However, we have observed this only for carefully crafted problem instances that are unlikely to occur in practice. On the contrary, the rank of these partial products is likely to converge to 1, as will be demonstrated in Section 6.1.

Consider a partial product $M_{i \rightarrow j}$ whose rank is 1. Intuitively, this implies a *weak* dependence between stages i and j . Instead of the actual solution vector $\vec{s}_j$, say the LTDP computation starts with a different vector $\vec{t}_i$ at stage i . From Lemma 1, the new solution vector at stage j , $\vec{t}_j = M_{i \rightarrow j} \odot \vec{t}_i$, is parallel to the actual solution vector $\vec{s}_j = M_{i \rightarrow j} \odot \vec{s}_i$. Essentially, the direction of

the solution vector at stage j is independent of stage i ; stage i determines only its magnitude. In the tropical semiring, where the multiplicative operator is $+$, this means that the solution vector at stage j will be, at worst, off by a constant if one starts stage i with an arbitrary vector.

4.3. Parallel forward phase

The parallel algorithm uses this insight to break dependences between stages, as shown pictorially in Figure 2. The figure uses three processors, P_0 , P_1 , and P_2 , and six stages for each processor as an example, for a total of 18 stages beyond $\vec{s}_0$. Figure 2a represents the forward phase of the sequential algorithm described in Section 4. Each stage is represented as a vertical column of cells. (For pictorial simplicity, we assume each solution vector has length 4, but in general they might have different lengths.) Note that P_0 also contains the initial solution $\vec{s}_0$; note also that stage $\vec{s}_6$ is shared between P_0 and P_1 , and similarly stage $\vec{s}_{12}$ is shared between P_1 and P_2 . These replicated stages are differentiated by dotted borders. Each stage $\vec{s}_i$ is computed by multiplying $\vec{s}_{i-1}$ by the transformation matrix A_i (Equation 2). Processor P_0 starts from the initial solution vector $\vec{s}_0$ and computes all its stages. As indicated by the arrow on the left, processor P_1 waits for P_0 to compute the shared stage $\vec{s}_6$ in order to start its computation. Similarly, processor P_2 waits for P_1 to compute the shared stage $\vec{s}_{12}$ as the arrow on the right shows.

In the parallel algorithm shown in Figure 2b, processors P_1 and P_2 start from arbitrary solutions $\vec{r}_6$ and $\vec{r}_{12}$ respectively in parallel with P_0 . Of course, the solutions for the stages computed by P_1 and P_2 will start out as completely wrong (shaded dark in the figure). However, if rank convergence occurs, then these erroneous solution vectors will eventually become parallel to the actual solution vectors (shaded gray in the figure). Thus, P_1 will generate some solution vector $\vec{s}_{12}$ parallel to $\vec{s}_{12}$.

In a subsequent *fixup phase*, shown in Figure 2c, P_1 uses $\vec{s}_6$ computed by P_0 , and P_2 uses $\vec{s}_{12}$ computed by P_1 , to fix stages that are not parallel to the actual solution vector at that

stage. The fixup phase terminates when a newly computed stage is parallel to (off by a constant from) the computed one from the forward phase. (If rank convergence had not occurred with P_1 , then in the fixup phase, P_1 might have had to fix all of its stages including $\vec{s}_{12}$ since it is not parallel to the $\vec{s}_{12}$ it had computed from $\vec{r}_6$. In such a case, P_2 would need to re-perform its fixup phase. Thus, in the worst case, the parallel algorithm reduces gracefully to the serial algorithm.) After the fixup, the solution vectors at each stage are either the same as or parallel to the actual solution vector at those respective stages.

For LTDP, it is not necessary to compute the actual solution vectors. Because parallel vectors generate the same predecessor products (Lemma 2), following the predecessors in Figure 2c will generate the same solution as the following the predecessors in Figure 2a.

A general version of this algorithm with pseudocode is discussed in Section 4 of Ref.¹³ For correctness (see Section 4.4 in Ref.¹³), the parallel algorithm requires that every entry in the arbitrary vectors ($\vec{r}_6$ and $\vec{r}_{12}$ in Figure 2) be non-0. The backward phase can also be parallelized using ideas similar to those used in the forward phase (see Section 4.6 in Ref.¹³).

4.4. Rank convergence discussion

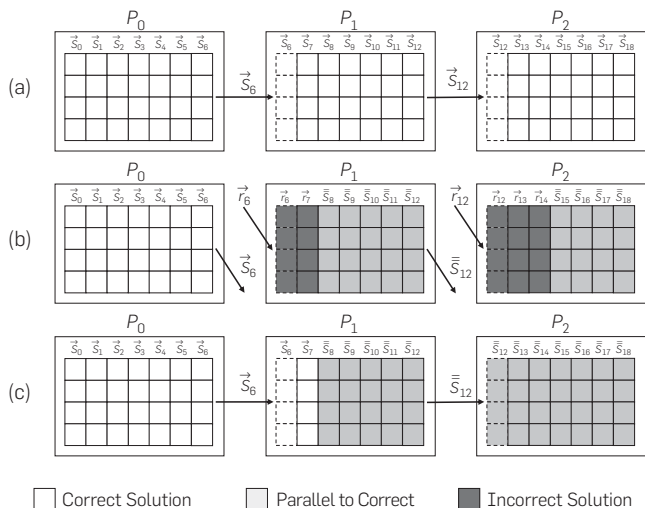
One can view solving a LTDP problem as computing shortest (or longest) paths in a graph. In this graph, each subproblem is a node and directed edges represent the dependences between subproblems. The weights on edges represent the constants $A_i[j, k]$ in Equation 1. In LCS (Figure 1b), each subproblem has incoming edges with weight 0 from the subproblems above it and to its left, and an incoming edge with weight $\delta_{i,j}$ from its diagonal neighbor. Finding the optimal solution to the LTDP problem amounts to finding the longest path in this graph from $C_{0,0}$ to $C_{m,n}$ (where m and n are the lengths of the two input strings). Alternatively, one can negate all the weights and change the max to a min in Equation (1) to view this as computing shortest paths.

Entries in the partial product $M_{l,r}$ represent the cost of the shortest (or longest) path from a node in stage l to a node in stage r . The rank of this product is 1 if these shortest paths all go through a single node in some stage between l and r . Road networks have this property. For instance, the fastest path from any city in Washington state to any city in Massachusetts is highly likely to go through Chicago, because routes that do not; choices of the cities at the beginning and at the end do not drastically change how intermediate stages are routed. Similarly, if problem instances have optimal solutions that are overwhelmingly better than others solutions, one should expect rank convergence.

5. LTDP EXAMPLES

This section shows LTDP solutions for four important optimization problems: Viterbi, Longest Common Subsequence, Smith–Waterman, and Needleman–Wunsch. Our goal in choosing these particular problems is to provide an intuition on how problems with different structure can be viewed as LTDP. Other applications for LTDP, not evaluated in this paper, include dynamic time warping and seam carving.

Figure 2. Parallelization algorithm using rank convergence. (a) the sequential forward phase, (b) the parallel forward phase, and (c) the fixup phase.



5.1. Viterbi

The Viterbi algorithm²³ finds the most likely sequence of states in a (discrete) hidden Markov model (HMM) for a given sequence of n observations. Its recurrence equation is shown in Figure 1a (refer to Ref.²³ for the meaning of the $p_{i,j}$ and $t_{k,j}$ terms). The subproblems along a column in the figure form a stage and they only depend on subproblems in the previous column. This dependence is not directly in the desired form of Equation (1), but applying the logarithm function to both sides of the recurrence equation brings it to this form. By transforming the Viterbi instance into one that calculates log-probabilities instead of probabilities, we obtain a LTDP instance.

5.2. Longest common subsequence

LCS finds the longest common subsequence of two input strings A and B .¹⁰ The recurrence equation of LCS is shown in Figure 1b. Here, $C_{i,j}$ is the length of the longest common subsequence of the first i characters of A and the first j characters of B (where adjacent characters of a subsequence need not be adjacent in the original sequence, but must appear in the same order). Also, $\delta_{i,j}$ is 1 if the i th character of A is the same as the j th character of B and 0 otherwise. The LCS of A and B is obtained by following the predecessors from the bottom-rightmost entry in the table in Figure 1b.

Some applications of LCS, such as the `diff` utility tool, are only interested in solutions that are at most a width w away from main diagonal, ensuring that the LCS is still reasonably similar to the input strings. For these applications, the recurrence relation can be modified such that $C_{i,j}$ is set to $-\infty$ whenever $|i - j| > w$. In effect, this modification limits the size of each stage δ_p , which in turn limits wavefront parallelism, increasing the need to execute multiple stages in parallel as we propose here.

Grouping the subproblems of LCS into stages can be done in two ways, as shown in Figure 3. In the first approach, the stages correspond to anti-diagonals, such as the stage consisting of z 's in Figure 3a. This stage depends on two previous stages (on x 's and y 's) and does not strictly follow the rules of LTDP. One way to get around this is to define stages as overlapping pairs of anti-diagonals, like stage x - y and stage y - z in

Figure 3. Two ways of grouping the subproblems in LCS into stages so that each stage depends on only one previous stage. (a) anti-diagonal grouping and (b) row grouping.

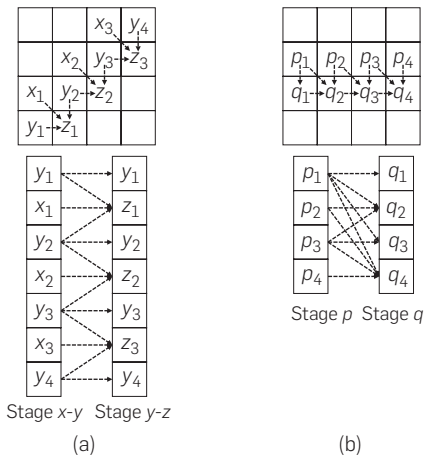


Figure 3a. Subproblems y 's are replicated in both stages, allowing stage y - z to depend only on stage x - y . While this representation has the downside of doubling the size of each stage, it can sometimes lead to efficient representation. For LCS, one can show that the difference between solutions to consecutive subproblems in a stage is either 1 or 0. This allows compactly representing the stage as a sequence of bits.¹¹

In the second approach, the stages correspond to the rows (or columns) as shown in Figure 3b. The recurrence needs to be unrolled to avoid dependences between subproblems within a stage. For instance, q_i depends on all p_j for $j \leq i$. In this approach, since the final solution is obtained from the last entry, the predecessor traversal in the backward phase has to be modified to start from this entry, say by adding an additional matrix at the end to move this solution to the first solution in the added stage.

5.3. Needleman-Wunsch

This algorithm¹⁷ finds a *global* alignment of two input sequences, commonly used to align protein or DNA sequences. The recurrence equation is very similar to the one in LCS.

$$s_{i,j} = \max \begin{cases} s_{i-1,j-1} + m[i,j] \\ s_{i-1,j} - d \\ s_{i,j-1} - d \end{cases}$$

In this equation, $s_{i,j}$ is the score of the best alignment for the prefix of length i of the first input and the prefix of length j of the second input, $m[i,j]$ is the matching score for aligning the last characters of the respective prefixes, and d is the penalty for an insertion or deletion during alignment. The base cases are defined as $s_{i,0} = -i \cdot d$ and $s_{0,j} = -j \cdot d$. Also, grouping subproblems into stages can be done using the same approach as in LCS.

5.4. Smith-Waterman

This algorithm¹⁹ performs a *local* sequence alignment, in contrast to Needleman-Wunsch. Given two input strings, Smith-Waterman finds the *substrings* of the input that have the best alignment, where longer substrings have a better alignment. In its simplest form, the recurrence equation is of the form

$$s_{i,j} = \max \begin{cases} 0 \\ s_{i-1,j-1} + m[i,j] \\ s_{i-1,j} - d \\ s_{i,j-1} - d \end{cases}$$

The key difference from Needleman-Wunsch is the 0 term in max which ensures that alignments “restart” whenever the score goes below zero. Because of this term, the constants in the A_i matrices in Equation 1 need to be set accordingly. This slight change significantly affects the convergence properties of Smith-Waterman (see Section 6.1).

6. EVALUATION

This section evaluates the parallel LTDP algorithm on the four problems discussed in Section 5.

6.1. LTDP rank convergence

Determining whether the LTDP parallel algorithm benefits a dynamic programming problem requires: (1) the

problem to be LTDP (discussed in Section 4) and (2) rank convergence to happen in a reasonable number of steps. This section demonstrates how rank convergence can be measured and evaluates it for the LTDP problems discussed in Section 5.

Rank convergence is an empirical property of a sequence of matrix multiplications that depends on both the LTDP recurrence relation and the input. Table 1 presents measurements of the number of steps required for rank convergence across different algorithms and inputs. For a LTDP instance, defined by the algorithm (column 1) and input (column 2), we first compute the actual solution vectors at each stage. Then, starting from a random all-non-zero vector at 200 different equally spaced stages, we measured the number of steps required to converge to a vector parallel to the actual solution vector. Columns 3, 4, and 5, respectively show the minimum, median, and maximum number of steps needed for convergence. For each input, column 2 specifies the computation width (the size of each stage). Each algorithm has a specific definition of width: for Viterbi decoder, width is the number of states for each decoder; in Smith–Waterman, it is the size of each query; and in LCS and Needleman–Wunsch, it is a fixed width around the diagonal of each stage. LCS in some cases never converged, so we left those entries blank. The rate of convergence is specific to the algorithm and input (e.g., Smith–Waterman converges fast while LCS sometimes does not converge) and, generally speaking, wider widths require more steps to converge. We use this table later in Section 6.3 to explain scalability of our approach.

6.2. Environmental setup

We conducted experiments on a shared-memory machine and on a distributed-memory machine. A shared-memory machine favors fast communication and is ideal for the wave-front approach. The distributed-memory machine has a larger number of processors, so we can better understand how our parallel algorithm scales. The shared-memory machine has 40 cores (Intel Xeon E7). The distributed-memory machine is called Stampede²¹; for our experiments we used up to 128 cores (Intel Xeon E5). See Ref.¹³ for more details.

6.3. Parallel LTDP benchmarks and performance

This section evaluates the parallel algorithm on four LTDP problems. To substantiate our scalability results, we evaluate each benchmark across a wide variety of real-world inputs. We break the results down by the LTDP problem. For each LTDP problem, we used the best existing sequential algorithm as our baseline, as described below. For the parallel LTDP algorithm, we implemented the algorithm described in Section 4 and used the sequential baseline implementation as a black box to advance from one stage to the next.

Viterbi decoder. As our baseline, we used Spiral’s¹⁸ Viterbi decoder, a highly optimized (via auto-tuning) decoder that utilizes SIMD to parallelize decoding within a stage. We use two real-world convolution codes, CDMA and LTE, which are commonly used in modern cell-phone networks. For each of these two convolution codes, we investigated the impact of four network packet sizes (2048, 4096, 8192, and 16,384), which determine the number of stages in the computation. For each size, we used Spiral’s input generator to create 50 network packets and considered the average performance.

Figure 4 shows the performance, the speedup, and the efficiency of the two decoders. To evaluate the impact of different decoder sizes, each plot has four lines (one per network packet size). A point (x, y) in a performance/speedup plot with the primary y -axis on left, gives the throughput y (the number of bits processed in a second) in megabits per second (Mb/s) as a function of the number of processors x used to perform the Viterbi Decoding. The same point with the secondary y -axis on right shows the speedup y with x number of processors over the sequential performance. Note that Spiral sequential performance at $x = 1$ is almost the same for different packet sizes. The filled data points in the plots indicate that convergence occurred in the first iteration of the fixup phase in the parallel LTDP algorithm described in Section 4 (i.e., each processor’s stage is large enough for convergence). The non-filled data points indicate that multiple iterations of the fixup loop were required.

Figure 4 demonstrates that (i) our approach provides significant speedups over the sequential baseline, and (ii) different convolution codes and network packet sizes have different performance characteristics. For example, with 64 processors, our CDMA Viterbi Decoder processing packets

Table 1. Number of steps to converge to rank 1.

Steps to converge to Rank-1		Min	Median	Max
Viterbi decoder	LTE: 2 ⁶	18	30	62
	CDMA: 2 ⁸	22	38	72
Smith–Waterman	Query-1: 603	2	6	24
	Query-2: 884	4	8	24
	Query-3: 1227	4	8	24
	Query-4: 1576	4	8	24
Needleman–Wunsch	Width: 1024	1580	19,483	192,747
	Width: 2048	3045	44,891	378,363
	Width: 4096	5586	101,085	404,437
	Width: 8192	12,005	267,391	802,991
LCS	Width: 8192	9142	79,530	370,927
	Width: 16,384	19,718	270,320	–
	Width: 32,768	42,597	626,688	–
	Width: 65,536	86,393	–	–

of size 16,384 decodes $\sim 24\times$ faster than the sequential algorithm while this number is $\sim 13\times$ for our LTE decoder. This difference is due to the fact that the amount of computation per bit in CDMA is four times as much as in LTE but the median of the convergence rate is almost the same (Table 1). Also, note that in Figure 4, larger network packet size provide better performance across all convolution codes (i.e., a network packet size of 16,384 is *always* the fastest implementation, regardless of convolution code) because the amount of re-computation (i.e., the part of the computation that has not converged), as a proportion of the overall computation, decreases with larger network packet size.

Smith–Waterman. Our baseline implements the fastest known CPU version, Farrar’s algorithm, which utilizes SIMD to parallelize within a stage.⁵ For data, we aligned chromosomes 1, 2, 3, and 4 from the human reference genome hg19 as databases and four randomly selected expressed sequence tags as queries. All the inputs are publicly available to download from Ref.¹⁶ We reported the average of performance across all combinations of DNA and query (16 in total).

A point (x, y) in the performance/speedup plot in Figure 5 with the primary y-axis on left, gives the performance y in giga cell updates per second (GigaCUPS) as a function of the number of processors used to perform the Smith–Waterman alignment. GigaCUPS is a standard metric used in bioinformatics to measure the performance of DNA-based sequence alignment problems and refers to the number of cells (in a dynamic programming table) updated per second. Similar to the Viterbi decoder plots, the secondary y-axis on the left shows the speedup for each number of processors.

Figure 4. Performance (Mb/S) and speedup of two Viterbi decoders. The non-filled data points indicate where processors have too few iterations to converge to rank 1

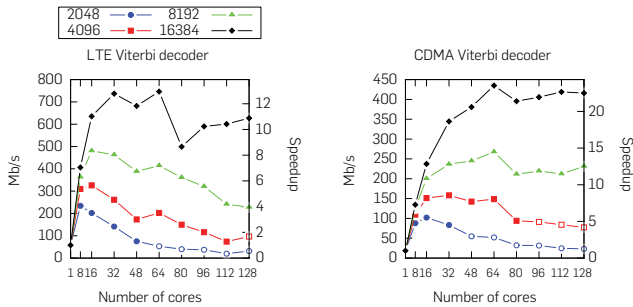
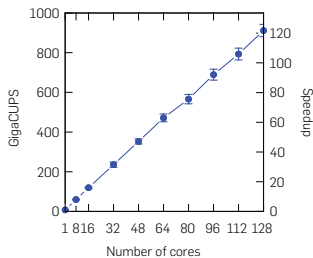


Figure 5. Smith–Waterman performance and speedup averages over four databases and four queries.



The performance gain of our approach for this algorithm is significant. As Figure 5 demonstrates, the parallel LTDP algorithm delivers almost linear speedup with up to 128 processors. Our algorithm scales well with even higher numbers of processors, but we report only up to 128 processors to keep Figure 5 consistent with the others.

Needleman–Wunsch. Our baseline utilized SIMD parallelization *within* a stage by using the grouping technique shown in Figure 3a. For data, we used two pairs of DNA sequences as inputs: Human Chromosomes (21, 22) and (X, Y) from the human reference genome hg19. We used only the first 1 million elements of each sequence since Stampede does not have enough memory on a single node to store the cell values for the complete chromosomes. We also used just 4 different widths (1024, 2048, 4096, and 8192) since we found that widths larger than 8192 do not affect the final alignment score.

Figure 6 shows the performance and speedup of the Needleman–Wunsch algorithm parallelized using our approach. The rank convergence is dependent on the input data; consequently, the parallel LTDP algorithm performs differently with pairs (X, Y) and (21, 22), as can be seen in the figure. The plots in the figure show results for each of the width sizes: 1024, 2048, 4096, and 8192. Similar to the Viterbi decoder benchmark, filled/non-filled data points show whether convergence occurred in the first iteration of the fixup phase.

In Figure 6, larger widths perform worse than smaller ones since the convergence rate depends on the size of each stage in a LTDP instance.

LCS. Our baseline adapts the fastest known single-core algorithm for LCS that exploits bit-parallelism to parallelize the computation *within* a column.^{3,11} This approach uses the grouping technique shown in Figure 3b. For data, we used the same input data as with Needleman–Wunsch except that we used the following 4 widths: 8192, 16,384, 32,768, and 65,536. We report performance numbers as for Needleman–Wunsch.

The performance and speedup plots in Figure 7 are very similar to those in Figure 6: The choice of input pair has a great impact on rank convergence, as can be seen in Figure 7.

LTDP versus Wavefront. We have also compared our LTDP parallel approach and the wavefront approach for LCS and Needleman–Wunsch. Refer to Section 6.4 in Ref.¹³ for details.

Figure 6. Performance and speedup of Needleman–Wunsch.

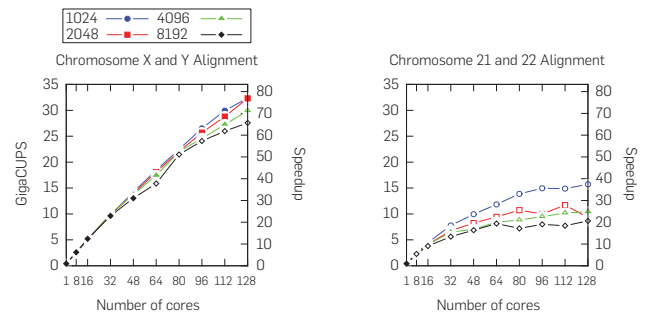
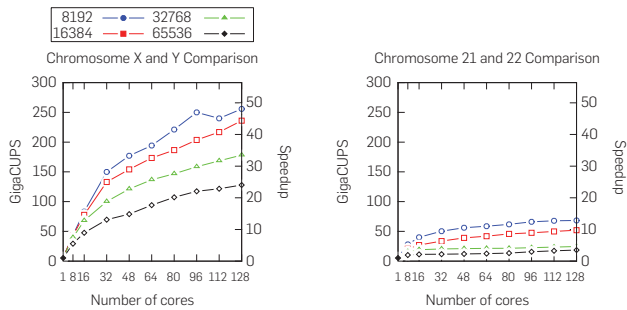


Figure 7. Performance and speedup results of Longest Common Subsequence.



7. RELATED WORK

Due to its importance, there is a lot of prior work on parallelizing dynamic programming. Predominantly, implementations use wavefront parallelism to parallelize within a stage. For instance, Martins et al. build a message-passing-based implementation of sequence alignment dynamic programs (i.e., Smith–Waterman and Needleman–Wunsch) using wavefront parallelism.¹⁴ In contrast, this paper exploits parallelism across stages, which is orthogonal to wave-front parallelism.

Stivala et al. use an alternate strategy for parallelizing dynamic programming.²⁰ They use a “top-down” approach that solves the dynamic programming problem by recursively solving the subproblems in parallel. To avoid redundant solutions to the same subproblem, they use a lock-free data structure that memoizes the result of the subproblems. This shared data structure makes it difficult to parallelize across multiple machines.

There is also a large body of theoretical work on the parallel complexity of instances of dynamic programming. Some of them^{1,8,22} view dynamic programming instances as finding a shortest path in an appropriate graph and compute all-pairs shortest paths in graph partitions in parallel. Our work builds on these insights and can be viewed as using rank convergence to compute the all-pairs shortest paths efficiently.

Prior works have also made and utilized observations similar to rank convergence. The classic work on Viterbi decoding²⁴ uses the convergence of decoding paths to synchronize decoding and to save memory by truncating paths for the backward phase. Fettweis and Meyr^{6,7} use this observation to parallelize Viterbi decoding by processing overlapping chunks of the input. However, their parallelization can produce an erroneous decoding, albeit under extremely rare conditions.

8. CONCLUSION

This paper introduces a novel method for parallelizing a class of dynamic programming problems called linear-tropical dynamic programming problems, which includes important optimization problems such as Viterbi and longest common subsequence. The algorithm uses algebraic properties of the tropical semiring to break data dependence efficiently.

Our implementations show significant speedups over optimized sequential implementations. In particular, the parallel Viterbi decoding is up to 24× faster (with 64 cores) than a highly optimized commercial baseline.

While we evaluate our approach on a cluster, we expect equally impressive results on a variety of parallel hardware platforms (shared-memory machines, GPUs, and FPGAs).

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. CNS 1111407. The authors thank the Texas Advanced Computing Center for providing computation time on the Stampede cluster. We also greatly thank Guy Steele for his invaluable comments and efforts for editing this paper. We also extend our thanks to Serdar Tasiran and anonymous reviewers for useful feedback on the paper.

References

1. Apostolico, A., Atallah, M. J., Larmore, L. L., McFaddin, S. Efficient parallel algorithms for string editing and related problems. *SIAM J. Comput.* 19, 5 (1990), 968–988.
2. Bellman, R. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.
3. Deorowicz, S. Bit-parallel algorithm for the constrained longest common subsequence problem. *Fund. Inform.* 99, 4 (2010), 409–433.
4. Develin, M., Santos, F., Sturm, B. On the rank of a tropical matrix. *Combin. Comput. Geom.* 52 (2005), 213–242.
5. Farrar, M. Striped Smith–Waterman speeds database searches six times over other SIMD implementations. *Bioinformatics* 23, 2 (2007), 156–161.
6. Fettweis, G., Meyr, H. Feedforward architectures for parallel Viterbi decoding. *J. VLSI Signal Process. Syst.* 3, 1–2 (June 1991), 105–119.
7. Fettweis, G., Meyr, H. High-speed parallel Viterbi decoding: algorithm and VLSI-architecture. *Commun. Mag. IEEE* 29, 5 (1991), 46–55.
8. Galil, Z., Park, K. Parallel algorithms for dynamic programming recurrences with more than $O(1)$ dependency. *J. Parallel Distrib. Comput.* 21, 2 (1994), 213–222.
9. Hillis, W. D., Steele, G. L., Jr. Data parallel algorithms. *Commun. ACM* 29, 12 (Dec. 1986), 1170–1183.
10. Hirschberg, D. S. A linear space algorithm for computing maximal common subsequences. *Commun. ACM* 18, 6 (June 1975), 341–343.
11. Hyiro, H. Bit-parallel LCS-length computation revisited. In *Proceedings of the 15th Australasian Workshop on Combinatorial Algorithms* (2004), 16–27.
12. Ladner, R. E., Fischer, M. J. Parallel prefix computation. *J. ACM* 27, 4 (Oct. 1980), 831–838.
13. Maleki, S., Musuvathi, M., Mytkowicz, T. Parallelizing dynamic programming through rank convergence. In *Proceedings of the 19th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP '14* (New York, NY, USA, 2014), ACM, 219–232.
14. Martins, W. S., Cuvillo, J. B. D., Useche, F. J., Theobald, K. B., Gao, G. A. multithreaded parallel implementation of a dynamic programming algorithm for sequence comparison. In *Pacific Symposium on Biocomputing* (2001), 311–322.
15. Muraoka, Y. Parallelism exposure and exploitation in programs. PhD thesis, University of Illinois at Urbana-Champaign (1971).
16. National Center for Biotechnology Information, <http://www.ncbi.nlm.nih.gov/>, 2013.
17. Needleman, S. B., Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* 48 (1970) 443–453.
18. Püschel, M., Moura, J. M. F., Johnson, J., Padua, D., Veloso, M., Singer, B., Xiong, J., Franchetti, F., Gacic, A., Voronenko, Y., Chen, K., Johnson, R. W., Rizzolo, N. SPIRAL: code generation for DSP transforms. *Proceedings of the IEEE, Special Issue on "Program Generation, Optimization, and Adaptation"* 93 (2005) 232–275.
19. Smith, T., Waterman, M. Identification of common molecular subsequences. *J. Mol. Biol.* 147, 1 (1981), 195–197.
20. Stivala, A., Stuckey, P. J., Garcia de la Banda, M., Hermenegildo, M., Wirth, A. Lock-free parallel dynamic programming. *J. Parallel Distrib. Comput.* 70, 8 (Aug. 2010), 839–848.
21. Texas Advanced Computing Center, <http://www.tacc.utexas.edu/resources/hpc/Stampede/DellPowerEdgeC8220ClusterwithIntelXeonPhiCoproprocessors>.
22. Valiant, L. G., Skyum, S., Berkowitz, S., Rackoff, C. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.* 12, 4 (1983), 641–644.
23. Viterbi, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inform. Theory* 13, 2 (1967), 260–269.
24. Viterbi, A. J., Omura, J. K. *Principles of Digital Communication and Coding*. Communications and information theory. McGraw-Hill, New York, 1979. Autréimpr.: 1985.

Saeed Maleki, Madanlal Musuvathi, and Todd Mytkowicz (saemal, madanm, toddm)@microsoft.com, Microsoft Research, Redmond, WA.

California Institute of Technology The Department of Computing and Mathematical Sciences (CMS) Lecturer

The Department of Computing and Mathematical Sciences (CMS) at California Institute of Technology invites applications for the position of Lecturer in Computing and Mathematical Sciences. This is a (non-tenure-track) career teaching position, with full-time teaching responsibilities. The start date for the position is September 1, 2017 and the initial term of appointment can be up to three years.

The lecturer will teach introductory computer science courses including data structures, algorithms and software engineering, and will work closely with the CMS faculty on instructional matters. The ability to teach intermediate-level undergraduate courses in areas such as software engineering, computing systems or compilers is desired. The lecturer may also assist in other aspects of the undergraduate program, including curriculum development, academic advising, and monitoring research projects. The lecturer must have a track record of excellence in teaching computer science to undergraduates. In addition, the lecturer will have opportunities to participate in research projects in the department. An advanced degree in Computer Science or related field is desired but not required.

Please view the application instructions and apply on-line at <https://applications.caltech.edu/job/cmslect>

The California Institute of Technology is an Equal Opportunity/Affirmative Action Employer. Women, minorities, veterans, and disabled persons are encouraged to apply.

Creighton University Assistant Professor and Clare Boothe Luce Faculty Chair

Creighton University invites applications for a Clare Boothe Luce Faculty Chair in Computer Science. The appointment is tenure-track at the Assistant Professor level, with the 5-year rotating chair established under the terms of the Luce Foundation. We seek an individual with the potential to be an excellent teacher-scholar and an exemplary mentor for undergraduate women interested in STEM careers. The CS program is housed in the innovative, cross-disciplinary Department of Journalism, Media & Computing. See <http://jmc.creighton.edu/jobs> for details.

EO/AA Employer: M/F/Disabled/Vet.

Mississippi State University Professor and Head Department of Computer Science and Engineering

Applications and nominations are being sought for the Professor and Head of the Department of Computer Science and Engineering (www.cse.msstate.edu) at Mississippi State University. The Head is responsible for the overall administration of the department and this is a 12-month tenured position.

The successful Head will provide:

- Vision and leadership for nationally recognized computing education and research programs
- Exceptional academic and administrative skills
- A strong commitment to faculty recruitment and development
- A strong commitment to promoting diversity

Applicants must have a Ph.D. in computer science, software engineering, computer engineering, or a closely related field. The successful candidate must have earned national recognition by a distinguished record of accomplishments in computer science education and research. Demonstrated administrative experience is desired, as is teaching experience at both the undergraduate and graduate levels. The successful candidate must qualify for the rank of professor.

Applicants must apply online by completing the Personal Data Information Form and submitting a cover letter outlining your experience and vision for this position, a curriculum vitae, and the names and contact information of at least three professional references. The online applicant site can be accessed by going to www.hrm.msstate.edu. Screening of candidates will begin November 1, 2016 and will continue until the position is filled. Inquiries and nominations should be directed to Dr. Pedro J. Mago, Department Head of Mechanical Engineering and Search Committee Chair (mago@me.msstate.edu or 662-325-3260).

MSU is an equal opportunity employer, and all qualified applicants will receive consideration for employment without regard to race, color, religion, ethnicity, sex (including pregnancy and gender identity), national origin, disability status, age, sexual orientation, genetic information, protected veteran status, or any other characteristic protected by law. We always welcome nominations and applications from women, members of any minority group, and others who share our passion for building a diverse community that reflects the diversity in our student population.

Montana State University Gianforte School of Computing

Montana State University's Gianforte School of Computing in beautiful Bozeman, Montana invites applications for the following positions: (1) an Assistant Professor of Computer Science (2) an Assistant Teaching Professor of Computer Science or (3) an Instructor of Computer Science.

For complete job announcement and application procedures, please visit cs.montana.edu/opportunities.html.

Equal Opportunity Employer, veterans/Disabled.

San Diego State University Department of Computer Science Chair of Computer Science

Department of Computer Science at SDSU seeks candidates for the Chair position with a PhD in Computer Science or a related discipline, and a sustained record of supported research. The Department is a dynamic and growing unit looking for a visionary Chair to lead its future development.

We strive to build and sustain a welcoming environment for all. SDSU is seeking applicants with commitment to working effectively with individuals from diverse backgrounds and members of underrepresented groups.

For more details and application procedures, please apply via <https://apply.interfolio.com/36248>.

SDSU is a Title IX, equal opportunity employer. A full version of this ad can be found at: <http://cs.sdsu.edu/>

The University of Michigan – Dearborn Department of Computer and Information Science Assistant/Associate Professors

The Department of Computer and Information Science (CIS) at the University of Michigan-Dearborn invites applications for several tenure-track faculty positions in all areas of computer science, with special emphasis on cybersecurity, computer systems, and data science. The expected starting date is September 1st, 2017. Review of applications will begin immediately and continue until suitable candidates are appointed. Rank and salary will be commensurate with qualifications and experience. We offer competitive salaries and start-up packages.

Qualified candidates must have, or expect to have, a Ph.D. in computer science or a closely related discipline by the time of appointment and will be expected to do scholarly and sponsored research, as well as teaching at both the undergraduate and graduate levels. Candidates at the associate professor rank should already have an established funded research program. The CIS Department offers several BS and MS degrees, and participates in several interdisciplinary degree programs, including a Ph.D. program in information systems engineering. A departmental Ph.D. in Computer and Information Science is currently under development. The current research areas in the department include artificial intelligence, complex systems, computational game theory, computer graphics, data management, game design, graphical models, machine learning, multimedia, natural language processing, networking, security, wearable sensing and health informatics, and software engineering. These areas of research are supported by several established labs and many of these areas are currently funded by federal agencies and industries.

The University of Michigan-Dearborn is located in the southeastern Michigan area and offers

excellent opportunities for faculty collaboration with many industries. We are one of three campuses forming the University of Michigan system and are a comprehensive university with over 9,000 students.

The University of Michigan-Dearborn is dedicated to the goal of building a culturally-diverse and pluralistic faculty committed to teaching and working in a multicultural environment, and strongly encourages applications from minorities and women.

A cover letter, curriculum vitae, teaching statement, research statement, and the names and contact information of three references should be sent to:

Dr. William Grosky, Chair
Department of Computer and Information Science
University of Michigan-Dearborn
4901 Evergreen Road, 105 CIS Building
Dearborn, MI 48128-1491
Email: wgrosky@umich.edu
Internet: <http://umdearborn.edu/cecs/CIS/>
Phone: 313.583.6424, Fax: 248.856.2582

The University of Michigan-Dearborn is an equal opportunity/affirmative action employer.

US Air Force Academy Department of Computer Science

The Department of Computer Science at the US Air Force Academy seeks to fill up to two full-time faculty positions at the Assistant Professor

level. The department is particularly interested in candidates with backgrounds in artificial intelligence, computer and network security, or unmanned aerial systems, but all candidates with a passion for undergraduate computer science teaching are encouraged to apply.

The Academy is a national service institution, charged with producing lieutenants for the US Air Force. Faculty members are expected to exemplify the highest ideals of professionalism and character. USAFA is located in Colorado Springs, an area known for its exceptional natural beauty and quality of life. The United States Air Force Academy values the benefits of diversity among the faculty to include a variety of educational backgrounds and professional and life experiences.

For information on how to apply, go to usa-jobs.gov and search with the keyword 447356400.

Wesleyan University Department of Mathematics and Computer Science

The Department of Mathematics and Computer Science at Wesleyan University invites applications for a tenure track assistant professorship in Computer Science (three courses per year) to begin in Fall 2017. We encourage applicants in all areas to apply.

We will begin reviewing applications on Dec. 1, 2016.

Applications must be submitted online at <https://academicjobsonline.org/ajo/jobs/7547>, where the full job description may be found.

York University Department of Electrical Engineering and Computer Science

The Department of Electrical Engineering and Computer Science, York University, invites applications for a tenure-track appointment at the rank of Assistant Professor in the area of Computer Science, to commence July 1, 2017, subject to budgetary approval. We are seeking an outstanding candidate with a particular research focus and ability to teach in Robotics or Machine Learning, although exceptional applicants in other areas of computer science will be considered. The successful candidate will have a PhD in Computer Science, or a closely related field, and a research record commensurate with rank.

For full position details, see <http://www.yorku.ca/acadjobs>. Applicants should complete the on-line process at <http://lassonde.yorku.ca/new-faculty/>. A complete application includes a cover letter, a detailed curriculum vitae, statement of contribution to research, teaching and curriculum development, three sample research publications and three reference letters. Complete applications must be received by November 30, 2016.

York University is an Affirmative Action (AA) employer. The AA Program can be found at <http://www.yorku.ca/acadjobs> or a copy can be obtained by calling the AA office at 416-736-5713.

All qualified candidates are encouraged to apply; however, Canadian citizens and permanent residents will be given priority.



Multiple Tenure-Track or Tenured Faculty Positions in Computer Science

The Department of Computer Science at the National University of Singapore (NUS) invites applications for several tenure-track or tenured faculty positions. We have positions dedicated to cyber security, Internet of Things, robotics and big data analytics as well as positions open to all areas of computer science. While our main focus is on the assistant professor level, we also welcome exceptional candidates at the associate and full professor levels. For applications at the assistant professor level, candidates should demonstrate excellent research potential and a strong commitment to teaching. Candidates at more senior levels should have an established record of outstanding and recognized research achievements. Truly outstanding assistant professor level applicants will also be considered for the prestigious Sung Kah Kay Assistant Professorship.

The Department of Computer Science at NUS is highly ranked internationally. It enjoys ample research funding, moderate teaching load, excellent facilities, and extensive international collaborations. The department covers all major research areas in computer science and boasts a thriving PhD program that attracts the brightest students from the region and beyond. More information is available at <http://www.comp.nus.edu.sg/>.

NUS offers highly competitive salaries and is situated in Singapore, an English-speaking cosmopolitan city and a melting pot of many cultures, both the east and the west. Singapore offers high-quality education and healthcare at all levels, high level of personal freedom and security, as well as very low tax rates.

Interested candidates are invited to send, via electronic submission, the following materials to our electronic application website: <https://faces.comp.nus.edu.sg> with the following combined into a single PDF document:

- A cover letter that clearly indicates main research interests
- Curriculum Vitae
- A teaching statement
- A research statement

Please also arrange for **at least 3 references** to be sent directly to csrec@comp.nus.edu.sg or provide the contact information at the submission website. Applicants are assumed to have obtained their references' consent to be contacted for this matter.

Application review will commence on October 1, 2016 and continue until the positions are filled. To ensure maximal consideration, please submit your application by December 15, 2016. If there are further queries, please feel free to send the Search Committee Chair Weng-Fai Wong an email at the above email address.



ADVERTISING IN CAREER OPPORTUNITIES

How to Submit a Classified Line Ad: Send an e-mail to acmm mediasales@acm.org. Please include text, and indicate the issue/or issues where the ad will appear, and a contact name and number.

Estimates: An insertion order will then be e-mailed back to you. The ad will be typeset according to CACM guidelines. NO PROOFS can be sent. Classified line ads are NOT commissionable.

Rates: \$325.00 for six lines of text, 40 characters per line. \$32.50 for each additional line after the first six. The MINIMUM is six lines.

Deadlines: 20th of the month/2 months prior to issue date. For latest deadline info, please contact:

acmm mediasales@acm.org

Career Opportunities Online: Classified and recruitment display ads receive a free duplicate listing on our website at <http://jobs.acm.org>

Ads are listed for a period of 30 days.

For More Information Contact:

**ACM Media Sales
at 212-626-0686 or
acmm mediasales@acm.org**

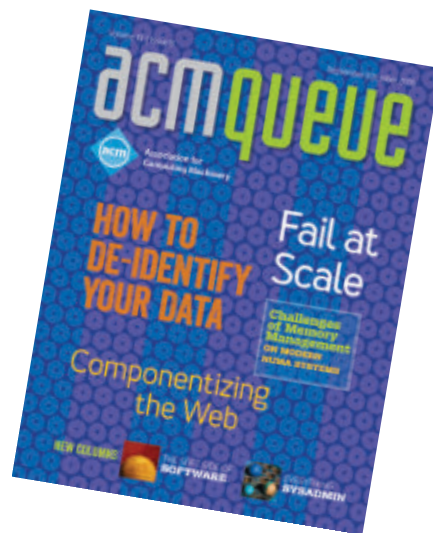
acmqueue

Check out the new acmqueue app

FREE TO ACM MEMBERS

acmqueue is ACM's magazine by and for practitioners, bridging the gap between academics and practitioners of computer science. After more than a decade of providing unique perspectives on how current and emerging technologies are being applied in the field, the new acmqueue has evolved into an interactive, socially networked, electronic magazine.

Broaden your knowledge with technical articles focusing on today's problems affecting CS in practice, video interviews, roundtables, case studies, and lively columns.



Keep up with this fast-paced world on the go. Download the mobile app.



Association for
Computing Machinery

Desktop digital edition also available at queue.acm.org.
Bimonthly issues free to ACM Professional Members.
Annual subscription \$19.99 for nonmembers.



DOI:10.1145/2987349

Dennis Shasha

Upstart Puzzles

Find Me Quickly

IN THIS COOPERATIVE game, two players want to meet each other in a graph as quickly as possible. Meeting each other means both players are at the same node at the same time or traverse an edge in opposite directions in some minute. Each player moves or stays put each minute. A move takes one player from one node across an edge to a neighboring node in the undirected graph.

Warm-up: Suppose the two players are in a graph consisting of a cycle of n nodes (see Figure 1). The nodes are numbered, and each player knows both the topology and the number of the node where he or she is placed. If both players move, say, clockwise, they may never meet. If player A does not move (the “stay-put” strategy) and player B moves in one direction, player B will find player A in $n-1$ minutes in the worst case. Alternatively, if both agree to move as quickly as possible to some node, say, node 4, and stay there, then the latter of the two will arrive at node 4 in $n/2$ minutes at most. Is there any other strategy that has a worst-case time complexity of

$n/2$ minutes but also a better average-case time complexity than the go-to-a-common-node strategy?

Solution to warm up. Player A can always move clockwise (given a map of the graph for which clockwise makes sense), and player B can always move counterclockwise. They will meet each other in at most $n/2$ minutes in the worst case, with an expected value less than the go-to-a-common-node strategy.

A graph consisting of a single cycle is, of course, a special case. For an arbitrary graph of size n , where each player knows his or her own position and the topology of the graph and where every node has a unique identifier, is there a solution that will take no more than $n/2$ minutes in the worst case?

Solution. Go to the centroid of the graph, or the node to which the maximum distance from any other node is minimized. If there are several such nodes, go to the one with the lexicographically minimum node id. Note that such a centroid cannot have a distance greater than $n/2$ to any other node.

We are just getting started. Now consider situations in which each player knows the topology but not where he or she is placed and the nodes have no identifiers.

Start by considering a graph consisting of a single path. If player A stays put and player B moves in one direction and bounces back from the end if player B does not find A, the worst-case time could be $2n-3$ minutes. Is there a strategy that takes no more than n minutes in the worst case?

Solution. Yes, each player goes in some direction, and when that player hits an end, he or she bounces back. In the worst case, this strategy takes $n-1$ minutes, with an expected value of approximately $3n/4$.

Now for the upstarts:

Upstart 1. Better than staying put.

When both players do not know where they are placed, nodes are unlabeled and the graph has at least one cycle, find a strategy that is better in the worst case than the one-player-stays-put strategy.

Upstart 2. Also better than staying put.

In the same setting as Upstart 1, say we allow both players to leave notes (see Figure 2) on nodes they have visited. Is there an approach that takes $n/2$ minutes for the two players to meet up in the worst case? If not, is there an approach that takes $3n/2$ minutes in the worst case? Please specify whichever approach you come up with. 

All are invited to submit their solutions to upstartpuzzles@cacm.acm.org; solutions to upstarts and discussion will be posted at <http://cs.nyu.edu/cs/faculty/shasha/papers/cacmpuzzles.html>

Dennis Shasha (dennisshasha@yahoo.com) is a professor of computer science in the Computer Science Department of the Courant Institute at New York University, New York, as well as the chronicler of his good friend the omniheurist Dr. Ecco.

Copyright held by the author.

Figure 1. The goal is for two players who know the topology of a graph to find one another as quickly as possible.

For a graph consisting of a single cycle where each player knows his or her position but not the position of the other player, which strategy is best?

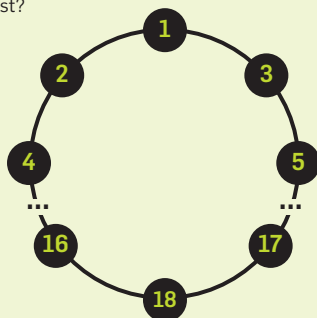
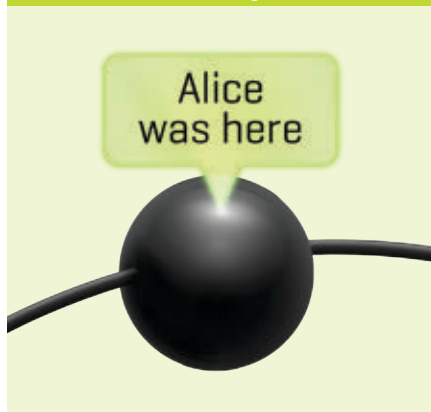


Figure 2. Suppose each player (“Alice” in this case) could leave a small number of notes identifying herself along with any other information she might want to include.



Computing Reviews

Connect with our Community of Reviewers

“I like CR because it covers the full spectrum of computing research, beyond the comfort zone of one’s specialty. I always look forward to the next Editor’s Pick to get a new perspective.”

- Alessandro Berni



Association for
Computing Machinery

ThinkLoud

www.computingreviews.com

<Programming> 2017

The Art, Science, and Engineering of
Programming Conference and Journal

Call for Papers

We started a conference and journal focused on everything to do with programming, including the experience of programming. We call the conference <Programming> for short. Paper submissions and publication are handled by the journal. Accepted papers must be presented at the <Programming> conference.

The Art, Science, and Engineering of Programming accepts scholarly papers including essays that advance knowledge of programming. Almost anything about programming is in scope, but in each case there should be a clear relevance to the act and experience of programming. There are several submission periods per year.

More Info & Program on
2017.programming-conference.org

Paper Submission Deadline
December 1, 2016 (issue #2)

Vrije Universiteit Brussel
3-6 April, 2017 – Brussels, Belgium



Vrije
Universiteit
Brussel

General Chair
Theo D'Hondt, VUB

Local Organizing Chair
Wolfgang De Meuter, VUB

Program Chair
Crista V. Lopes, UC Irvine

Workshops Chair
Jörg Kienzle, McGill U.

Demos Chair
Hidehiko Masuhara, Tokyo Tech

Programming Contests Chair
Ralf Lämmel, U. Koblenz-Landau

Web Technology Chair
Tobias Pape, HPI

Program Committee
Andrew Black, Portland State U.
Shigeru Chiba, U. Tokyo
Yvonne Coady, U. Victoria
Robby Findler, Northwestern U.
Lidia Fuentes, U. Málaga
Richard P. Gabriel, IBM
Elisa Gonzalez Boix, VUB

Jeff Gray, U. Alabama
Robert Hirschfeld, HPI
Roberto Ierusalimsky, PUC-Rio
Jörg Kienzle, McGill U.
Hidehiko Masuhara, Tokyo Tech
Sasa Misailovic, ETH Zurich
Guido Salvaneschi, TU Darmstadt
Mario Südholt, Mines Nantes
Jurgen Vinju, CWI / TUE
Tijs van der Storm, CWI

