

SEX

AS AN ALGORITHM

**THE THEORY OF EVOLUTION
UNDER THE LENS OF COMPUTATION**

31st IEEE INTERNATIONAL Parallel and Distributed Processing SYMPOSIUM



May 29-June 2, 2017
Buena Vista Palace Hotel
Orlando, Florida USA

www.ipdps.org

Orlando is home to a rich offering of indoor and outdoor attractions. Located a mile from Walt Disney World® and 4 miles from Epcot, the Buena Vista Palace Hotel is a 5-minute walk from Downtown Disney with a complimentary shuttle to all Disney Theme Parks and Water Parks. The sprawling Lake Buena Vista resort offers a full menu of amenities and family friendly activities as well as ideal meeting space for IPDPS 2017.

ANNOUNCING 24 WORKSHOPS PLANNED FOR IPDPS 2017 IN ORLANDO

IPDPS Workshops are the "bookends" to the three-day technical program of contributed papers, keynote speakers, roundtable workshops, a PhD student forum, and industry participation. They provide the IPDPS community an opportunity to explore special topics and present work that is more preliminary or cutting-edge than the more mature research presented in the main symposium. Each workshop has its own website and submission requirements, and the submission deadline for most workshops is after the main conference author notification date. See the IPDPS Workshops page for links to Call for Papers for each workshop and due dates.

IPDPS WORKSHOPS MONDAY 29 MAY 2017 (Check final schedule)

HCW	Heterogeneity in Computing Workshop
RAW	Reconfigurable Architectures Workshop
HiComb	High Performance Computational Biology
EduPar	NSF/TCPP W. on Parallel and Distributed Computing Education
ParLearning	Parallel and Distributed Computing for Machine Learning and Big Data Analytics
PDCO	Parallel / Distributed Computing and Optimization
GABB	Graph Algorithms Building Blocks
AsHES	Accelerators and Hybrid Exascale Systems
HIPS	High Level Programming Models and Supporting Environments
APDCM	Advances in Parallel and Distributed Computational Models
HPPAC	High-Performance, Power-Aware Computing
HPBDC	High-Performance Big Data Computing

IPDPS WORKSHOPS FRIDAY 2 JUNE 2017 (Check final schedule)

CHIUV	Chapel Implementers and Users Workshop
LSP	Large-Scale Parallel Processing: Practices and Experiences
PDSEC	Parallel and Distributed Scientific and Engineering Computing
JSSPP	Job Scheduling Strategies for Parallel Processors
DPDNS	Dependable Parallel, Distributed and Network-centric Systems
IPDRM	Emerging Parallel and Distributed Runtime Systems and Middleware
iWAPT	International Workshop on Automatic Performance Tunings
ParSocial	Parallel and Distributed Processing for Computational Social System
BigDataEco	Big Data Regional Innovation Hubs and Spokes: Accelerating the Big Data Innovation Ecosystem
GrML	Graph Algorithms and Machine Learning
EMBRACE	Evolvable Methods for Benchmarking Realism and Community Engagement
REPPAR	Reproducibility in Parallel Computing

GENERAL CHAIR

Michela Taufer (University of Delaware, USA)

PROGRAM CHAIR

Marc Snir (University of Illinois at Urbana Champaign, USA)

WORKSHOPS CHAIR

Bora Uçar (CNRS and ENS Lyon, France)

WORKSHOPS VICE-CHAIR

Erik Saule (University of North Carolina Charlotte, USA)

STUDENT PARTICIPATION CHAIR

Trilce Estrada (University of New Mexico, USA)

ROUNDTABLE WORKSHOPS

These condensed workshops, organized and animated by a few people, will be held on Tuesday and Thursday in a "roundtable" setting designed to promote one-on-one interaction. They will focus on an emerging area of interest to IPDPS attendees, especially topics that complement and "round out" the areas covered by the regular workshops.

PHD FORUM & STUDENT MENTORING

This event will include traditional poster presentations by PhD students enhanced by a program of mentoring and coaching in scientific writing and presentation skills and a special opportunity for students to hear from and interact with senior researchers attending the conference.

INDUSTRY PARTICIPATION

IPDPS extends a special invitation for companies to become an IPDPS 2017 Industry Partner and join us in Orlando to share in the benefits of associating with an international community of top researchers and practitioners in fields related to parallel processing and distributed computing. Visit the IPDPS website to see ways to participate.

IMPORTANT DATES

Conference Author Notification	January 8, 2017
Workshop Call for Papers Deadlines	Most Fall After January 8, 2017

SPONSORED BY:



Technical Committee on Parallel Processing

IN COOPERATION WITH:



ACM SIGARCH & SIGHPC

IEEE Computer Society Technical Committee on Computer Architecture

IEEE Computer Society Technical Committee on Distributed Processing

**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 **Editor's Letter**
Globalization, Computing, and Their Political Impact
By Moshe Y. Vardi
-
- 7 **Cerf's Up**
Heidelberg Anew
By Vinton G. Cerf
-
- 8 **Letters to the Editor**
Learn to Live with Academic Rankings
-
- 10 **BLOG@CACM**
Introducing CS to Newcomers, and JES As a Teaching Tool
Valerie Barr gets high schoolers thinking about CS, while Mark Guzdial mulls the benefits of Jython Environment for Students.
-
- 23 **Calendar**
-
- 123 **Careers**

Last Byte

- 136 **Future Tense**
The Candidate
Seeking the programmer vote, an AI delivering a slogan like "Make Coding Great Again" could easily be seen as a threat.
By Brian Clegg

News

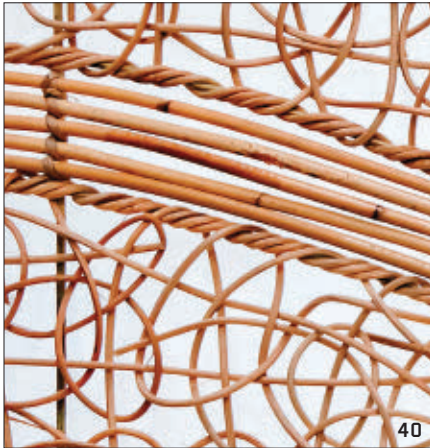


- 12 **Learning Securely**
Because it is easy to fool, machine learning must be taught how to handle adversarial inputs.
By Erica Klarreich
-
- 15 **Blockchain Beyond Bitcoin**
Blockchain technology has the potential to revolutionize applications and redefine the digital economy.
By Sarah Underwood
-
- 18 **Farm Automation Gets Smarter**
As fewer people work the land, robots pick up the slack.
By Tom Geller

Viewpoints

- 20 **Privacy and Security**
Cyber Defense Triad for Where Security Matters
Dramatically more trustworthy cyber security is a choice.
By Roger R. Schell
-
- 24 **Legally Speaking**
Fair Use Prevails in Oracle v. Google
Two software giants continue with legal sparring after an initial judicial decision.
By Pamela Samuelson
-
- 27 **Economic and Business Dimensions**
Visualization to Understand Ecosystems
Mapping relationships between stakeholders in an ecosystem to increase understanding and make better-informed strategic decisions.
By Bala R. Iyer and Rahul C. Basole
-
- 31 **Education**
Growing Computer Science Education Into a STEM Education Discipline
Seeking to make computing education as available as mathematics or science education.
By Mark Guzdial and Briana Morrison
-
- 34 **Viewpoint**
Time to Reinspect the Foundations?
Questioning if computer science is outgrowing its traditional foundations.
By Jack Copeland, Eli Dresner, Diane Proudfoot, and Oron Shagrir
-
- 37 **Viewpoint**
Technology and Academic Lives
Considering the need to create new modes of interaction and approaches to assessment given a rapidly evolving academic realm.
By Jonathan Grudin

Practice



- 40 **The Power of Babble**
Expect to be constantly
and pleasantly befuddled.
By Pat Helland

- 44 **Scaling Synchronization
in Multicore Programs**
Advanced synchronization
methods can boost the performance
of multicore software.
By Adam Morrison

- 52 **Research for Practice: Distributed
Consensus and Implications of NVM
on Database Management Systems**
Expert-curated guides to the best
of CS research for practitioners.
*By Peter Bailis, Camille Fournier,
Joy Arulraj, and Andrew Pavlo*



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

Contributed Articles

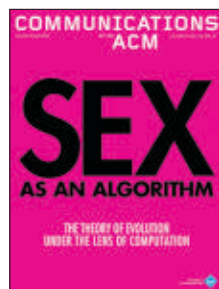
- 56 **Apache Spark: A Unified Engine
for Big Data Processing**
This open source computing
framework unifies streaming, batch,
and interactive big data workloads to
unlock new applications.
*By Matei Zaharia, Reynold S. Xin,
Patrick Wendell, Tathagata Das,
Michael Armbrust, Ankur Dave,
Xiangrui Meng, Josh Rosen,
Shivaram Venkataraman,
Michael J. Franklin, Ali Ghodsi,
Joseph Gonzalez, Scott Shenker,
and Ion Stoica*



Watch the authors discuss
their work in this exclusive
Communications video.
[http://cacm.acm.org/videos/
spark](http://cacm.acm.org/videos/spark)

- 66 **Pushing on String: The 'Don't Care'
Region of Password Strength**
Enterprises that impose stringent
password-composition policies
appear to suffer the same fate
as those that do not.
*By Dinei Florêncio, Cormac Herley,
and Paul C. van Oorschot*

- 75 **A Theory on Power in Networks**
Actors linked to central others
in networks are generally central,
even as actors linked to powerful
others are powerless.
*By Enrico Bozzo and
Massimo Franceschet*

**About the Cover:**

When it comes to
unlocking the secrets
of evolution, much
can be gained from
exploring these stories
from a computer
science perspective.
Adi Livnat and Christos
Papadimitriou detail how
CS can be used to trace
the origins and role of
sexual reproduction.

Review Articles

- 84 **Sex as an Algorithm: The Theory
of Evolution Under the Lens
of Computation**
Looking at the mysteries of evolution
from a computer science point of
view yields some unexpected insights.
*By Adi Livnat and
Christos Papadimitriou*



Watch the authors discuss
their work in this exclusive
Communications video.
[http://cacm.acm.org/videos/
sex-as-an-algorithm](http://cacm.acm.org/videos/sex-as-an-algorithm)

- 94 **Recommender Systems—
Beyond Matrix Completion**
The future success of these
systems depends on more than
a Netflix challenge.
*By Dietmar Jannach, Paul Resnick,
Alexander Tuzhilin, and Markus Zanker*

Research Highlights

- 104 **Technical Perspective**
If I Could Only Design One Circuit ...
By Kurt Keutzer
- 105 **DianNao Family: Energy-Efficient
Hardware Accelerators for
Machine Learning**
*By Yunji Chen, Tianshi Chen, Zhiwei Xu,
Ninghui Sun, and Olivier Temam*
- 113 **Technical Perspective**
**FPGA Compute Acceleration
Is First About Energy Efficiency**
By James C. Hoe

- 114 **A Reconfigurable Fabric for
Accelerating Large-Scale
Datacenter Services**
*By A. Putnam, A.M. Caulfield, E.S. Chung,
D. Chiou, K. Constantinides, J. Demme,
H. Esmaeilzadeh, J. Fowers, G.P. Gopal,
J. Gray, M. Haselman, S. Hauck, S. Heil,
A. Hormati, J.-Y. Kim, S. Lanka, J. Larus,
E. Peterson, S. Pope, A. Smith, J. Thong,
P. Yi Xiao, and D. Burger*



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

Vicki L. Hanson

Vice-President

Cherri M. Pancake

Secretary/Treasurer

Elizabeth Churchill

Past President

Alexander L. Wolf

Chair, SGB Board

Patrick Madden

Co-Chairs, Publications Board

Jack Davidson and Joseph Konstan

Members-at-Large

Gabriele Anderst-Kotis; Susan Dumais;

Elizabeth D. Mynatt; Pamela Samuelson;

Eugene H. Spafford

SGB Council Representatives

Paul Beame; Jenna Neeffe Matthews;

Barbara Boucher Owens

BOARD CHAIRS

Education Board

Mehran Sahami and Jane Chu Prey

Practitioners Board

George Neville-Niel

REGIONAL COUNCIL CHAIRS

ACM Europe Council

Dame Professor Wendy Hall

ACM India Council

Srinivas Padmanabhuni

ACM China Council

Jianguang 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 Guerrin; Carol Hutchins;

Yannis Ioannidis; Catherine McGeoch;

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

acmhelp@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

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-Niel; Theo Schlossnagle;

Jim Waldo

The Practice section of the CACM

Editorial Board also serves as

the Editorial Board of *COMMUNIQUE*.

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;

Mari-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;

Alexei Efros; Alon Halevy; Norm Jouppi;

Andrew B. Kahng; Sven Koenig; Xavier Leroy;

Steve Marschner; Kobbi Nissim; Guy Steele,

Jr.; David Wagner; Margaret H. Wright;

Nicholai Zeldovich; 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 acmhelp@acm.org.

COMMUNICATIONS OF THE ACM

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

POSTMASTER

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

Printed in the U.S.A.



Association for
Computing Machinery





DOI:10.1145/3003732

Moshe Y. Vardi

Globalization, Computing, and Their Political Impact

PERCHED AS WE are on the crest of the current tech and computing-enrollment boom, it is hard to remember the dark days of the early 2000s. The NASDAQ Index peaked on March 10, 2000, declining almost 80% over the next two years. The stock-market crash in the U.S. caused the loss of \$5 trillion in the market valuations from 2000 to 2002. Computing enrollments in North America went into a steep dive. At the same time, the Internet enabled the globalization of software production, giving rise to the phenomenon of offshore outsourcing. There were daily stories in the media describing major shifts in employment that were occurring largely as a result of software offshoring. Combined with the dot-com bust, these reports raised concerns about the future of information technology (IT) as a viable field of study and work in developed countries.

In response to these concerns, ACM Council commissioned a Task Force in 2004 to “look at the facts behind the rapid globalization of IT and the migration of jobs resulting from outsourcing and offshoring.” The Task Force, co-chaired by Frank Mayadas and myself, with the assistance of William Aspray as Editor, issued its report *Globalization and Offshoring of Software* (<http://www.acm.org/globalizationreport>) in 2006. The report concluded that there is no real reason to believe that IT jobs are migrating away from developed countries. The passing decade has vindicated that conclusion.

But while the report conceded that “trade gains may be distributed differentially,” meaning some individuals gain and some lose, some localities gain and some lose; it was focused narrowly on the IT industry. Had we looked

at the broader impact of globalization on the economy, we might have reached somewhat less sanguine conclusions. Globalization exerted tremendous competitive pressure on manufacturing in developed countries. It is instructive to examine the response to this competitive pressure, taking U.S. manufacturing as an example. To survive in the intensely competitive global economy, U.S. manufacturing had to increase its productivity dramatically, substituting technology for labor. U.S. manufacturing productivity roughly doubled between 1995 and 2015. As a result, while U.S. manufacturing output today is essentially at an all-time high, employment peaked around 1980, and has been declining precipitously since 1995. Neoclassical economists argue that when technology destroys jobs “people find other jobs, albeit possibly after a long period of painful adjustment.” They are definitely right about the painful adjustment! The impact of globalization and automation over the past 20 years on working- and middle-class Americans has been quite harsh.

This impact has been succinctly captured recently by economist Branko Milanovic’s “Elephant Curve” (see <http://prospect.org/article/worlds-inequality>), which shows how people around the world, ranked by their income in 1998, saw their incomes increase by 2008. While the incomes of the very poor were stagnant, rising incomes in emerging economies lifted hundreds of millions of people out of poverty. People at the top of the income scale also benefited from globalization and automation. But the income of working- and middle-class people in the developed world stagnated over that period. In the U.S., for example, income of production workers today, adjusted

for inflation, is at the level it was in 1968! Ironically, automation is now reaching developing-world economies. Manufacturing employment in China peaked around 1995, where rising wages are driving automation, and a recent report from the International Labor Organization found that more than two-thirds of Southeast Asia’s 9.2 million textile and footwear jobs are threatened by automation.

Until very recently, the global professional class, which includes most computing professionals, was somewhat oblivious to the plight of working- and middle-class people in developed countries. In fact, some have argued that this class lives in “an economic and cultural bubble.” Political developments of the past few months have made it clear that the issue of shared prosperity cannot be ignored. It is now evident that the Brexit vote in the U.K., as well as the temporary rise of Bernie Sanders and the rise of Donald Trump in the U.S., were driven to a large extent by economic grievances. However the outcome of this month’s U.S. presidential election, globalization and automation will remain policy issues of the utmost priority and will resist simplistic solutions.

Globalization and automation provide huge benefits to society, but their adverse effects cannot and should not be ignored. Technology is not destiny and public policy has a key role to play. As actors in and beneficiaries of this societal transformation, we have, I believe, a social responsibility that goes beyond our technical roles.

Follow me on Facebook, Google+, and Twitter.

Moshe Y. Vardi, EDITOR-IN-CHIEF

Copyright held by author.

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:10.1145/3005354

Vinton G. Cerf

Heidelberg Anew

I have just returned from the fourth annual Heidelberg Laureate Forum and I want to emphasize how very important it has been for ACM Turing laureates to participate in

the program. Each year 200 math and computer science undergraduates participate in the program, approximately 100 each. Speeches by laureates are mixed with undergraduate workshops and plenary open sessions. There is ample opportunity for interaction among students and laureates and between students.

This year, Brian Schmidt gave the Lindau lecture (from the annual Nobel Prize winners meeting). Schmidt discovered that the universe is not only expanding, the expansion is accelerating. It would be difficult to imagine a more profound discovery. In the very long term, it appears the universe will expand to the point that only a certain amount of local gravity will hold a galaxy or small group of galaxies together. The rest will accelerate away and the universe will end in a cold whimper.

Fortunately, there was nothing but stimulating conversation at this year's Heidelberg Forum. More than ever we are seeing how mathematics and computer science are interacting, especially with the arrival of neural networks and quantum computers that have capabilities quite different from the conventional von Neumann designs that have dominated computing for over seven decades. Fundamental questions about what is computable, illuminated by Gödel, are getting attention in the light of these new computing engines.

Leslie Lamport delivered another extraordinary lecture reinforcing the value of thinking mathematically while considering the process of pro-

gramming. The value of abstraction to aid in reasoning about expected program function resonated very strongly with me and, I think, with others in attendance.

As always, the mathematics and computer science students were full of energy, ideas, and eagerness to interact with each other and with the laureates present. The organizers worked hard to maximize student opportunities to meet with laureates including a number of workshops where some in-depth discussion could be supported. Some of the laureates voiced a strong recommendation that every effort should be made to allow rich interaction between students of the two disciplines.

Looking at the available laureate attendee lists, I can't help but imagine that future Heidelberg events would benefit from a cohort of additional younger laureates so I look forward to the possibility

that other ACM awardees might be invited to attend the annual event.

Looking back on the Lindau event that I attended in late June at which Nobel Prize winners mingle with students, I was struck by the increasingly important role of computing in discovery science. Simulations of physical phenomena are revealing new insights into the nature of our universe. One of the dramatic examples I have seen shows an evolving universe from the big bang that takes into account dark matter and dark energy and produces a simulated universe with many of the large scale structures we actually see in the observable universe. We see huge reticular structures emerging that are largely the product of masses of dark matter that organize ordinary matter into a lacework of stars and gas. That these predictions can be tested through observation reinforces the importance of computing in our exploration of the natural world.

We are reaching an exciting period in scientific discovery in which computation is as important as laboratory experiment and observation. We can invent our own universes and test them for compatibility with the real one we can measure. Indeed, we may find that our predictions could draw our attention to phenomena we might never have looked for, were it not for the revelation of computation. 

We are reaching an exciting period in scientific discovery in which computation is as important as laboratory experiment and observation.

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

Copyright held by author.

Learn to Live with Academic Rankings

NO ONE LIKES being reduced to a number. For example, there is much more to my financial picture than my credit score alone. There is even scholarly work on weaknesses in the system to compute this score. Everyone may agree the number is far from perfect, yet it is used to make decisions that matter to me, as Moshe Y. Vardi discussed in his Editor's Letter "Academic Rankings Considered Harmful!" (Sept. 2016). So I care what my credit score is. Many of us may even have made financial decisions taking into account their potential impact on credit score.

As an academic, I also produce such numbers. I assign grades to my students. I strive to have the assigned grade accurately reflect a student's grasp of the material in my course. But I know this is imperfect. At best, the grade reflects the student's knowledge today. When a prospective employer looks at it two years later, it is possible an A student had crammed for the exam and has since completely forgotten the material, while a B student deepened his or her understanding substantially through a subsequent internship. The employer must learn to get past the grade to develop a richer understanding of the student's strengths and weaknesses.

As an academic, I am also a consumer of these numbers. Most universities, including mine, look at standardized test scores. No one suggests they predict success perfectly. But there is at least some correlation—enough that they are used, often as an initial filter. Surely there are students who could have done very well if admitted but were not considered seriously because they did not make the initial cutoff in test scores. A small handful of U.S. colleges and universities have recently stopped considering standardized test scores for undergraduate admission. I admire their courage. Most others have not followed suit because it takes a tremendous amount of work to get behind the numbers. Even if better decisions might result, the process simply requires too much effort.

As an academic, I appreciate the rich

diversity of attributes that characterize my department, as well as peer departments at other universities. I know how unreasonable it is to reduce it all to a single number. But I also know there are prospective students, as well as their parents and others, who find a number useful. I encourage them to consider an array of factors when I am trying to recruit them to choose Michigan. But I cannot reasonably ask them not to look at the number. So it behooves me to do what I can to make it as good as it can be, and to work toward a system that produces numbers that are as fair as they can be. I agree it is not possible to come anywhere close to perfection, but the less bad we can make the numbers, the better off we all will be.

H.V. Jagadish, Ann Arbor, MI

Author Responds:

My Editor's Letter did not question the need for quantitative evaluation of academic programs. I presume, however, that Dr. Jagadish assigns grades to his students rather than merely ranking them. These students then graduate with a transcript, which reports all their grades, rather than just their class rank. He argues that we should learn to live with numbers (I agree) but does not address any of the weaknesses of academic rankings.

Moshe Y. Vardi, Editor-in-Chief

More Negative Consequences of Academic Rankings

I could not agree more with Moshe Y. Vardi's Editor's Letter (Sept. 2016). The ranking systems—whether U.S.-focused (such as *U.S. News and World Report*) or global (such as *Times Higher Education*, *World University Reputation Ranking*, *QS University Ranking*, and *Academic Ranking of World Universities*, compiled by Shanghai Jiaotong University in Shanghai, China)—have all acquired lives of their own in recent years. These rankings have attracted the attention of governments and funding bodies and are widely reported in the media. Many universities worldwide have reacted by

establishing staff units to provide the diverse data requested by the ranking agencies and boosting their communications and public relations activities. There is also evidence that these league tables are beginning to (adversely) influence resource-allocation and hiring decisions despite their glaring inadequacies and limitations.

I have been asked to serve on the panels of two of the ranking systems but have had to abandon my attempts to complete the questionnaires because I just did not have sufficient information to provide honest responses to the kinds of difficult, comparative questions about such a large number of universities. The agencies seldom report how many "experts" they actually surveyed or their survey-response rates. As regards the relatively "objective" ARWU ranking, it uses measures like number of alumni and staff winning Nobel Prizes and Fields Medals, number of highly cited researchers selected by Thomson Reuters, number of articles published in journals of *Nature* and *Science*, number of articles indexed in *Science* and *Social Science Citation Index*, and "per capita performance" of a university. It is not at all clear to what extent the six narrowly focused indicators can capture the overall performance of modern universities, which tend to be large, complex, loosely coupled organizations. As well, the use of measures like number of highly cited researchers named by Thomson Reuters/ISI can exacerbate some of the known citation malpractices (such as excessive self-citations, citation rings, and journal-citation stacking). As Vardi noted, the critical role of commercial entities in the rankings—notably *Times*, *QS*, *USNWR*, and Thomson Reuters—is also a concern.

Joseph G. Davis, Sydney, Australia

Acknowledge Crowdworkers in Crowdwork Research

Crowdwork promises to help integrate human and computational processes while also providing a source of paid work for those who might otherwise be excluded from the global economy.

Daniel W. Barowy et al.'s Research Highlight "AutoMan: A Platform for Integrating Human-Based and Digital Computation" (June 2016) explored a programming language called AutoMan designed to integrate human workers recruited through crowdwork markets like Amazon Mechanical Turk alongside conventional computing resources. The language breaks new ground in how to automate the complicated work of scheduling, pricing, and managing crowdwork.

While the attempt to automate this managerial responsibility is clearly of value, we were dismayed by the authors' lack of concern for those who carry out the actual work of crowdwork. Humans and computers are not interchangeable. Minimizing wages is quite different from minimizing execution time. For example, the AutoMan language is designed to minimize crowdwork requesters' costs by iteratively running rounds of recruitment, with tasks offered at increasing wages. However, such optimization is quite different from the perspective of the workers compared to the requesters. The process is clearly not optimized for economic fairness. Systems that minimize payments could exert negative economic force on crowdworker wages, failing to account for the complexities of, say, Mechanical Turk as a global labor market.

Recent research published in the proceedings of Computer-Human Interaction and Computer-Supported Cooperative Work conferences by Lily Irani, David Martin, Jacki O'Neill, Mary L. Gray, Aniket Kittur, and others shows how crowdworkers are not interchangeable cogs in a machine but real humans, many dependent on crowdwork to make ends meet. Designing for workers as active, intelligent partners in the functioning of crowdwork systems has great potential. Two examples where researchers have collaborated with crowdworkers are the Turkopticon system, as introduced by Irani and Silberman,¹ which allows crowdworkers to review crowdwork requesters, and Dynamo, as presented by Salehi et al.,² which supports discussion and collective action among crowdworkers. Both projects demonstrate how crowdworkers can be treated as active partners in improving the various crowdwork marketplaces.

We hope future coverage of crowdwork in *Communications* will include research incorporating the perspective of workers in the design of such systems. This will help counteract the risk of creating programming languages that could actively, even if unintentionally, accentuate inequality and poverty. At a time when technology increasingly influences political debate, social responsibility is more than ever an invaluable aspect of computer science.

References

1. Irani, L.C. and Silberman, M.S. Turkopticon: Interrupting worker invisibility in Amazon Mechanical Turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France, Apr. 27–May 2). ACM Press, New York, 2013, 611–620.
2. Salehi, N., Irani, L.C., Bernstein, M.S. et al. We are Dynamo: Overcoming stalling and friction in collective action for crowd workers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea, Apr. 18–23). ACM Press, New York, 2015, 1621–1630.

Barry Brown and Airi Lampinen,
Stockholm, Sweden

Authors Respond:

We share the concerns Brown and Lampinen raise about crowdworker rights. In fact, AutoMan, by design, automatically addresses four of the five issues raised by workers, as described by Irani and Silberman in the letter's Reference 1: AutoMan never arbitrarily rejects work; pays workers as soon as the work is completed; pays workers the U.S. minimum wage by default; and automatically raises pay for tasks until enough workers agree to take them. Our experience reflects how much workers appreciate AutoMan, consistently rating AutoMan-generated tasks highly on Turkopticon, the requester-reputation site.

Daniel W. Barowy, Charles Curtsinger,
Emery D. Berger, and
Andrew McGregor, Amherst, MA

Computational Biology Is Parallel

Bonnie Berger et al.'s article "Computational Biology in the 21st Century: Scaling with Compressive Algorithms" (Aug. 2016) described how modern biology and medical research benefit from intensive use of computing. Microbiology has become data rich; for example, the volume of sequence data (such as strings of DNA and RNA bases and protein sequences) has grown exponentially, particularly since the initial sequencing of

the human genome at the start of the third millennium. Berger et al. pointed out that the biologist's growth exponent is greater even than Moore's Law. This growth has led to an increasing fraction of medical research funding being directed to data-rich 'omics. But the difference between what Moore's Law makes affordable and a greater medical exponent is itself in the long term also exponential. Berger et al. proposed smarter algorithms to plug the gap.

The article described bioinformatics' ready adoption of cloud computing, but the true parallel nature of much of e-biology went unstated; for example, Illumina next-generation sequencers can generate more than one billion short DNA strings, each of which can be processed independently in parallel. Top-end graphics hardware—GPUs—already contain several thousands of processing cores and deliver considerably more raw processing power than even multi-core CPUs. So it is no wonder that bioinformatics has turned to GPUs.

Berger et al. did mention BWA's implementation of the Burrows-Wheeler compression transform. BarraCUDA is an established port of BWA to Nvidia's hardware that was optimized for modern GPUs; results were presented at the 2015 ACM Genetic and Evolutionary Computation Conference.¹ Also, Nvidia keeps a list of the many bioinformatics applications and tools that run on its parallel hardware.

After CPU clocks maxed out at 3GHz more than 10 years ago, Moore's Law pushed 21st century computing to be parallel. GPUs and GPU-style many-core hardware are today at the center of the leading general-purpose parallel computing architectures. Much of microbiology data processing is inherently parallel. Computational biology and GPUs are a good match and set to continue to grow together.

Reference

3. Langdon, W.B. et al. Improving CUDA DNA analysis software with genetic programming. In *Proceedings of the 2015 ACM Genetic and Evolutionary Computation Conference* (Madrid, Spain, July 11–15). ACM Press, New York, 2015, 1063–1070.

W.B. Langdon, London, U.K.

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

© 2016 ACM 0001-0782/16/11 \$15.00

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:10.1145/2994590

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

Introducing CS to Newcomers, and JES As a Teaching Tool

Valerie Barr gets high schoolers thinking about CS, while Mark Guzdial mulls the benefits of Jython Environment for Students.



Valerie Barr
A Very Local Snapshot of K-12 CS Education

<http://bit.ly/2cqfRDg>
June 30, 2016

I had an interesting experience recently. I agreed to run a session on computer science for the STEP (Science and Technology Entry Program) students at Union College's Kenney Community Center. The range of students was large, from 7th to 12th grade. Usually in a session like this I start by asking two things. First, in what ways are computers already in their lives? Second, what kind of problems do they think computers can help solve? There were lots of interesting answers to both questions, and I am always intrigued there are many computers in their lives that kids (and adults) do not think about. It is also fun to get sidetracked by discussion about user interfaces—go heat something up in your microwave, for example, and contemplate the fact that you can run it for 99 seconds, but 100 gets you a minute!

My overall goal is to try to get the kids to think about problems that are not obviously numeric in nature. This allows us to talk about the dual challenge of figuring out a solution to the problem and then figuring out how to get the “data” into the computer. Of course, I want things to be fun too! I have two aids I typically use: SET (a visual perception card game) and 3D Squares matching puzzles (unfortunately, no longer available for purchase). These lead us to talk about how a human might solve a problem versus what we would have to tell the computer to do. Then there is that pesky question about how to get the SET cards or the puzzle pieces “into” the computer. This is where things got interesting.

I had asked the students, as they introduced themselves, to tell me what sort of prior experience they had had with computers. Just about everyone one had done something formal either in school or in a camp setting. Several of them had programmed in Python, at least one knew some Java, and several had done robotics work, but as a group,

none of them seemed to have learned anything about data representation inside the computer. Furthermore, though many of them were gamers, that experience did not seem to keep them from being surprised that you could come up with ways to represent cards or puzzle pieces inside the computer in ways that would allow a program to reason about those entities in the same ways that a human might. Certainly I do not expect K-12 CS education to get deep into mechanisms for data representation, but I do expect that today's students, who as a cohort carry around devices that hold audio files, image files, and video files, should be a bit more facile with the idea that other entities, like cards and hard plastic puzzle pieces, could also be represented in some way inside the computer.

We wrapped up with a quick overview of Traveling Salesperson (always fun to ask kids where in the world they would like to go) because I like to prove to them that, as amazing as computers are, there are still problems that computers cannot solve. That concept also seemed quite surprising to them.

As a snapshot of the state of K-12 CS education, my experience was not particularly encouraging, but it does make a strong case for both the coherent curricula and the formal professional development on which many in the CS community are working. I am ever optimistic that as I continue to do these sorts of sessions, I will begin to see evidence of the impact of new curricula and better-trained teachers. Stay tuned!



Mark Guzdial 14 Years of a Learner-Centered Python IDE

<http://bit.ly/2aNXAnC>

August 10, 2016

I recently discovered that the latest version of our Python IDE for learners, JES (Jython Environment for Students), passed 10,000 downloads (see the count at <http://bit.ly/2cyt1l2>); 10,562 when I started writing this post. We created JES in 2002 in support of our Media Computation course at Georgia Tech, which is a required course for students in our colleges of Liberal Arts, Business, and Design. Schools that adopted our Python Media Computation textbook have also mostly adopted JES, or used options that support the same libraries, such as Pythy or Pyjama. The current download count is probably an underestimate of users, since some schools download JES only once and then distribute it across campus.

Most student Python programmers use IDLE, the integrated development environment (IDE) provided with Python. While Python is more learner-centered than many languages used by professionals (see “Five Principles for Programming Languages for Learners,” <http://bit.ly/2chERzG>), the IDE should also be tailored for learners. Learners have different goals and needs than professional programmers. They need programming environments that take a learner-centered design approach (<http://bit.ly/2cEcQnz>), like JES or DrPython.

JES is likely one of the most-used pedagogical programming environments for Python, and the fact that it is still frequently used at the ripe old age of 14 suggests that it has been pretty successful. It is probably worth considering why it has worked. As lead on the team that built it and maintained it for the last 14 years, I am not a good judge for why it has worked.

Instead, I offer four brief stories about JES’s development and maintenance over the last 14 years.

Keep It Simple: From DrScheme and DrJava, we took the principle to keep it simple. We did not want an interface with many options for many kinds of uses. We wanted an interface that worked very well for learners.

In JES, you can only edit one file at a time, with an interaction pane (a REPL, <http://bit.ly/2cEcZrl>) for testing and running code. Most Python programs by professionals use multiple files. We opted to design for the beginner who could get lost in a sea of files. Having more than one file open requires some interface for switching files. One file means no additional interface. You can build bigger things in JES, but we optimized for the most common learner case.

Imagineering Media Computation to be Normal Python: We built JES to facilitate students programming in Media Computation. Students can program anything in JES; it is a full Python implementation. We added additional features to JES to facilitate Media Computation with a minimum of cognitive load.

We imagineered a community of practice around Media Computation (a story we told in <http://bit.ly/2bRCSmd>). In JES, it is normal for Python to know how to make pictures with `makePicture`, access individual pixels with `getPixels`, change colors with `setRed`, and access sound samples with `getSamples`. Invisibly, we load libraries for the student so that JES is a Python that supports programming with multimedia from the first moment of class. In normal Python, you can `print` anything to get its value. In JES, you explore any picture or sound to open an explorer to see color values in pictures or to visualize the samples in a sound.

Later on, we tell students there are libraries and how JES is automatically loading them. But on Day One, JES is a friendly Python that knows about pixels and pictures, sounds and samples, and frames and videos. No imported libraries, no dots, no extra details.

Sometimes, you explain errors after they occur: When we first started teaching Python to non-technical majors at Georgia Tech, students struggled with indentation errors. Even today in our ebooks (<http://bit.ly/2bW119J>), indentation problems in Python are among the most common and the most difficult for students to fix.

We decided to add a small feature to JES to help with indentation errors. We wanted indentation to be salient and obvious. JES draws a small

blue box around the lines in the same block as the current line (containing the cursor).

Now, when students come to me with indentation errors, I ask them “Where’s the blue box?” And they invariably ask, “*What* blue box?” I point it out to them. “Ohhhh—that blue box! Yeah, that’s useful.” Until students realize they have to attend to indentation, they do not see the support we are providing. *Telling* them that indentation is important and pointing out the blue box is not nearly as effective as encountering the error once.

Over the years, remove features: The original 2002 JES had more features in it than the most recent version. Two examples:

- We used to have a menu item to automatically turn in homework. It worked for a while at Georgia Tech, but then we changed how we handled homework. Other schools have always had their own mechanisms, and most schools try to make it easy to turn in homework. We dropped that.

- We worried about students loading a function from their homework file, then deleting the source code (by accident). The program might work correctly for them, but the file would be incomplete when grading. So, we wrote complicated code that compared the internal namespace to the source code, but that complicated code often caused problems of one sort. We decided it was not worth a big maintenance task for a rare error case, so we simply removed all of that complicated code.

The lesson here is Yogi Berra’s famous quote, “It’s tough to make predictions, especially about the future.” Some of the issues that seemed critical in the beginning simply were not all that important in actual practice later. The features with more staying power were the ones that we added in response to learner behavior. Our most successful features were the ones we added to meet real learners’ needs, not what we initially imagined the needs to be.

Valerie Barr is a professor in the Computer Science

Department at Union College, Schenectady, NY.

Mark Guzdial is a professor in the College of Computing at the Georgia Institute of Technology, Atlanta, GA.

© 2016 ACM 0001-0782/16/11 \$15.00

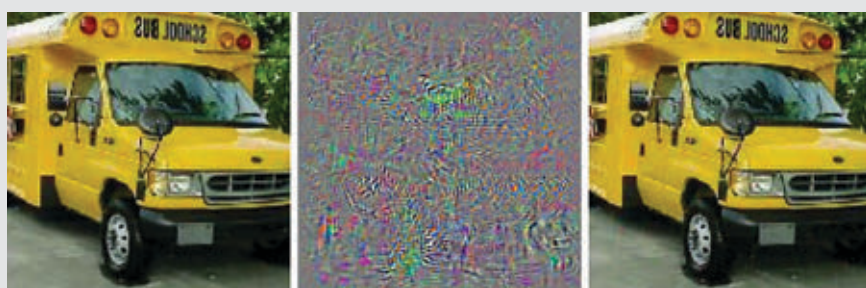
Learning Securely

Because it is easy to fool, machine learning must be taught how to handle adversarial inputs.

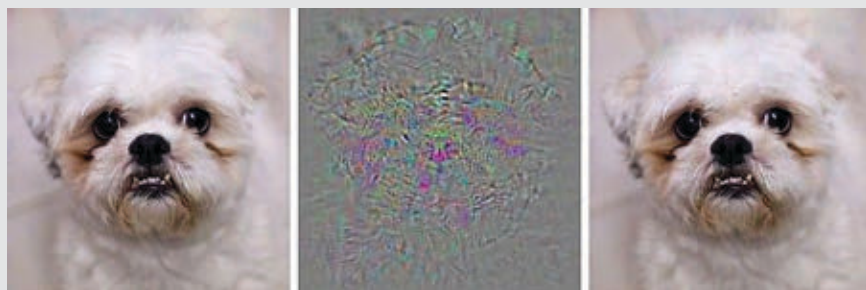
OVER THE PAST five years, machine learning has blossomed from a promising but immature technology into one that can achieve close to human-level performance on a wide array of tasks. In the near future, it is likely to be incorporated into an increasing number of technologies that directly impact society, from self-driving cars to virtual assistants to facial-recognition software.

Yet machine learning also offers brand-new opportunities for hackers. Malicious inputs specially crafted by an adversary can “poison” a machine learning algorithm during its training period, or dupe it after it has been trained. While the creators of a machine learning algorithm usually benchmark its average performance carefully, it is unusual for them to consider how it performs against adversarial inputs, security researchers say.

The emerging field of adversarial machine learning is exploring these vulnerabilities. In the past few years, researchers have figured out, for example, how to make tiny, imperceptible changes to an image to fool vision processing systems into interpreting an image humans see as a school bus as an ostrich instead. Such deceptions often can be carried out with virtually no knowledge about the inner workings



School bus + tiny adversarial perturbation = “ostrich”



Dog + tiny adversarial perturbation = “ostrich”

Adversarial input can fool a machine-learning algorithm into misperceiving images.

of the machine learning algorithm under attack.

Machine learning can be easy to fool, computer scientists warn. “We don’t want to wait until machine learning algorithms are being used on billions of devices, and then wait for people to mount attacks,” said Nicholas Carlini, a graduate student in adversarial machine learning at

the University of California, Berkeley, who has crafted audio files that sound like white noise to humans, but like commands to speech recognition algorithms. “We need to think of the attacks as early as possible.”

Attacking the Black Box

Adversarial machine learning has been studied for more than a decade in a few

security-related settings, such as spam filtering and malware detection. However, the advent of deep neural networks has greatly expanded the scope both of the potential applications of machine learning and the attacks that can be carried out upon it.

Until very recently, it did not make much sense to study how to trick a neural network, said Ian Goodfellow, an adversarial machine learning expert at OpenAI, a research institute in San Francisco. “Five years ago, if I said ‘hey, I found an example that my neural network misclassifies,’ that would be the rule, not the exception,” he said. “Only recently has machine learning become accurate enough for it to be interesting when it’s inaccurate.”

A paper posted online in 2013 launched the modern wave of adversarial machine learning research by showing, for three different image processing neural networks, how to create “adversarial examples”—images that, after tiny modifications to some of the pixels, fool the neural network into classifying them differently from the way humans see them. Subsequent papers have created many more such adversarial examples—for instance, one that forced a deep neural network to classify what humans see as a “stop” sign instead as a “yield” sign.

While it is not clear whether such examples could be used in the real world to attack self-driving cars, “it’s important for the designers of self-driving cars to understand that one subsystem might totally fail because of one adversarial example,” said Goodfellow, one of the authors of the 2013 paper. “It highlights the importance of using several redundant safety systems, and making sure an attacker can’t compromise them all at the same time.”

The reason deep neural networks can be fooled by adversarial examples, Goodfellow said, is that while these complex networks are made up of many layers of processors, each individual layer essentially uses a piecewise linear function to transform the input. This linearity makes the model prone to overconfidence about its inferences, he said.

“If you find a direction to move in that makes the output more likely to say the image is a cat, then you can keep moving in the same direction for

“Only recently has machine learning become accurate enough for it to be interesting when it’s inaccurate.”

a long time, and the output will be saying more and more strongly that it’s a cat,” Goodfellow said. “This says that neural networks can extrapolate in really extreme ways.” Subtle adjustments to pixels that nudge an image along this “cat” direction can make the neural network mistake the image for a cat, even if to a human being, the altered image looks indistinguishable from the original.

At the same time, linearity does not explain all the vulnerabilities of machine learning algorithms to adversarial examples. Last year, for instance, three researchers at the University of California, Berkeley—Alex Kantchelian, Doug Tygar, and Anthony Joseph—showed a highly nonlinear machine learning model called “boosted trees” is also highly susceptible to adversarial examples. “It’s easy to design evasions for them,” Kantchelian said.

The earliest adversarial examples were concocted by researchers who had full access to the machine learning model under attack. Yet even with those first examples, researchers started noticing something strange: examples designed to fool one machine learning algorithm often fooled other machine learning algorithms, too.

That means an attacker does not necessarily need access to a machine learning model’s architecture or training data to attack it. As long as the attacker can query the model to see how it classifies various data, he or she can use that data to train a different model, build adversarial examples for that model, then use those to trick the original model.

Researchers do not fully understand why adversarial examples transfer from one model to another, but

ACM Member News

SEEKING DISTRIBUTED AI



“I was 14 years old in 1980 and a friend said the local Tandy store was selling Radio Shack

computers, and I didn’t believe him,” recalls Michael Wooldridge, head of the Department of Computer Science at the University of Oxford, where he also serves as a computer science professor. Wooldridge and his buddy went to have a look, and started hanging around the electronics store. “We persuaded the guys in the shop to let us write programs on the computer in the window, and within a month I had written my first program. I knew that was what I wanted to do.”

Wooldridge went on to earn an undergraduate degree in computer science in 1989, specializing in networks and artificial intelligence (AI). He was interested in becoming a research scientist, and it occurred to him that you could put networks and AI together and have something that constituted distributed artificial intelligence. “I wondered what that might look like, and what the potential applications might be.”

Driven by curiosity, Wooldridge earned a Ph.D. in distributed artificial intelligence from the University of Manchester. He became a full professor at the University of Liverpool in 2000, and joined the University of Oxford as a professor of computer science in 2012.

Today, Wooldridge, an ACM Fellow, is focused on the intersection of logic, computational complexity, and game theory. “My main research interests are in the behavior of network systems—where network nodes have their own preferences, goals, and agendas—and using techniques from model checking together with ideas from game theory to study these systems.”

—John Delaney

they have confirmed the phenomenon in increasingly broad settings. In a paper posted online in May, Goodfellow, together with Nicolas Papernot and Patrick McDaniel of Pennsylvania State University, showed that adversarial examples transfer across five of the most commonly used types of machine learning algorithms: neural networks, logistic regression, support vector machines, decision trees, and nearest neighbors.

The team carried out “black box” attacks—with no knowledge of the model—on classifiers hosted by Amazon and Google. They found after only 800 queries to each classifier, they could create adversarial examples that fooled the two models 96% and 89% of the time, respectively.

Not every adversarial example crafted on one machine learning model will transfer to a different target, Papernot said, but “for an adversary, sometimes just a small success rate is enough.”

Getting Ready for Prime Time

Human vision systems have adversarial examples of our own, more commonly known as optical illusions. For the most part, though, people process visual data remarkably effectively. As computer vision systems approach human-level performance on particular tasks, adversarial examples offer a new way to benchmark performance, besides measuring how well an algorithm performs on typical inputs. Adversarial inputs “help find the flaws in neural networks that do really well on the more traditional kinds of benchmarks,” Goodfellow said.

“In a perfect world, what I would like to see in papers is, ‘Here is a new machine learning algorithm, and here is the standard benchmark for how it does on accuracy, and here is the standard benchmark on how it performs against an adversary,’ Carlini said. Adversarial machine learning experts have some leads on what such a benchmark should look like, Papernot said, but no such benchmark has been established yet.

Software developers have a history of adding security to their products after the fact rather than integrating it into the development phase, Carlini said, even though that approach

makes it easy to miss vulnerabilities. Now, he warned, the machine learning community is poised to make the same mistake: machine learning is a huge field, but only a tiny slice of the community is focused on security. “We should be developing security in from the start,” he said. “It shouldn’t be an afterthought.”

The good news is that adversarial examples do not just offer a way to benchmark how vulnerable a machine learning model is; they also can be used to make the model more robust against an adversary and, in some cases, even improve its overall accuracy.

Some researchers are making machine learning algorithms more robust by essentially “vaccinating” them: adding adversarial examples, correctly labeled, into the training data. In a 2014 paper, Goodfellow and two colleagues demonstrated in a classification task involving handwritten numbers that this kind of training not only made it harder to fool a neural network, but even brought down its error rate on non-adversarial inputs. Kantchelian, Tygar, and Joseph have shown a similar vaccination process can greatly improve the robustness of boosted trees.

Most of the research on adversarial machine learning has focused on “supervised” learning, in which the algorithm learns from labeled data. Adversarial training offers a potential way for machine learning algorithms to learn from unlabeled data—an exciting prospect, Goodfellow said, since labeled data is expensive and time-consuming to create. In a paper posted online in May, Takeru Miyato of Kyoto University—along with Goodfellow and Andrew Dai of Google Brain—was able to improve the performance of a movie review classifier by taking unlabeled reviews, creating adversarial versions of them, and then teaching the classifier to group those reviews in the same category. With the plethora of texts available on the Internet, “you can get as many examples for unlabeled learning as you want,” Goodfellow said.

Meanwhile, in another paper published in May, Papernot, McDaniel, and other researchers detail the creation of another defense against adversarial examples, called “defensive distillation.” This approach uses a neural network to label images with prob-

ability vectors instead of single labels, and then trains a new neural network using the probability vector labels; this more nuanced training makes the second neural network less prone to overfitting. Fooling such a neural network, the researchers showed, required eight times as much distortion to the image as before the distillation.

The work done so far is just a start, Tygar said. “We haven’t even begun to understand all the potential different environments in which you might have an attack” on machine learning. The field of adversarial machine learning is full of opportunities, he said. “The area is ready to move.”

Although Tygar thinks it is a near certainty we will start seeing more attacks on machine learning as its use becomes prevalent, it would be a mistake, he said, to conclude machine learning is too risky to be used in adversarial settings. Rather, he said, the question should be, “How can we strengthen machine learning so it is ready for prime time?” **C**

Further Reading

Goodfellow, I., Shlens, J., and Szegedy, C. **Explaining and Harnessing Adversarial Examples**
<http://arxiv.org/pdf/1412.6572v3.pdf>

Kantchelian, A., Tygar, J. D., and Joseph, A. **Evasion and Hardening of Tree Ensemble Classifiers**
<http://arxiv.org/pdf/1509.07892.pdf>

Miyato, T., Dai, A., and Goodfellow, I. **Virtual Adversarial Training for Semi-Supervised Text Classification**
<http://arxiv.org/pdf/1605.07725v1.pdf>

Papernot, N., McDaniel, P., Wu, X., Jha, X., and Swami, A. **Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks**
Proceedings of the 37th IEEE Symposium on Security and Privacy, May 2016.

Papernot, N., McDaniel, P., and Goodfellow, I. **Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples**
<http://arxiv.org/pdf/1605.07277v1.pdf>

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. **Intriguing Properties of Neural Networks**
<https://arxiv.org/pdf/1312.6199v4.pdf>

Erica Klarreich is a mathematics and science journalist based in Berkeley, CA.

© 2016 ACM 0001-0782/16/11 \$15.00

Blockchain Beyond Bitcoin

Blockchain technology has the potential to revolutionize applications and redefine the digital economy.

BLOCKCHAIN TECHNOLOGY HAS attracted attention as the basis of cryptocurrencies such as Bitcoin, but its capabilities extend far beyond that, enabling existing technology applications to be vastly improved and new applications never previously practical to be deployed.

Also known as distributed ledger technology, blockchain is expected to revolutionize industry and commerce and drive economic change on a global scale because it is immutable, transparent, and redefines trust, enabling secure, fast, trustworthy, and transparent solutions that can be public or private. It could empower people in developing countries with recognized identity, asset ownership, and financial inclusion; and it could avert a repeat of the 2008 financial crisis, support effective healthcare programs, improve supply chains and, perhaps, clean up unethical behavior in high-value businesses such as diamond trading.

Blockchain, like the Internet, is an open, global infrastructure that allows companies and individuals making transactions to cut out the middleman, reducing the cost of transactions and the time lapse of working through third parties. The technology is based on a distributed ledger structure and consensus process. The structure allows a digital ledger of transactions to be created and shared between distributed computers on a network. The ledger is not owned or controlled by one central authority or company, and can be viewed by all users on the network.

When a user wants to add a transaction to the ledger, the transaction data is encrypted and verified by other computers on the network using cryptographic algorithms. If there is consensus among the majority of computers that the transaction is valid, a new



block of data is added to the chain and shared by all on the network. Transactions are secure, trusted, auditable, and immutable. They also avoid the need for copious, often duplicate, documentation, third-party intervention, and remediation.

Blockchains can be either public and unpermissioned, allowing anybody to use them (bitcoin is a case in point) or private and permissioned, creating a closed group of known participants working, perhaps, in a particular industry or supply chain.

Michael Versace, global research director for digital strategies at research firm IDC, describes blockchain as an industry and innovation accelerator based on the capability of the third platform of technology—the first platform being mainframes and their networks,

the second Internet, personal computers, and local area networks. The third platform delivers computing anywhere, immediately, and allows organizations to deploy and consume computing resources in shared communities.

Says Versace, “The core capabilities of the third platform of technology are beyond any we have seen before. Innovation accelerators like blockchain mean we can achieve technology value outcomes that we couldn’t achieve before.”

This is promising, but there are caveats. Sandeep Kumar, managing director of capital markets and a blockchain specialist at digital business consulting and technology services firm Synechron, names data privacy, scalability, and interoperability as three key challenges to blockchain technology that are pervasive across applica-

Benefits of Blockchain in Financial Services

Secure transactions

- ▶ Avoid information leakage
- ▶ Reduce transaction time
- ▶ Remove transaction intermediaries
- ▶ Reduce risk of fraud and cybercrime
- ▶ Observe transactions in real time

Source: IBM

Early Adopter Views on Blockchain

- ▶ Platform openness is required.
- ▶ Features like identity, privacy, security, operations management, and interoperability need to be integrated.
- ▶ Performance, scale, support, and stability are crucial.
- ▶ Consortium blockchains, which are permissioned networks on which consortium members may execute contracts, are ideal.

Source: Microsoft

tions and have not yet been solved cleanly. Other sticking points are data transfer, and integrating with existing systems and sometimes security, which depends on application coding.

Financial Applications

The financial services sector, which must innovate to cut the costs of legacy systems and manage increasing regulation, is leading the way with blockchain and taking advantage of the technology's security, immutability, transparency, and ability to cut out the middleman. Fintech startup R3, backed by over 40 global banks, is developing a standardized architecture for private ledgers that could significantly cut the cost and time of settling transactions. Similarly, the Linux Foundation's Hyperledger project is an industry initiative including tech giant IBM that is evolving open source

technology and building the foundation of a standardized, production-grade digital ledger.

Deloitte is working with clients and startups to develop solutions including Smart Identity, which can support banks' regulatory client onboarding and Know Your Customer (KYC) processes, while individual financial institutions, insurance companies, exchanges, and solutions vendors also have thrown their weight behind blockchain.

Many are taking advantage of the technology's ability to act as a giant time stamp. Nasdaq is using its Linq blockchain technology to complete and record private securities transactions, and the Depository Trust & Clearing Corporation, working with market participants and technology firm Axoni, is managing post-trade events for credit default swaps. Regulators are also interested in the technology, as its transparency and integrity allow market activity to be monitored in real time.

These early applications show great potential, but there are problems around data privacy, scale and latency in financial markets. The privacy issue is about how much information needs to be exposed to verify a transaction. This could be more than at present and could compromise the privacy of a trade. Scale and latency are also issues in a market managing huge data volumes. These problems are being addressed by industry consortia and individual firms, but robust solutions remain elusive.

Commercial Applications

In the commercial world, two startups making progress with blockchain are Factom and Everledger.

Factom's focus is on securing data. The company is participating in the Honduran land registry project and working on a number of projects in China, including data infrastructure for 80 smart cities, financial technology solutions, and integrating blockchain technology with electronic data notarization services to enhance integrity in information management.

The company has also secured funding from the U.S. Department of Homeland Security's Science and Technology Directorate under the 'Blockchain Software to Prove Integrity of Captured

Data from Border Devices' project.

Everledger's focus is on the identity and legitimacy of objects. Blockchain works well here because its history cannot be changed and it enables trust by consensus. The company's initial work provides a distributed ledger of diamond ownership and transaction history verification for owners, insurance companies, claimants, and law enforcement agencies. The system assists with prevention of fraud in the supply chain, but also helps consumers decide whether to buy particular diamonds.

Leanne Kemp, founder and CEO of Everledger, explains, "The ultimate goal is to track diamonds from mine to market, so that consumers can see if correct duties and taxes have been paid and whether a diamond is a 'blood diamond' that has been mined and traded in a war zone and contributed to human atrocity." The company also is considering applying its technology to other big-ticket items, such as fine art, vintage cars, and wine.

In addition, blockchain is expected to be well suited, with the addition of smart contracts that use computerized transaction protocols to execute the terms of contracts agreed by users of a blockchain, to applications such as product manufacturing, supply chain management, vehicle provenance, and sharing resources such as electricity.

Emin Gün Sirer, an associate professor of computer science at Cornell University and a participant in a number of blockchain projects, says blockchain could democratize the in-

The financial services sector takes advantage of blockchain's security, immutability, transparency, and ability to cut out the middleman.

insurance industry by using smart contracts to pay out against insurance policies without policyholders having to make a claim. He adds: “The Internet of Things could be an enormous application area where people want to communicate with devices, but not through intermediaries. There is no killer app yet, but it is likely to feature the transparency of blockchain.”

While start-ups can skip some of the challenges presented by blockchain technology, established firms must set up a network of blockchain participants, perhaps suppliers and customers, and agree on technology protocols. Commercial firms, like others, will also hit the interoperability barrier identified by Synechron and by Microsoft in feedback from early blockchain adopters. Kumar explains: “Blockchain is evolving in many ecosystems, such as Hyperledger and Ethereum, but there needs to be a native way to integrate blockchains that would allow, for example, a transaction on Hyperledger to invoke information from Ethereum.”

Sirer warns of less-advantageous applications such as gambling and ongoing security problems. He cites the spectacular rise and fall of The DAO, a distributed autonomous organization based on Ethereum technology that acted as an investment vehicle, raising \$220 million, then swiftly losing \$53 million to a hacker. “We looked at the DAO code and found it was written so badly it was open to attack from nine different angles. Incidents like this uncover the need for more multi-disciplinary research on blockchain technology.”

Developing Countries

The potential of blockchain is also diverse in developing countries, but where the commercial world is concentrating on outstanding technology challenges, developing countries are initially focusing on the trust element of blockchain.

Mariana Dahan, senior operations officer at the World Bank in charge of the 2030 development agenda and United Nations (U.N.) relations, says, “We believe blockchain is a major breakthrough and has great potential. It will make an impact on, and bring value to, any transaction that requires trust, a social resource that is all too of-

The potential of blockchain is diverse in developing countries, where the initial focus is on the trust element.

ten in short supply.”

Dahan suggests the trust element of blockchain will play well into the 2030 Sustainable Development Goals adopted by U.N. members in 2015 and designed to end poverty, protect the planet, and ensure prosperity for all. More specifically, she notes high-potential applications of blockchain in land registration, digital identity, and finance for small and medium-sized enterprises.

Land registration and awareness of its relevance to issues such as food security, climate change, urbanization, and indigenous people's rights has increased over recent years, yet the Independent Evaluation Group of the World Bank says 70% of the world's population lacks access to proper land titling or demarcation.

Beginning to solve this problem are projects like one in the Republic of Georgia, where the National Agency of Public Registry is working with BitFury on a pilot project that will use a transparent, secure ledger to manage land titles and, if successful, cut property registration fees by up to 95%, increase transparency of land ownership, and reduce fraud. A similar project partially funded by the World Bank is being developed in Honduras, where Factom is working with the government to prototype a blockchain-based land registry.

Beyond land registration, Dahan explains how the ability to store and update property titles on a blockchain could, for the first time, allow poor people to assert reliable title claims to their homes and use them as collateral for borrowing. Small and medium-sized enterprises also could prove ownership of assets, perhaps equip-

ment or livestock, and provide access to working capital and, by extension, a wider market.

Digital identity enabled by blockchain has the potential to change lives. Says Dahan, “If blockchain technology can be used to secure robust, self-sovereign digital identities around personal data, there's a real possibility that people in places with poor documents, registries and rules of law can establish trusted measures of their good reputation. This would allow them to assert who they are and access proof of their digital identity anywhere using a private key.”

With the benefit of digital identity, many of the world's two billion unbanked individuals could store their identities on a blockchain, permission banks to fulfill regulatory requirements such as Know Your Customer, and gain access to bank accounts, loans, and other financial services previously inaccessible to them.

The potential of blockchain to revolutionize applications and drive global economic change is certainly there, but problems persist in wide-scale execution. As Kumar concludes: “Blockchain is not yet ready for prime time.” **C**

Further Reading

The future of financial infrastructure: An ambitious look at how blockchain can reshape financial services, World Economic Forum, <https://www.weforum.org/reports/the-future-of-financial-infrastructure-an-ambitious-look-at-how-blockchain-can-reshape-financial-services/>

Cuomo, J. How Businesses and Governments Can Capitalize on Blockchain, <http://www.ibm.com/blogs/think/2016/03/16/how-businesses-and-governments-can-capitalize-on-blockchain/>

Sirer, E.G. Introducing Virtual Notary Hacking, Distributed <http://hackingdistributed.com/2013/06/20/virtual-notary-intro/>

Casey, M., and Dahan, M. Blockchain technology: Redefining trust for a global, digital economy http://blogs.worldbank.org/ic4d/blockchain-technology-redefining-trust-global-digital-economy?cid=EXT_WBBlogSocialShare_D_EXT

Sarah Underwood is a technology writer based in Teddington, U.K.

© 2016 ACM 0001-0782/16/11 \$15.00

Farm Automation Gets Smarter

As fewer people work the land, robots pick up the slack.

FIELD FARMING IS “the world’s oldest profession,” and not just because food plants have been cultivated for over 10,000 years. Its individual practitioners are old as well, the median age rising rapidly as young people abandon the farming lifestyle (the U.S. Department of Agriculture reports a median age of 58 in 2012, up from 55 in 2002, with other countries showing similar data). Those who remain face the same repetitive work of seeding, weeding, feeding, and harvesting, the tedium of each task increasing as farms grow ever larger.

However, today’s agricultural robots excel at repetitive tasks, letting farmers tend to more strategic matters.

“When we develop robots, people always say they should increase yields or lower costs, but that’s not always the case,” says Eldert van Henten, a professor in Wageningen University’s Farm Technology Group. “Robots that offer the farmer time to devote attention to business and management also have economic value.”

Allen Lash, CEO of the farm management company Family Farms Group, sees agricultural robots filling in for the shrinking agricultural workforce. “In a lot of our small, rural communities, we just don’t have adequate labor to run big, sophisticated equipment anymore,” he says. “With autonomous equipment, you’ll need less labor, although the people you have will need more computer technology skills. They’ll control equipment remotely to a great extent, from a control room, scanning at least three cameras all the time. So we’ll get more acres covered in a more efficient way; maybe several thousand acres per employee, instead of today’s 1,000.”

Proponents believe robotic agriculture also will help ensure food is available for everyone, everywhere, from local sources. “Every country is now worried about food security, and the country



The BoniRob is a multipurpose robotic platform for agricultural applications featuring independently steerable drive wheels and adjustable track width.

next door has cheaper labor” said Salah Sukkarieh, director of Research and Innovation at the Australian Centre for Field Robotics (ACFR) at the University of Sydney.

Food security is one of the driving forces behind “The Vegetable Factory,” an indoor farm that the Japan-based company SPREAD Co. Inc. plans to open in Kyoto next year. In addition to computer-controlled lighting, temperature, and humidity, virtually every aspect of a plant’s development from seedling to harvest will be automated. “We expect automation to reduce human error, therefore stabilizing the quality and quantity of production,” says SPREAD global marketing manager J.J. Price. “Such ‘plant factories’ will increase in the future to cover agricultural issues, food shortages, and environmental problems.”

Command and Control

Farm automation is particularly difficult because the environment varies so much—especially when compared to clean labs. “Computer scientists need to get outdoors more,” said ACFR’s Sukkarieh. “A lot of the [computer vision] algorithms get developed on very structured data, like a camera focusing on objects in a kitchen. But in the

field, the wind blows leaves around, lighting conditions change, you get a bit of dust; these all affect the algorithm. We try things out in the lab, but then have to play with them because they don’t work as well outdoors, in an unstructured environment.”

Traditional farm equipment also adds unpredictability to the mix, as Autonomous Tractor Corporation (ATC) president and CEO Kraig Schulz learned when his company started planning to build an autonomous tractor. “Traditional [drive train] vendors couldn’t deliver what we needed, so we turned to electric drive trains, like Google and Tesla have for their cars. A traditional drive train is hard to control; there’s a lot of mechanical nonsense between the driver and the ground. But I can control an electric motor to milliseconds and fractions of an inch.” The company later released its “eDrive” system as an electric engine replacement for tractors already in the field.

As another step toward autonomous farming, ATC next supplemented traditional GPS navigation with a local laser-and-radio system named AutoDrive. “If you talk to farmers, they’ll say that GPS is good, but far from perfect. They’re in very rural areas, where reception isn’t great, and

GPS tends to suffer from problems like sunspots, reflections, and tree coverage,” says Schulz. “A lot of driverless cars use optics and radar and such, because there’s stuff around them to see. When you’re in a field, there’s nothing to see! So you need a second or third source of data to get relative positioning in the field.” The company is now working on a system that can be retrofit to any tractor with its eDrive, adding AutoDrive and its own “FieldSmart” artificial intelligence software.

An “App” for Planting

Another challenge for autonomous farming is the variety of tasks that agriculture demands. The hardest, according to Wageningen University’s van Henten, is selective harvesting. “Robots can do the trick now at relatively slow speeds, and without full success,” he says. “We still have a long way to go. But humans excel in terms of eye-hand coordination, intelligence, flexibility, and robustness against variation in the environment. So harvesting is done by humans very effectively.”

One problem, he says, is that “Plants that are members of the same type can differ a lot due to a lack of water, light, or nutrition.” Switching from one variety to another is also difficult. “Consider the tomato; usually a tomato is red, but there are some that are yellow. There are also different shapes, and beefsteak and cherry tomatoes have very different sizes. But the robot still has to recognize them as tomatoes.”

A somewhat simpler task is weeding, which is the bailiwick of the French company Naïo Technologies. “We started with weeding because it’s very arduous for farmers,” says Julien Laffont, the company’s international business developer, “and it’s a repetitive task, so it’s perfect for robots.” The company’s currently available “Oz” system uses two cameras and a laser to guide itself between rows of crops as it drags a weeding tool behind it.

Oz controls only the robot, not the tool it drags behind. Laffont said the company’s future “Dino” robot also will control the tools themselves, letting them do more than simply avoid harming crops. “We’ll use cameras that analyze the image to detect, circle, and count crops. We might also have appli-

cations to analyze the images for things like disease.”

A different approach to weeding can be found in “Lettucebot,” a product from U.S.-based Blue River Technology that puts its intelligence in the implement, combining vision technology with microspraying to apply herbicides to weeds and overplanted lettuce sprouts. According to the company, Lettucebot can identify 1.5 million plants per hour, weeding and thinning a 40-acre field within a day. For vice president of business development Ben Chostner, speed is only one benefit to this precision approach. “Most inputs or chemicals on the farm are applied in a broadcast way, with hardly any sensors. It’s like if someone in San Francisco had an infection, and the only solution was to give everyone an antibiotic; that would solve the problem, but it would be very expensive and inefficient, and would lead to negative consequences like resistance. That’s the same thing that’s happening with weed control today, because those are the tools we have.”

ATC’s Schulz looks forward to the day that weeding, planting, and harvesting are all treated as “apps” that a single machine (with specialized attachments) could learn to perform through artificial intelligence (AI) techniques. “The computer has to become a farmer just like a person becomes a farmer, over time, by learning. That’s where AI comes in. The driver should think of the tractor riding along like a son or daughter: it observes what the farmer is doing and thinks, ‘this is what I have to replicate.’ The tractor will start with small tasks and grow competence over time. Its tillage ‘app’ is very different from a planting ‘app’ because the task has fundamentally different controls, but the learning process could be similar.”

From Automation to Automatons

Kubota Corp. director and senior managing executive officer Satoshi Iida differentiates modern, computer-driven agents from centuries-old farming technology like the seed drill. “An agricultural machine can be defined as a ‘robot’ when it has the three technology components,” says Iida, who also is general manager of Kubota’s Research and Development (R&D) headquarters, and of the company’s Water and Environment R&D function. “It must have sensors

that perceive its own situation; intelligence and control mechanism for determining and selecting the appropriate action by itself; and driving mechanisms for achieving what it has determined.”

Blue River Technology’s Chostner says, “A robot sees, decides, and acts. Then it closes that loop by seeing how it acted and adjusting its future actions.” On the other hand, Chostner says, “We’d much rather focus on the simplest tool that will create value for our customers, rather than the ultimate ‘robot’ that is the company vision. Robots are always something out in the future; automation is something that’s here today, creating value. We found that farmers don’t like to buy robots; they like to buy machines that work.”

Except for limited tasks by “machines that work,” fully automated growing may be years away. Kubota’s Iida believes the technology is now in the second of four stages: we have achieved automation of individual functions and auto-steering under operator control, but have yet to master unguided operation and fully autonomous farming.

Sukkariieh believes economic changes have cleared the way to reach that final phase. “Farming is probably the last major primary industry to focus on automation, and that’s because of the cost,” he says, “but technology costs have dropped dramatically. Even places with cheaper labor are considering agricultural automation. I think over the next year or two, you’ll see a large amount of activity happening in this space.” 

Further Reading

Robotics & Automation Society, Technical Committee on Agricultural Robotics and Automation, IEEE: <http://www.fieldrobot.com/ieeeras/>

Past IEEE webinars on agricultural robotics: <http://www.fieldrobot.com/ieeeras/Events.html>

Video of Lettucebot: <https://www.youtube.com/watch?v=yCPe9Sy0TFY>

Robohub focus on agricultural robotics: <http://robohub.org/tag/robhub-focus-on-agricultural-robotics/>

Australian Centre for Field Robotics (Agriculture Projects): <http://sydney.edu.au/acfr/agriculture>

Tom Geller is an Oberlin, Ohio-based writer and documentary producer.

© 2016 ACM 0001-0782/16/11 \$15.00

Privacy and Security Cyber Defense Triad for Where Security Matters

Dramatically more trustworthy cyber security is a choice.

IN THE EARLY days of computers, security was easily provided by physical isolation of machines dedicated to security domains. Today's systems need high-assurance controlled sharing of resources, code, and data across domains in order to build practical systems. Current approaches to cyber security are more focused on saving money or developing elegant technical solutions than on working and protecting lives and property. They largely lack the scientific or engineering rigor needed for a trustworthy system to defend the security of networked computers in three dimensions at the same time: mandatory access control (MAC) policy, protection against subversion, and verifiability—what I call a defense triad.

Fifty years ago the U.S. military recognized subversion^a as the most serious threat to security. Solutions such as cleared developers and technical

The security problem will remain as long as manufacturers remain committed to current system architectures, produced without a firm requirement for security.

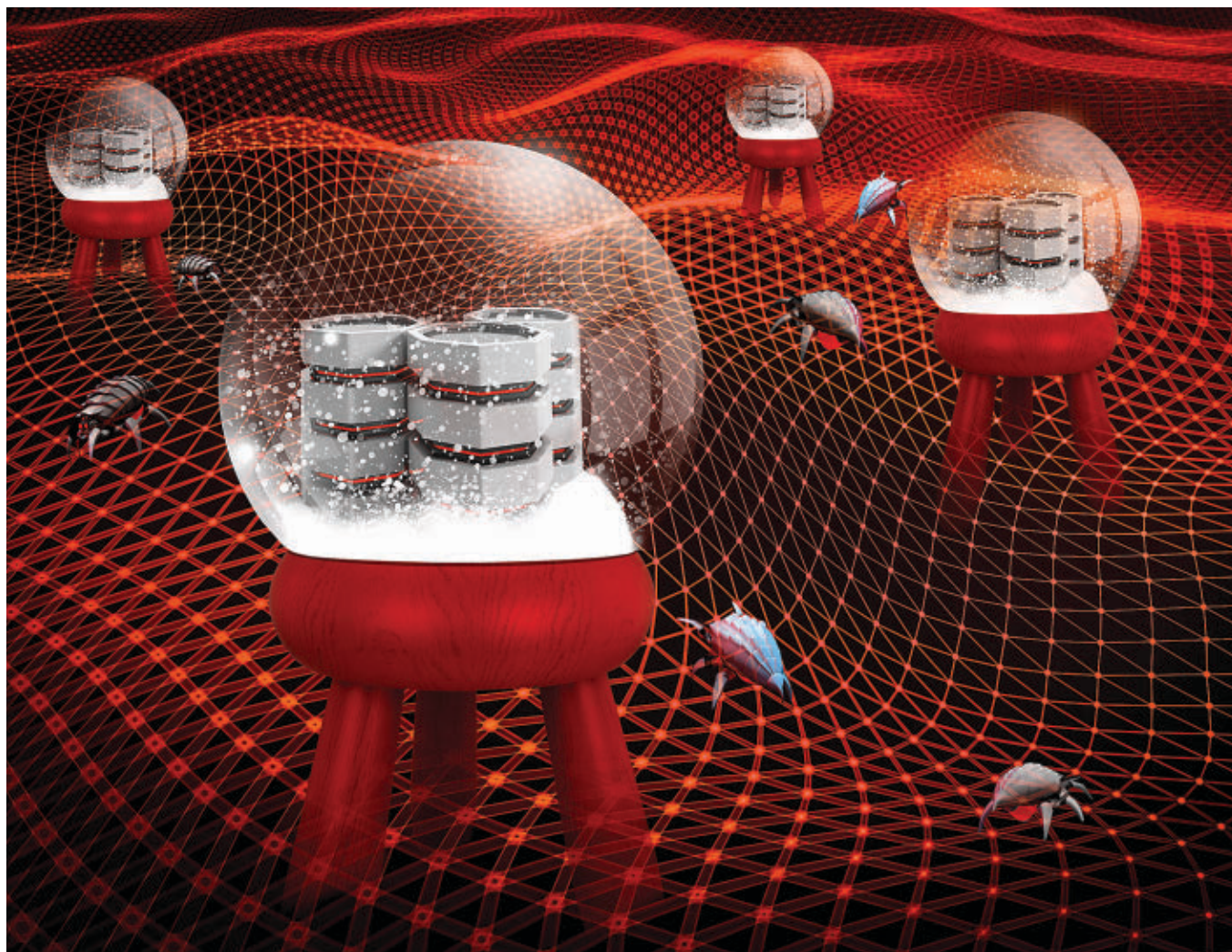
development processes were neither scalable nor sustainable for advancing computer technology and growing threats. In a 1972 workshop, I proposed “a compact security ‘kernel’ of the operating system and supporting hardware—such that an antagonist could provide the remainder of the system without compromising the protection provided.” I concluded: “We are

confident that from the standpoint of technology there is a good chance for secure shared systems in the next few years. However, from a practical standpoint the security problem will remain as long as manufacturers remain committed to current system architectures, produced without a firm requirement for security. As long as there is support for ad hoc fixes and security packages for these inadequate designs, and as long as the illusory results of penetration teams are accepted as a demonstration of computer system security, proper security will not be a reality.”⁸

Current Approaches Aren't Working

Our confidence in “security kernel” technology was well founded, but I never expected decades later to find the same denial of proper security so widespread. Although *Forbes* reports spending on information security reached \$75 billion for 2015, our adversaries are still greatly outpacing us. With that large financial incentive for vested interests, resources are mostly devoted to doing more of what we knew didn't work then, and still doesn't.

^a As characterized by Anderson, et al.,² “System subversion involves the hiding of a software or hardware artifact in the system that creates a ‘backdoor’ known only to the attacker.”



Why does cyber security seem so difficult? Today's emphasis on surveillance and monitoring tries to discover that an adversary has found and exploited a vulnerability to penetrate security and cause damage—or worse, subverted the security mechanism itself. Then that hole is patched. But science tells us trying to make a system secure in this way is effectively non-computable. Even after fixing known flaws, uncountable flaws remain. Recently, Steven Lipner, formerly of Microsoft, wrote a *Communications* Privacy and Security column advocating technical “secure development processes.”⁶ But, similar to surveillance, “as new classes of vulnerabilities ... are discovered, the process must be updated.”

This paradigm has for decades been known as “penetrate and patch.” The defender needs to find and patch most (if not all) of the holes, while the adversary only needs to find and exploit one

remaining hole. Even worse, a witted adversary has numerous opportunities to subvert or sabotage a computer's protection software itself to introduce insidious new flaws. This is an example

of “malware,” a preferred attack for many of the most serious breaches. An IBM executive a few years ago described the penetrate-and-patch cycle as “an arms race we cannot win.”⁵

Figure 1. Cyber security defense triad.

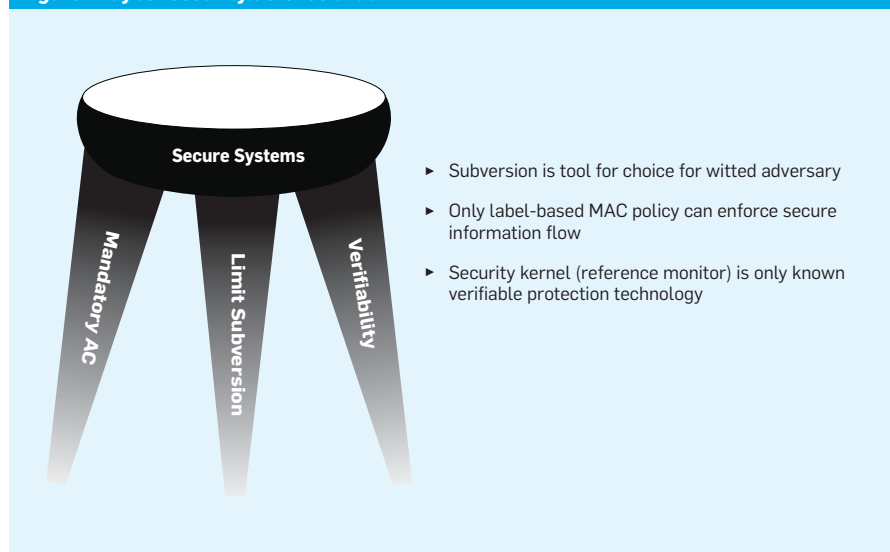
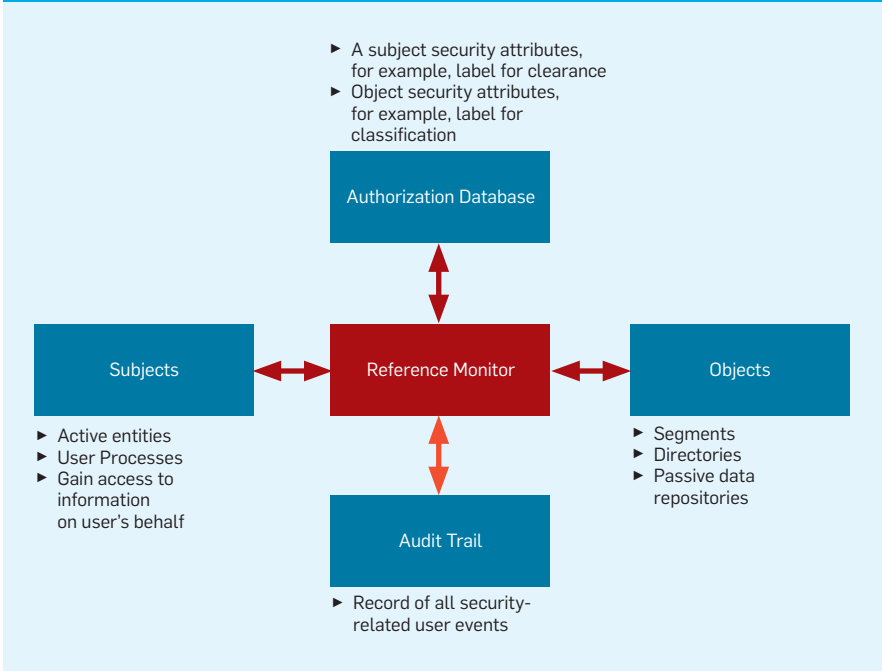


Figure 2. Reference monitor.

Cyber Defense Triad for Secure Systems

All three defense triad components are critical for defense of both confidentiality and integrity of information—whether the sensitive information is personally identifiable information, financial transactions (for example, credit cards), industrial control systems in the critical infrastructure, or something else that matters. Although not sufficient for perfect security, all three are practically necessary. These dimensions can be thought of as three strong “legs of a stool,” as illustrated in Figure 1.

Security for cyber systems built without a trustworthy operating system (OS) is simply a scientific impossibility. NIST has emphasized, “security dependencies in a system will form a partial ordering ... The partial ordering provides the basis for trustworthiness reasoning.”³ Proven scientific principles of the “Reference Monitor” model enable engineering a verifiably secure OS on which we can build secure cyber systems.

For a Reference Monitor implementation to work, it must ensure three fundamental properties. First, it must validate enforcement of the security policy for every reference to information. Second, it must be tamper-proof, that is, it cannot be subverted. Lastly, it must be verifiable, so we have high

assurance it always works correctly. These three fundamental properties are directly reflected in the cyber defense triad.

As illustrated in Figure 2, a Reference Monitor controls access by subjects to information in objects. A security kernel is a proven way to implement a reference monitor in a computer. Whenever a user (or program acting on behalf of a user) attempts to access information in the computer system, the Reference Monitor checks the user’s clearance against a label indicating the sensitivity of that class of data. Only authorized users are granted access.

Applying the Cyber Defense Triad

The flawed foundation of current systems is evident in the unending stream of OS security patches that are today considered part of best practices. But we can choose a better alternative. At least a half-dozen security kernel-based operating systems have been produced that ran for years (even decades) in the face of nation-state adversaries without a single reported security patch.⁷ These successes were not unexpected. As a 1983 article put it, “the security kernel approach provides controls that are effective against most internal attacks—including some that many designers never consider.”¹ That is a fundamentally different result than penetrate and patch.

These systems did not survive long after the end of the Cold War, and much of the “institutional memory” is now lost. But fortunately, some security kernel products were maintained and this original equipment manufacturer (OEM) technology is still commercially available today. And many commodity processors (for example, those that implement the Intel IA32 architecture) still include the hardware segmentation and protection rings essential to efficient security kernels. High assurance of no security patches is truly a paradigm shift. What alternative approach comes close?

Mark Heckman of the University of San Diego and I recently published a paper focused on techniques for applying those Reference Monitor properties, leveraging the fact that “associated systematic security engineering and evaluation methodology was codified as an engineering standard in the Trusted Computer System Evaluation Criteria (TCSEC)”⁴ created by NSA. However, the TCSEC didn’t include administration and acquisition mandates to actually use this knowledge to create a market in the face of entrenched vested interests. I refer interested readers to our paper for more details on the triad components summarized here.

► Mitigating software subversion.

Several cyber security professionals have concluded that subversion “is the attack of choice for the professional attacker.”² The primary means for software subversion are Trojan horses and trap doors (commonly called malware). Under the seven well-defined security classes in the TCSEC, only Class A1 systems substantially deal with the problems of subversion.

► Mandatory access control (MAC) policy.

The reference monitor is fundamentally about access control. All access control policies fall into two classes: Discretionary Access Control (DAC) and MAC. Only a label-based MAC policy can, with high assurance, enforce secure information flow. Even in the face of Trojan horses and other forms of malicious software, MAC policies can protect against unauthorized modification of information (integrity), as well as unauthorized disclosure (confidentiality).

Lipner asserts that this reference monitor approach is “not able to cope with systems large enough to be useful.”⁶ Heckman and I respond that “this quite widely-spread assertion has been repeatedly disproven by counterexamples from both real systems and research prototypes.”⁴ The paper gives numerous examples of how, by leveraging MAC, complex integrated systems can be composed from logically distinct hardware and software components that may have various degrees of security assurance or no assurance at all.

► **Verifiability.** The Reference Monitor implementation defined as a security kernel is the only proven technology for reliably achieving verifiable protection. It does not depend on unproven elegant technical solutions, such as open source for “source code inspection” or “gratuitous formal methods.”² Security kernels have been shown to be effective for systematic, repeatable, systems-oriented security evaluation of large, distributed, complex systems.

Lipner in his paper⁶ asks a critical, but largely unanswered, question: How can customers have any assurance that they are getting a secure system? His answer is limited to development process improvements that don’t address fundamentally what it means for a system to be “secure.” Heckman, by contrast, details how the Reference Monitor approach, with its strong definition of “secure system,” can answer precisely that question.⁴

What Should We Do Then?

It can be expected to take 10–15 years and tens of millions of dollars to build and evaluate a high-assurance security kernel. However, once completed, a general-purpose security kernel is highly reusable for delivering a new secure system in a couple of years. It is economical to use the same kernel in architectures for a wide variety of systems, and the TCSEC’s Ratings Maintenance Phase (RAMP) allows the kernel to be re-verified using the latest technology, without the same investment as the original evaluation. Heckman summarizes several real-world examples where, “This is demonstrated by OEM deployments of highly secure systems and products, ranging from enterprise ‘cloud technology’ to general-purpose database management sys-

tems (DBMS) to secure authenticated Internet communications, by applying commercially available security kernel technology.”⁴ Heckman additionally describes completed research prototypes in the past few years for things like source-code compatible secure Linux and a standards-compliant highly secure Network File Service (NFS).

A first necessary step is to identify where high-assurance security matters for a system. As just one example, several U.S. government leaders have expressed concern that we face an existential cyber security threat to industrial control systems (ICS) in the critical infrastructure, such as the power grid. Use of an integrity MAC security kernel can within a couple of years make our critical infrastructure dramatically more trustworthy. The U.S. government has a unique opportunity to change the cyber security game and should aggressively engage ICS manufacturers by sponsoring prototypes and providing a market using proven commercial security kernel OEM technology. Otherwise, costs may soon be measured in lives instead of bits or dollar signs. 

References

1. Ames Jr, S.R., Gasser, M., and Schell, R.R. Security kernel design and implementation: An introduction. *Computer* 16, 7 (1983), 14–22.
2. Anderson, E.A., Irvine, C.E., and Schell, R.R. Subversion as a threat in information warfare. *J. Inf. Warfare* 3 (2004), 51–64.
3. Clark, P., Irvine, C. and Nguyen, T. Design Principles for Security. NIST Special Publication 800-160, September 2016, pp. 207-221; http://csrc.nist.gov/publications/drafts/800-160/sp800_160_final-draft.pdf
4. Heckman, M.R. and Schell, R.R. Using proven reference monitor patterns for security evaluation. *Information* 7, 2 (Apr. 2016); <http://dx.doi.org/10.3390/info7020023>
5. Higgins, K.J. IBM: The security business 'has no future'. *Information Week Dark Reading*, (4/10/2008); <http://www.darkreading.com/ibm-the-security-business-has-no-future/d/d-id/1129423>
6. Lipner, S.B. Security assurance. *Commun. ACM* 58, 11 (Nov. 2015), 24–26.
7. Schell, R.R. A University Education Cyber Security Paradigm Shift. Presented at the National Initiative for Cybersecurity Education (NICE), (San Diego, CA, Nov. 2015); <https://www.fbcinc.com/e/nice/ncec/presentations/2015/Schell.pdf>
8. Schell, R.R., Downey, P.J. and Popek, G.J. Preliminary Notes on the Design of Secure Military Computer Systems. ESD, Air Force Systems Command, Hanscom AFB, MA. [MCI-73-1], Jan 1973; <http://csrc.nist.gov/publications/history/sche73.pdf>

Roger R. Schell (schellr@ieee.org) is president of Aesec Corporation, and is currently a Distinguished Fellow at the University of San Diego Center for Cyber Security Engineering and Technology. Previously he was a Professor of Engineering Practice at University of Southern California.

The author wishes to thank Michael J. Culver, Mark R. Heckman, and Edwards E. Reed for their valuable feedback on an early draft of this Viewpoint.

Copyright held by author.

Calendar of Events

November 2–4

VRST ‘16: 22nd ACM Symposium on Virtual Reality Software and Technology
Garching bei München, Germany,
Co-Sponsored: ACM/SIG,
Contact: Gudrun J. Klinker,
Email: klinker@in.tum.de

November 6–9

ISS ‘16: Interactive Surfaces and Spaces Surfaces
Niagara Falls, ON, Canada,
Sponsored: ACM/SIG,
Contact: Mark Hancock,
Email: mshancock@gmail.com

November 6–9

SIGUCCS ‘16: ACM SIGUCCS Annual Conference
Denver, CO,
Sponsored: ACM/SIG,
Contact: Laurie J. Fox,
Email: fox@geneseo.edu

November 7–10

ICCAD ‘16: IEEE/ACM International Conference on Computer-Aided Design
Austin, TX,
Co-Sponsored: Other Societies,
Contact: Frank Liu,
Email: frankliu@us.ibm.com

November 12–16

ICMI ‘16: International Conference on Multimodal Interaction
Tokyo, Japan,
Sponsored: ACM/SIG,
Contact: Yukiko Nakano,
Email: y.nakano@st.seikei.ac.jp

November 13–16

GROUP ‘16: 2016 ACM Conference on Supporting Groupwork
Sanibel Island, FL,
Sponsored: ACM/SIG,
Contact: Stephan Lukosch,
Email: S.G.Lukosch@tudelft.nl



DOI:10.1145/3000608

Pamela Samuelson

Legally Speaking

Fair Use Prevails in *Oracle v. Google*

Two software giants continue with legal sparring after an initial judicial decision.

ORACLE AND GOOGLE have been battling in the courts for more than six years about whether Google infringed Oracle copyrights by using 37 packages of the Java application program interface (API) in developing the Android platform for smartphones. In May 2016, a jury rejected Oracle's copyright claim and decided that Google's use of these 37 packages was fair and non-infringing. Oracle's lawyers have announced that the company plans to appeal. This column will explain why Oracle's appeal is unlikely to succeed and why that's good news for Java programmers, for the software industry, and for the public.

But before getting to that, this column will relate some facts about the litigation, about fair use as a defense to copyright infringement, and about Oracle and Google's arguments about the fair-use defense.

Background About *Oracle v. Google*

Oracle acquired Sun Microsystems in 2010. Sun's assets included intellectual property rights in Java technologies. Before that acquisition, Google negotiated with Sun about a possible license to use Java technologies in Android. Although those negotiations broke down, Google went ahead with using certain packages of the Java API, and in particular, the declarations that invoke implementing code for specific functions and the structure, sequence and organization (SSO) of classes



within each package, without a license from Sun. It also developed more than 100 new packages in Java and C++ for smartphone functions. Soon after acquiring Sun, Oracle sued Google claiming that Android infringed Oracle's patents and copyrights. In an earlier trial, a jury decided against the patent claims.

Initially, Google's main defense to Oracle's copyright claim was not fair use. Instead, Google asserted that the Java API packages, classes, and declarations it used in Android were not protectable by copyright law because they were too functional as components of the Java API system. An alternative for-

mulation was that any expressiveness the Java API elements Google used in Android had "merged" with the API functionality and so should not be a basis for copyright liability. Because Judge Alsup found Google's copyrightability defense persuasive, there was no need to reach Google's backup fair-use defense.

Oracle appealed that ruling and convinced the Court of Appeals for the Federal Circuit (CAFC) that the Java API elements incorporated into Android—principally, the 7,000 declarations that Google had literally copied in Android source code—were protectable expression under U.S. copyright law. (My No-

vember 2012 *Communications* Legally Speaking column incorrectly predicted that the CAFC would affirm; my March 2015 column criticized the CAFC decision and incorrectly predicted that the Supreme Court would take Google's appeal. Oh well.) The CAFC sent the case back for trial on the fair-use issue.

What Is Fair Use?

Fair use is a statutorily recognized defense to a claim of copyright infringement in the U.S. When this defense is successful, the defendant will be vindicated and no copyright liability will be found. (Most nations do not have fair-use provisions in their copyright laws.) The copyright statute says that four factors should be considered:

Purpose of Use. The purpose and character of the defendant's use of the plaintiff's work is the first factor to consider. This includes subfactors such as whether the use was commercial or noncommercial. Also significant is whether the use was "transformative." That is, did the defendant's use enable the creation of a new work that builds upon the plaintiff's work, giving it a different purpose, meaning, or message, as a parody might do? Non-transformative uses consume the work for its original purpose, as a photocopy might do. Transformative uses are more likely to be fair uses than non-transformative ones.

Nature of Plaintiff's Work. The nature of the copyrighted work is a second fair-use factor. If the plaintiff's work is highly creative, entertaining, fanciful, or artistic, fair use is likely to be narrow. If the plaintiff's work is functional or factual, fair use will tend to be broader.

Amount Taken. The substantiality of the defendant's taking of expression from the plaintiff's work is often measured quantitatively, that is, in terms of what proportion of expression from the plaintiff's work the defendant appropriated. But qualitative assessments of substantiality are sometimes made, especially if the defendant copied the "heart" of the work. When the defendant's use is transformative, however, the question becomes whether what the defendant took was reasonable in light of its transformative purpose.

Harm to Market. The effect of the defendant's use of expression from the

plaintiff's work on the market for or value of the plaintiff's work is also important. When the defendant's use is transformative, the focus is on whether the defendant's work would serve as a substitute for the plaintiff's work, not whether the plaintiff wants the defendant to pay a license fee.

Fair-Use Arguments in *Oracle v. Google*

Oracle and Google had starkly different views about whether the reuse of 37 Java API packages could be fair use. Oracle emphasized that Google acted in bad faith because some internal email correspondence showed that some Google employees thought Google needed to a license to use the Java API. Oracle also contended that Google had made non-transformative use of the Java API packages because it copied the declarations verbatim. Its purpose, moreover, was commercial. Oracle argued all three considerations weighed against fair use.

As for the other fair-use factors, Oracle insisted that the Java API was highly creative. Google's appropriation was substantial because the Android source code included 11,500 lines of declaring code that Google copied from the Java API. Oracle's economic expert witness testified that Google's reuse of the Java API packages had caused substantial harm to the market for the Java API because Oracle had been unable to collect licensing revenues from Google and from others as well. Oracle also contended that the network effects arising from the success of the Android platform had made it impossible for Oracle to make a successful entrance into the smartphone market.

Google's lawyer urged the jury to find

As for the other fair-use factors, Oracle insisted that the Java API was highly creative.

that Google had made transformative uses of the Java API packages by building them into the Android platform. He compared the Java API to a file cabinet with folders, a functional device that was far from the core of copyright. He argued that Google took no more from the Java API than was necessary to achieve its transformative purpose. The transformative nature of Google's use mitigated the harm factor because Android did not supplant demand for the Java API for its original purpose. Fair use is intended to promote ongoing innovation, which Google's lawyer argued Android had done.

Oracle's Effort to Overturn the Jury Verdict

Oracle v. Google was tried to a jury because the two software giants did not agree about key facts pertinent to the resolution of their dispute about Android. The role of a jury is to decide which litigant's view of the facts is most persuasive, and then to apply the law to those facts in accordance with the instructions the judge reads to jurors after the trial testimony has ended and the lawyers have made their closing arguments. Juries generally come back with a verdict for one party or the other. They do not have to explain their findings or the reasons for their verdict.

After the jury found in favor of Google's fair-use defense, Oracle's lawyers filed a motion asking Judge Alsup to set aside the jury verdict and rule in its favor as a matter of law. The judge denied that motion and wrote an opinion to explain why a reasonable jury might have found in Google's favor on several key fact issues:

For one thing, Oracle made much during the trial about Google acting in bad faith in using the Java API declarations when it should have gotten a license. Counteracting that evidence was testimony by some witnesses that Google's reimplementations of interfaces was customary in the software industry. A reasonable jury, Judge Alsup concluded, could have concluded that Google's good faith defense was more persuasive than Oracle's bad faith accusation.

Second, Oracle insisted that the Java language, which all were free to use without permission, was distinct from the Java API declarations. Google



Association for
Computing Machinery

ACM Conference Proceedings Now Available via Print-on-Demand!

*Did you know that you can
now order many popular
ACM conference proceedings
via print-on-demand?*

Institutions, libraries and individuals can choose from more than 100 titles on a continually updated list through Amazon, Barnes & Noble, Baker & Taylor, Ingram and NACSCORP: CHI, KDD, Multimedia, SIGIR, SIGCOMM, SIGCSE, SIGMOD/PODS, and many more.

**For available titles and
ordering info, visit:
librarians.acm.org/pod**



The jury verdict in Google's favor was obviously good news for Larry Page and the shareholders of Google.

contended that the Java language and the API were inextricably intertwined. A reasonable jury could have been persuaded that there was less separability than Oracle contended. The jury could also have decided that it would be unreasonable to require Java programmers to have to learn and keep straight two dialects of Java-based APIs, one for Android and one for other Java platforms, which is what an Oracle victory would have meant.

Third, Judge Alsup thought that the jury could have reasonably found that Google's reuse of the Java declarations was transformative because Google reimplemented the interfaces in independently written code and developed new API packages to enable smartphone functionalities that were quite different from the computing platforms for which the Java API was originally developed.

Fourth, the jury could reasonably have found that the Java declarations, while creative enough to be copyrightable, were predominantly functional.

Fifth, the jury could have believed that Google "duplicated the bare minimum of the 37 Java API packages" that was reasonably necessary to achieving its transformative purpose.

Sixth, the jury could have reasonably found that Oracle suffered no harm from Google's use of the 37 API packages because the Java API was developed for a different computing environment. Sun's effort to promote a version of Java for mobile devices was unsuccessful before Android entered the market. And even before Android was released, Sun had opened the Java API for free use under an open source license.

Why Do These Facts Matter?

By emphasizing the disagreements over these facts, Judge Alsup not only provided ample reasons for denying Oracle's post-trial motion. He also set the stage for narrowing what the CAFC can (or should) do with an Oracle appeal. Appellate courts are supposed to defer to jury findings and to construe evidence in the record as consistent with the jury's verdict.

The CAFC is already on record that Google's fair-use defense raised a triable issue of fact. So the best that Oracle can realistically hope for on appeal is that its attack on Judge Alsup's jury instructions will prevail. Yet, this would just mean the case would go back to Judge Alsup for another trial. Not even Oracle's lawyers could look forward to that.

Conclusion

The jury verdict in Google's favor was obviously good news for Larry Page and the shareholders of Google. Yet, it was also good news for programmers who have been using those 37 Java API packages when developing apps for the Android platform. Judge Alsup recognized that if Java programmers had to "master and keep straight two different [Java] SSOs as they switched between the two systems for different projects," as a verdict for Oracle would have required, this would have "fomented confusion and error to the detriment of both Java-based systems and to the detriment of Java programmers at large."

Google's victory is also good news for competition in the software industry, at least in the U.S., because fair use now a meaningful defense for those who reimplement other companies' APIs in independent code. While most cases have struck down copyright claims in interfaces necessary for interoperability on lack of copyrightability grounds, fair use is now a proven alternative path to defense victories. The public benefits from Google's victory because of the ongoing competition and innovation that reuse of APIs has brought and will bring. 

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

Copyright held by author.

Economic and Business Dimensions Visualization to Understand Ecosystems

Mapping relationships between stakeholders in an ecosystem to increase understanding and make better-informed strategic decisions.

COMPANIES CAN NO longer rely on just their internal competencies. They must complement their own competencies with those of other firms through alliances, partnerships, and digital relationships. Google, by opening its mobile Android platform to manufacturers and developers, has become the leading apps provider in the mobile space. These multifaceted interdependencies create complex connections, or ecosystems, that affect and are affected by multiple stakeholders such as suppliers, distributors, outsourcing firms, makers of related products or services, and technology providers.³ Competition today is between ecosystems. Understanding ecosystems can then help managers improve strategic decisions and reshape the boundaries of their industries. Interpreting this network of interactions, however, can be difficult. For example, a nascent ecosystem like the Internet of Things (IoT) has no dominant player and is increasing in scale, scope, and complexity.⁴ Under these circumstances, companies can have difficulty determining the most effective business opportunities.

We describe a graph theoretic visualization method to help companies map relationships between ecosystem stakeholders. To visualize ecosystems



effectively, many entities, activities, and tools are required. Capturing, generating, and interpreting ecosystems requires a sponsor who wants to learn more about a company, a technology, or an industry, data scientists to collect salient data and prepare it for analysis, visualization tools, and a domain expert

to interpret the visuals and make recommendations. Finally, companies must manage the data, visuals tools, and lessons learned in order to be effective.

A Visual Approach

Decision makers that want to visualize ecosystems need a structured

Figure 1. An Internet of Things (IoT) stack.

Application Layer for Sense Making	for example, Analytics
System Integrators	for example, Accenture, Cognizant
Infrastructure	for example, Azure, Amazon Web Services, Telstra
Platforms for Device Connectivity	for example, Thingworkx, Xively, Iotivity
Sensory Layer	for example, ARM, Intel, Sigfox

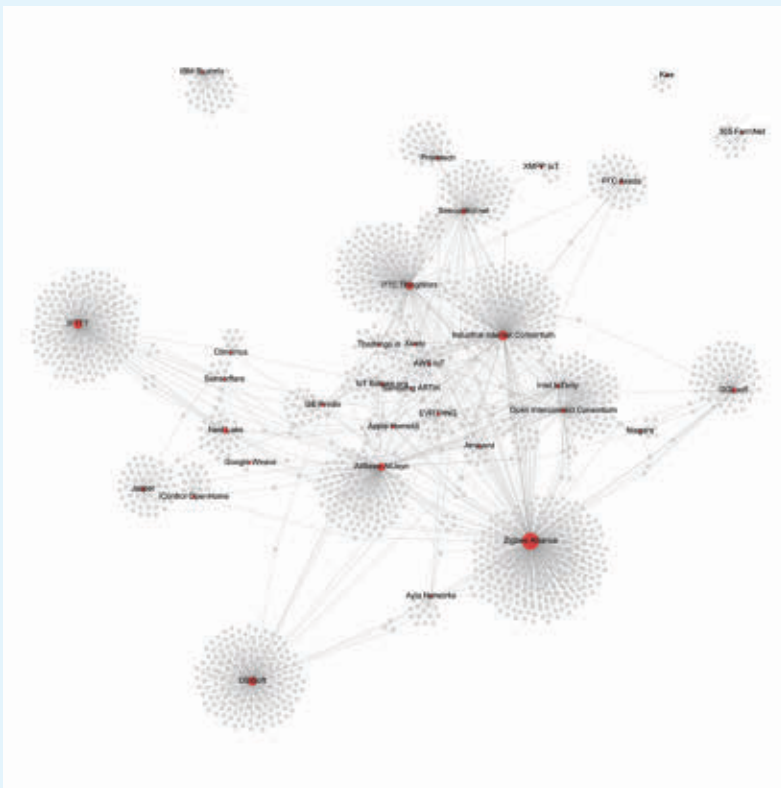
approach and several visual representation methods can be used.¹ The methodology consists of steps discussed here. This approach is not linear; there are many feedback loops between the stages. Progress is guided by human decision making along the way. We use The Internet of Things (IoT) as an illustration of this methodology.

Determine industry structure: Think of an industry as a company, its complementors and competitors. Identify the value chain or stacks of activities that deliver something of benefit to customers. These are inferred from industry publications and company websites. Industry structure helps the analyst identify the companies that are the major platform companies in an

industry. For the IoT industry, the stack shown in Figure 1 is an illustration based on current industry consensus. Refinements to the stack can be made as new companies emerge. An independent list of available platforms can be collected from trade publications and mapped onto the stack.

Identify companies and their attributes: Platforms are important to technology-centric industries.⁵ Identifying platform companies for the ecosystem can be done by searching articles in industry publications. Use Internet search engines, news portals, socially curated news websites, or social media sources to locate these articles. Industry publications complemented with discussions with practitioners identified 34

Figure 2. The IoT ecosystem.



platform companies for the IoT ecosystem. Lists can be augmented by semi-open sources such as Crunchbase or Angel.co or paid services such as Dun & Bradstreet, DataFox, and CB Insights. Once finalized, the analyst can visit each company's website to identify listed partners that provide services or components for platforms and document dependencies. Most company websites list strategic alliances or partners, and describe the types of relationships (for example, technical, marketing, licensing, and so forth) and content (for example, date formed, nature, and other characteristics).

Finalize semantics for nodes and dependencies: Based on the data collected so far, the analyst can prepare it for visualization. Some software requires explicit specification of the visual encodings of nodes and edges (including the attributes that drive color, shape, size, and dependencies), and particular consideration to data type (that is, quantitative, ordinal, categorical). More recent software packages (for example, Tableau, Gephi, ecovis2) allow dynamic selection of attributes and assignments. Node sizes are usually based on a company's revenue or number of employees. Dependencies are often color coded, based on relationship.

Visualize, analyze, and interpret: The visualizations in the figures in this column use Gephi, an open source graph visualization tool.² Many visualization packages offer different network layout algorithms. Most commonly used are derivations of force-directed layouts, in which prominent nodes are drawn in the center and less prominent ones are pushed to the periphery. Nodes close to each other have stronger associations. After visualization, the sponsoring organization can be asked for feedback to determine if they have any insights about companies in key network positions or clusters of interest, find any surprises, and identify companies that did not make the list. Corrections are incorporated in subsequent analysis. This iterative process creates confidence in the visualizations.

Interpreting the Visualization

As noted, the IoT is a complex ecosystem not yet dominated by any single

player. In Figure 2, platform companies are the red-colored circles and component providers are the gray-colored circles. Dependencies between platforms and components are shown as links. This visualization can be used to identify key players and determine business opportunities.⁴

Alliance Strength. Density of connections, number of partners, and network centrality all provide a sense of the resources available to any given firm that serves as an axis. In Figure 1, it's easy to see, for example, that the Zigbee Alliance and Industrial Internet Consortium each represent an important nexus of activity. If one is a small player looking to join a winning alliance then choosing from among the bigger players could be wise. While, if one is a competing nexus, choosing compatibility with other midsize players could reshape the network landscape.

Vertical Integration. Companies that provide end-to-end solutions become vertically integrated and tightly controlled. For example, in its early days, the software industry was dominated by IBM, Digital Equipment, and International Computers. These end-to-end solutions providers fragmented into multiple companies and divided technical leadership. The IoT ecosystem (see Figure 3) reveals many clusters, suggesting they are either providing end-to-end solutions or leaving integration to the user. IBM among those three early pioneers survives today, but Bluemix (integrating IBM products and services), Kaa (providing middleware services that work with any stack), and 365 Farmnet (specialized for agricultural services) might be relevant for individual specializations and doomed from a network perspective.

Preferential Attachment. With proliferation of platforms in the IoT ecosystem, solutions providers must select one platform over another. If they simultaneously commit to too many platforms they make significant resource commitments. In the software industry, solutions providers attach themselves to one or two platforms based on incentives, momentum or technological superiority. Here solutions providers are unsure, they can

Figure 3. Vertical differentiation in the IoT ecosystem.

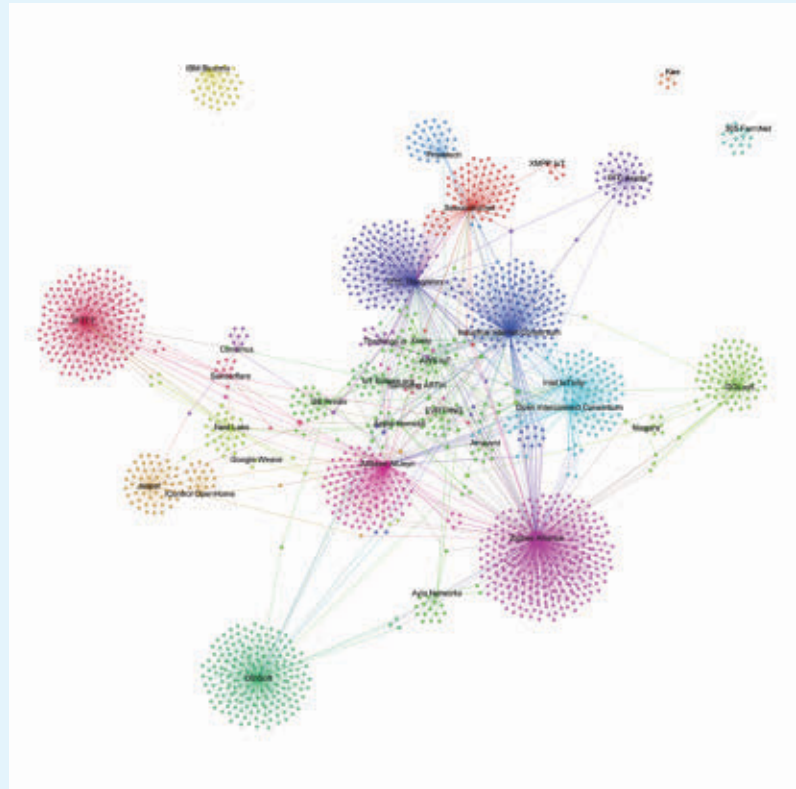


Figure 4. Preference of Android to IFTTT.

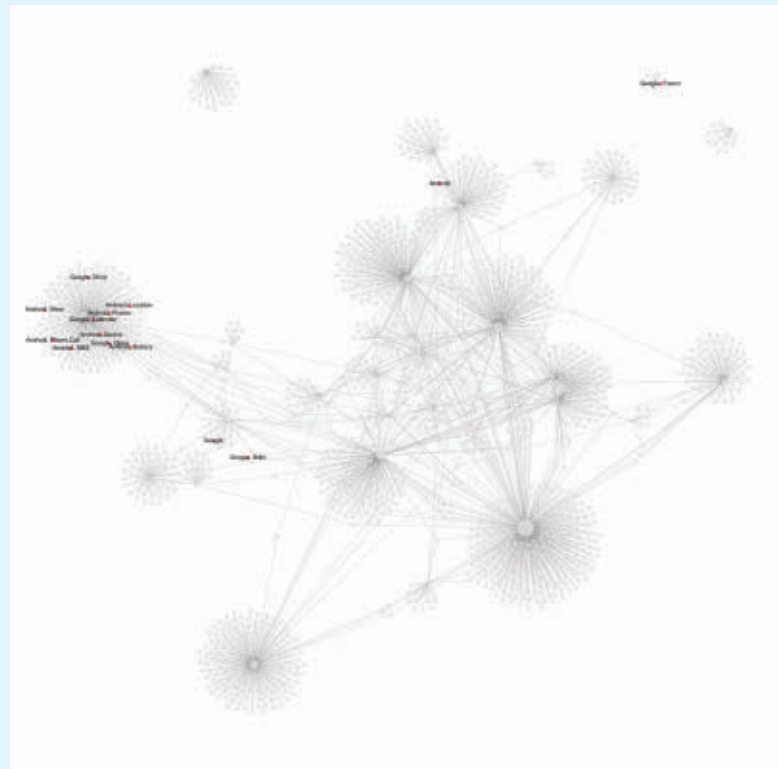
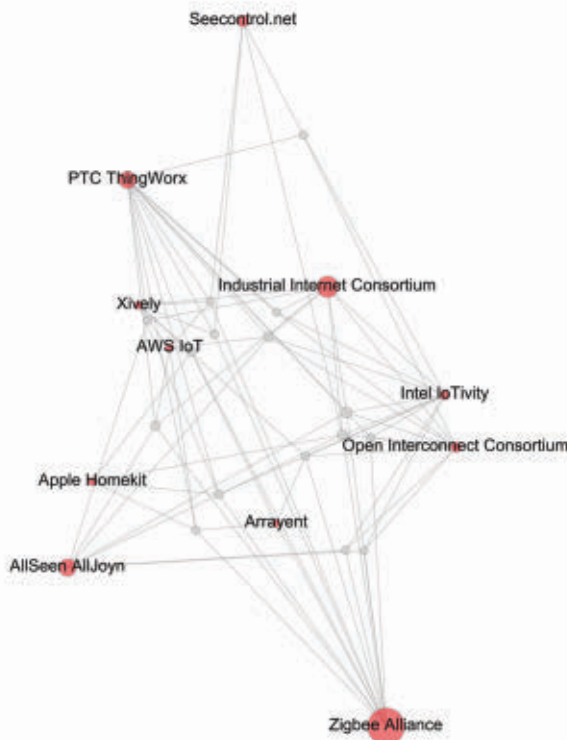


Figure 5. Core of the IoT ecosystem.



hedge their bets by selecting multiple platform. Oracle built huge market share by supporting multiple platforms like Windows and Linux. Android (see Figure 4) is preferential to IFTTT, ensuring that Android can integrate with most other products.

New Entrant and Network Effects. Major players in non-IoT industries (for example, Samsung, Apple, and Alphabet) still fight for dominance in the IoT ecosystem. To gain developers, these companies open up their IoT platforms and programming interfaces. Brand recognition, existing devices, platforms, and relationships with developers might give them an advantage. As Alphabet expands into home automation with Nest, many Google Play developers might move with it. Developer familiarity with Google's API in one setting could also be used to develop products in other settings. A similar trend in the software industry enabled Microsoft's strong position in operating systems to help it dominate the web browser market. In Figure 3, Apple HomeKit occupies a key position in the network. With Apple's

decision to open its API, it instantly brings several hundreds of thousands of developers into the equation.

Focus on the Core. An emergent ecosystem builds infrastructure first (see the core of the ecosystem in Figure 5). Personal computers needed an operating system before the explosion of applications could occur. The IoT ecosystem needs constant monitoring. Communicating data across networks is critical. Low-power solutions and connectivity are needed in the infrastructure (see the figure showing the core of the current IoT ecosystem) to help connect devices and ensure efficiency.

The Power of Interoperability Emergent ecosystems require integration of devices and services. They also require interoperability between competing platforms. These significant challenges benefit from learning about the early days in the software industry. Application Program Interfaces (APIs) allowed firms to interact and share information with other firms, and made it easy to achieve integration and interoperability across platforms and devices. They could be used for more: APIs could

track the ways users integrate services and devices, providing insights into how users derive value through a product's use. Alphabet's Nest has used this very effectively to implement the "Works With Nest" program. Because of this program, customers buying Nest will have the option to integrate it with the most common devices like LG washing machines or Amazon's Echo. Nest's position can be further solidified, if it became interoperable with highly connected platforms (as shown in Figure 3). This position gives it access to critical knowledge flows across the network.

Challenges

Visualizations can help with understanding industry structure and emergence of key players. The dynamic nature of competition makes every new entrant and every move by an existing player relevant to the topology the ecosystem changes and the competitive position of companies. Companies must constantly track nodes and dependencies in networks. Computational tools for data gathering and AI techniques for text processing and understanding can be automated so executives can focus on understanding and exploring interactive visuals. Ecosystem visualization continues to grow and mature as data scientists, graph theorists, and visualization researchers create new techniques. In this way, ecosystem visualization is likely to become ubiquitous in high-velocity business environments. 

References

1. Basole, R.C. et al. Understanding business ecosystem dynamics: A data-driven approach. *ACM Trans. Management Information Systems (TMIS)* 6, 2 (Feb. 2015), 6.
2. Bastian, M., Heymann, S., and Jacomy, M. Gephi: An open source software for exploring and manipulating networks. *ICWSM 8*, (2009), 361–362.
3. Tansiti, M. and Levien, R. *The Keystone Advantage: What the New Dynamics of Business Ecosystems Mean for Strategy, Innovation, and Sustainability*. Harvard Business Press, 2004.
4. Iyer, B. To predict the trajectory of the Internet of Things, look to the software industry. *Harvard Business Review* (Feb. 25, 2016).
5. Parker, G.G., Van Alstyne, M.W., and Choudary, S.P. *Platform Revolution*. Norton Publishing, New York, 2016.

Bala R. Iyer (biyer@babson.edu) is Professor and Chair TOIM Division, Babson College, MA. Twitter: @BalaIyer

Rahul C. Basole (basole@gatech.edu) is Associate Professor, School of Interactive Computing, and Director, Tennenbaum Institute, Georgia Institute of Technology, GA. Twitter: @basole

Copyright held by authors.

Education

Growing Computer Science Education Into a STEM Education Discipline

Seeking to make computing education as available as mathematics or science education.

COMPUTING EDUCATION IS changing. At this year's CRA Snowbird Conference, there was a plenary talk and three breakout sessions dedicated to CS education and enrollments. In one of the breakout sessions, Tracy Camp showed that much of the growth in CS classes is coming from non-CS majors, who have different goals and needs for computing education than CS majors.^a U.S. President Obama in January 2016 announced the CS for All initiative with a goal of making computing education available to all students.^b

Last year, the U.S. Congress passed the STEM Education Act of 2015, which officially made computer science part of STEM (science, technology, engineering, and mathematics). The federal government offers incentives to grow participation in STEM, such as scholarships to STEM students and to prepare STEM teachers. Declaring CS part of STEM is an important step toward making computing education as available as mathematics or science education.

The declaration is just a first step. Mathematics and science classes are common in schools today. Grow-



High school students and teachers engaging in collaborative meetings about computer science represents an important step toward making computing education as available as science or mathematics education.

ing computing education so it is just as common requires recognition that education in computer science is different in important ways from education in STEM. We have to learn to manage those differences.

Computer Science Is Invisible, Formally and Informally

Students enter mathematics or science classes at the post-secondary level with years of knowledge and experience be-

hind them. Informally, many students talk to their parents about science issues ("Why is the sky blue?"), think about the numbers in their lives ("How much is the tax?"), visit science museums, and see media about mathematics and science. Students live in a world where living, chemical, and physical behavior is explained by biology, chemistry, and physics. They develop ideas about how the world works, some of which are wrong (like simple Lamarckian evolu-

a <http://cra.org/wp-content/uploads/2016/07/BoomCamp.pdf>

b <https://www.whitehouse.gov/blog/2016/01/30/computer-science-all>

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

tion). Students start formal learning about mathematics and science in the earliest grades. Mathematics and science classes can help answer their questions and improve the theories that a reflective child has about the world.

While computing is ubiquitous in the developed world, and cellphones and other handheld computing devices are increasingly common in the developing world, few students get the opportunity to look under the covers of those devices to reflect on questions about computing. Maybe children might ask, “Why is my browser so slow?” The concepts that computer science explains are mostly invisible to children, such as digital representations, algorithms, and networks. If they don’t see the computation in their world, it’s difficult to reflect on and develop questions about it. We have less museum or media coverage than the rest of STEM. Few students enter secondary or post-secondary computer science classes with any previous formal education in computing.

The difference means it is difficult for a teacher to set expectations. Some students do have experience coming in to class; most do not. When does a student need remedial help? We are in a transitional stage where the gap between the haves and have-nots in computing education is large.

There is a positive side to the invisibility of computation in children’s daily life. Mathematics and science students develop their misunderstandings of the world from daily life, too. Students’ incorrect science theories are wrong, but mostly work. The world is not flat, but it seems so, and you can live much of your life believing the world is flat. On the other hand, students develop incorrect theories in computer science only in computer science classes and mostly because of how we taught them. If we change the way we teach, we can reduce misunderstandings.

Developing Learning Progressions

We have been doing mathematics and science education for a long time. We have a good idea of what students can do in science and mathematics from early ages, and how quickly they can develop new concepts and skills. Educators call this a learning progression.

Since computing education is younger than science and mathemat-

ics education, we do not have learning progressions. Computer science educators mostly have had the goal of preparing future professional programmers, but goals change when we talk about teaching everyone. Not every introductory biology student will become a biologist and not every intro physics student will become a physicist. We need to determine what pieces of computing the educated citizen needs to know. Then we can plan how and in what order we teach those pieces.

What can we expect a first-year undergraduate to learn in a single term CS class if they have no previous computing experience? What can we expect to be able to teach any eight-year-old or a 12-year-old about computer science? What are the challenges to teaching computing successfully to students with cognitive disabilities, or who are blind, or who have inadequate mathematics preparation?

We first realized in the early 1980s that we often overestimate what first-time CS students can do.⁵ The McCracken Study² showed that problems that introductory computer science instructors find reasonable are not solvable by a majority of introductory students.

Part of the trick is simply learning how to measure what students have learned. Briana Morrison and Lauren Margulieux showed⁴ that textual programming is cognitively complex. We want students to learn the concepts and skills of programming, but programming itself is a complex activity—asking students to program requires students to be able to do *everything*. Morrison and Margulieux showed they can measure learning toward programming using Parsons Problems.³ A Parsons Problem asks students to solve a programming problem, then gives them all the lines of code that solve that problem, on tiles or “refrigerator magnets.” They showed they could tease out differences in students’ understanding of programming, even when the students could not successfully program the solutions yet. Being able to measure the development of these skills, without requiring students to master the skills, is critical to developing learning progressions.

If we want to teach computer science at younger ages, we have to figure out how to reduce that complexity. We

While computer science is now part of STEM in the U.S. by fiat, students cannot access computer science classes as easily as mathematics and science education.

want students to be able to build programs they find motivating and engaging, without mastering all the skills of textual programming first. David Weintrop and Uri Wilensky have shown that students using blocks-based languages (like Scratch, Snap, and Blockly) make far fewer errors than in textual programming languages. Again, part of their achievement is in measurement. They developed a commutative assessment⁶ that allowed them to compare the same concepts in both textual and blocks-based programming languages. We will need many kinds of measures to develop learning progressions for computer science.

Computer Science Is Valued but Misunderstood

Students enter undergraduate mathematics and science classes after years of formal education, so they enter with a good idea about what those fields mean. That's not true for computer science. Even undergraduate CS majors do not know what computer science is. Mike Hewner showed that even undergraduate students who declare a major in computer science only have an unclear idea of what the field is about.¹

Large-scale surveys in collaboration between Google and Gallup have shown that parents and principals think courses in computer science are about how to use personal computers.^c The surveys show parents and principals highly value computing,

and want more computing education for their children. But the parents and principals mostly do not understand what it is.

Students want computer science, whatever they think it is. Many of them want it because of the economic value of knowledge of computer science. They don't know what it is, but they know it can get them a good job and make them more effective at the job they want.

The CS situation is different from science or mathematics. Contrast the number of coding boot camps available in your area to the number of biology boot camps or algebra boot camps. While having a demand for CS is mostly positive, it creates a strange dynamic in the computer science class. Students demand the "real thing" (which we might interpret as "what will help me in a job"), even if they don't know what that is. For example, students might complain about learning a blocks-based language or using a pedagogical IDE because it's not "real"—even if they are not quite clear what "real" is.

Building the Infrastructure for CS Classes

In many countries and U.S. states, you can learn the number of students taking primary or secondary school reading, mathematics, or science classes. In the U.S., hardly any state can tell you the number of students in their computer science classes at any level, or what is being taught in those courses. (In many states, "computer science" and "computing applications" courses are considered the same.) Because computer science has only recently been declared part of STEM, it has not been tracked like other STEM subjects. We don't know with certainty how much computing education is offered in the U.S. today nor where it is offered, which makes it difficult to plan and grow access to computing education.

We believe (from looking at data about Advanced Placement CS and in those states that do track) that far less than 30% of secondary school students even have the opportunity take a computer science course in the U.S. today, and less than 10% of primary school students. To reach the ubiquity of access to mathematics and science edu-

cation classes, we need to increase the number of computer science classes and teachers by a magnitude. That is an enormous change with dramatic implications. What we have today may not tell us much about tomorrow. The preparation, abilities, and preferences of existing computer science teachers may not be predictive when we have a 10-times-larger population of teachers. We have to invent whole new teacher education programs.

Steps Toward Pervasive Computing Education

While computer science is now part of STEM in the U.S. by fiat, students cannot access computer science classes as easily as mathematics and science education. Many countries are ramping up computing education, so the situation is going to change. As it does, we will have to develop more accurate expectations of how students learn CS, improve our ability to measure learning in computing, develop learning progressions, and create an infrastructure to develop teachers and track progress as we reach the pervasiveness of mathematics and science education. 

References

1. Hewner, M. Undergraduate conceptions of the field of computer science. In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research (ICER '13)*. ACM, New York, 2013, 107–114.
2. McCracken, M. et al. A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *ACM SIGCSE Bulletin* 33, 4 (2001), 125–140.
3. Morrison, B.B., Margulieux, L.E., and Guzdial, M. Subgoals, context, and worked examples in learning computing problem solving. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research (Omaha, Neb., 2015)*, 21–29.
4. Morrison, B.B., Margulieux, L.E., Ericson, B., and Guzdial, M. Subgoals help students solve Parsons problems. Paper presented at the *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (Memphis, Tenn., 2016).
5. Soloway, E. Learning to program = learning to construct mechanisms and explanations. *Commun. ACM* 29, 9 (Sept. 1986), 850–858.
6. Weintrop, D. and Wilensky, U. 2015. Using commutative assessments to compare conceptual understanding in blocks-based and text-based programs. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research (ICER '15)*. ACM, New York, NY, 2015, 101–110; DOI: <http://dx.doi.org/10.1145/2787622.2787721>

Mark Guzdial (guzdial@cc.gatech.edu) is a professor at the Georgia Institute of Technology.

Briana Morrison (bbmorrison@unomaha.edu) is an assistant professor of computer science at the University of Nebraska Omaha.

Copyright held by author.

c <https://services.google.com/fh/files/misc/images-of-computer-science-report.pdf>

Viewpoint

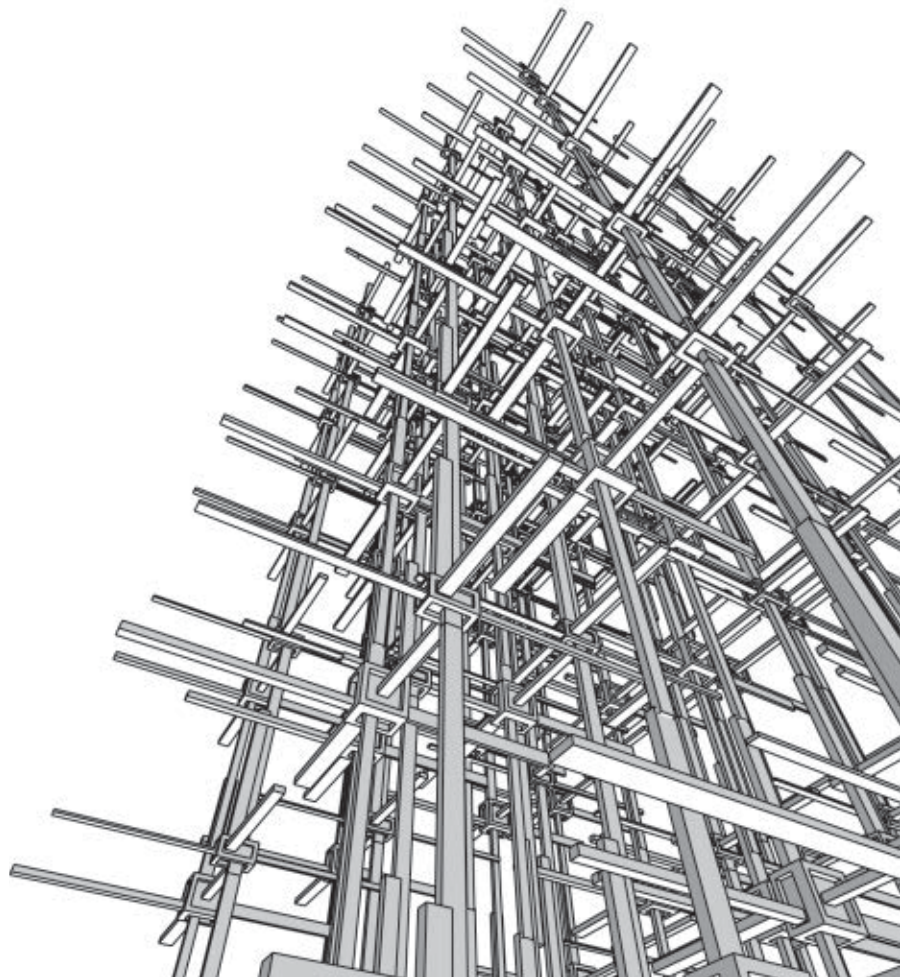
Time to Reinspect the Foundations?

Questioning if computer science is outgrowing its traditional foundations.

THE THEORY OF computability was launched in the 1930s, by a group of logicians who proposed new characterizations of the ancient idea of an algorithmic process. The most prominent of these iconoclasts were Kurt Gödel, Alonzo Church, and Alan Turing. The theoretical and philosophical work that they carried out in the 1930s laid the foundations for the computer revolution, and this revolution in turn fueled the fantastic expansion of scientific knowledge in the late 20th and early 21st centuries. Thanks in large part to these groundbreaking logico-mathematical investigations, unimagined number-crunching power was soon boosting all fields of scientific enquiry. (For an account of other early contributors to the emergence of the computer age see Copeland and Sommaruga.⁹)

The motivation of these three revolutionary thinkers was not to pioneer the disciplines now known as theoretical and applied computer science, although with hindsight this is indeed what they did. Nor was their objective to design electronic digital computers, although Turing did go on to do so. The founding fathers of computability would have thought of themselves as working in a most abstract field, far from practical computing. They sought to clarify and define the limits of human computability in order to resolve open questions at the foundations of mathematics.

The 1930s revolution was a critical moment in the history of science: ideas devised at that time have become cor-



nerstones of current science and technology. Since then many diverse computational paradigms have blossomed, and still others are the object of current theoretical enquiry: massively parallel and distributed computing, quantum computing, real-time interactive asynchronous computing, hypercomput-

ing, nano-computing, DNA computing, neuron-like computing, computing over the reals, and computing involving quantum random-number generators. The list goes on ... (for example, see Cooper et al.,³ and recent issues of *Natural Computing* and *International Journal of Unconventional Computing*).

Few of these forms of computation were even envisaged at the time of the 1930s analysis of computability—and yet the ideas forged then are still typically regarded as constituting the very basis of computing.

So here's the elephant in the room. Do the concepts introduced by the 1930s pioneers provide a logico-mathematical foundation for what we call *computing* today, or do we need to overhaul the foundations in order to fit the 21st century? Much work has been devoted in recent years to analysis of the foundations and theoretical bounds of computing. However, the results of this diverse work—carried out by computer scientists, mathematicians, and philosophers—do not so far form a unified and coherent picture. It is time for the reexamination of the logico-mathematical foundations of computing to move center stage. Not that we should necessarily expect an entirely unified picture to emerge from these investigations, since it is possible that there is no common thread uniting the very different styles of computation countenanced today. Perhaps investigation will conclude that nothing more than 'family resemblance' links them.

A question pressed by foundational revisionists is: How are the bounds of computability, as delineated by the 1930s pioneers—bounds that theoretical computer science has by and large simply inherited and enshrined in the textbooks—related to the bounds of *physical* computing? The famous Church-Turing thesis delineates the bounds of computability in terms of the action of a Turing machine. But could there be mathematical functions that are physically computable and yet not Turing-machine computable? Various physical processes have been proposed that allegedly allow for computation beyond Turing-machine computability, appealing to physical scenarios that involve, for example, special or general relativity^{11,14,16,17,19} (see Davis¹⁰ for a critique of the whole idea of super-Turing computation). The possibility of super-Turing computation or hypercomputation is not ruled out by the Church-Turing thesis, once the latter is understood as originally intended, namely as an analysis of the bounds of what is computable by an idealized *human* computer—a mathematical clerk who works by

The founding fathers of computability would have thought of themselves as working in a most abstract field, far from practical computing.

rote with paper and pencil.^{5-7,21-23} The limits of a human computer don't necessarily set the limits of every conceivable physical process (for example, see Pour El and Richards¹⁸), nor the limits of every conceivable machine. So it is an open question whether the Turing-machine model of computation captures the physical, or even the logical, limits of machine computability.

The Turing machine has also traditionally held a central place in complexity theory, with the so-called 'complexity-theoretic' Church-Turing² or 'extended' Church-Turing thesis¹ asserting that there is always at most a polynomial difference between the time complexity of any reasonable model of computation and that of a probabilistic Turing machine. (A deterministic version of the thesis is also known as the Cobham-Edmonds thesis.¹³) This traditional picture is today under threat, when some propose counterexamples to the extended Church-Turing thesis. Bernstein and Vazirani² gave what they called 'the first formal evidence' that quantum Turing machines violate the extended Church-Turing thesis. Shor's quantum algorithm for prime factorization is arguably another counterexample.²⁰ All this is highly controversial, but once again the signs are that we should take a systematic look at the foundations.

The most fundamental question of all is, of course: what *is* computa-

tion? Some will argue that the Turing-machine model gives an adequate answer to this question. But, given the enormous diversity in the many species of computation actually in use or under theoretical consideration, does the Turing-machine model still capture the nature of computation? For example, what about parallel asynchronous computations? Or interactive, process-based computations?^{215,24} Now that such diversity is on the table, it may be that the Turing-machine model no longer nails the essence of computation. If so, is there any sufficiently flexible and general model to replace it? Some of the authors of this Viewpoint would bet on Turing's model, while the others would not. Either way this question is central.

Turning to the mysterious computer in our skulls ... just as the limits of an idealized human clerk working by rote do not necessarily dictate the limits of computing hardware, nor do they necessarily dictate the limits of the human brain. For example, do human beings qua creative mathematicians carry out brain-based computations that transcend the limits of human beings qua mathematical clerks? The question is not merely philosophical: it seems highly likely that the brain will prove to be a rich source of models for new computing technologies. This gives us yet more reason to return to the writings of the 1930s pioneers, since both Gödel and Turing appear to have held that mathematical thinking possesses features going beyond the classical Turing-machine model of computation (see Copeland and Shagrir⁸).

In 2000, the first International Hypercomputation Workshop was held at University College, London. Now there is a growing community of hundreds of researchers in this and allied fields, communicating results at conferences (for example, 'Computability in Europe,' 'Theory and Applications of Computing,' and 'Unconventional Computing') and in journals (for example, *Applied Mathematics and Computation*, *Theoretical Computer Science*, *Computability*, and *Minds and Machines*). We hope this is only the beginning. If it was important to clarify the logico-mathematical foundations of computing in the 1930s, when computer science was

no more than a gleam in Turing's eye, how much more important it seems today, when computing technology is diversifying and mutating at an unprecedented rate. Via the great pioneers of electronic computing (such as Freddie Williams, Tom Kilburn, Harry Huskey, Jay Forrester, John von Neumann, Julian Bigelow, and of course Turing himself) the 1930s analysis of computation led to the modern computing era. Who knows where a 21st-century overhaul of that classical analysis might lead. **C**

References

- Aharonov, D. and Vazirani, U.V. Is quantum mechanics falsifiable? A computational perspective on the foundations of quantum mechanics. In B.J. Copeland, C. Posy, and O. Shagrir, Eds., *Computability: Gödel, Turing, Church and Beyond*, MIT Press, Cambridge, MA, 2013.
- Bernstein, E. and Vazirani, U.V. Quantum complexity theory. *STOC '93 Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing* (1993), 11–20.
- Cooper, B., Lowe, B. and Sorbi, A., Eds. *New Computational Paradigms: Changing Conceptions of What Is Computable*. Springer, 2008.
- Copeland, B. J. The Church-Turing Thesis. *The Stanford Encyclopedia of Philosophy*. E.N. Zalta, Ed., 2002; <http://plato.stanford.edu/entries/church-turing/>.
- Copeland, B.J. Narrow versus wide mechanism. *The Journal of Philosophy* 97, (2000), 5–32.
- Copeland, B.J. Hypercomputation: Philosophical issues. *Theoretical Computer Science* 317 (2004), 251–267.
- Copeland, B.J. and Proudfoot, D. Alan Turing's forgotten ideas in computer science. *Scientific American* 280, (1999), 76–81.
- Copeland, B.J. and Shagrir, O. Turing versus Gödel on computability and the mind. In B.J. Copeland, C. Posy, and O. Shagrir, Eds., *Computability: Gödel, Turing, Church and Beyond*, MIT Press, Cambridge, MA, 2013.
- Copeland, B.J. and Sommaruga, G. The stored-program universal computer: Did Zuse anticipate Turing and von Neumann? In G. Sommaruga and T. Strahm, Eds., *Turing's Revolution*, Birkhäuser, Basel, 2006.
- Davis, M. The myth of hypercomputation. In C. Teuscher, Ed., *Alan Turing: Life and Legacy of a Great Thinker*. Springer Verlag, Berlin, 2004, 195–212.
- Etesi, G. and Németi, I. Non-Turing computations via Malament-Hogarth space-times. *International Journal of Theoretical Physics* 41, (2002), 341–370.
- Gandy, R. Church's thesis and principles for mechanisms. In J. Barwise, D. Kaplan, H.J. Keisler, P. Suppes, and A.S. Troelstra, Eds., *The Kleene Symposium*, North-Holland, Amsterdam, 1980.
- Goldreich, O. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.
- Hogarth, M.L. Does general relativity allow an observer to view an eternity in a finite time? *Foundations of Physics Letters* 5, (1992), 173–181.
- Milner, R. Elements of interaction: Turing Award lecture. *Commun. ACM* 36, 1 (Jan. 1993), 78–89.
- Németi, I. and Andréka, H. Can general relativistic computers break the Turing barrier? In A. Beckmann, U. Berger, B. Löwe, and J.V. Tucker, Eds., *Logical Approaches to Computational Barriers*, Springer-Verlag, Berlin, 2006, 398–412.
- Pitowsky, I. The physical Church thesis and physical computational complexity. *Iyyun* 39 (1990), 81–99.
- Pour-El, M. and Richards, I. The wave equation with computable initial data such that its unique solution is not computable. *Advances in Mathematics* 39, (1981), 215–239.
- Shagrir, O. and Pitowsky, I. Physical hypercomputation and the Church-Turing Thesis. Special issue on hypercomputation. *Minds and Machines* 13, 1 (2003), 87–101.
- Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing* 26, 5 (1997), 1484–1509.
- Sieg, W. Calculations by man and machine: Conceptual analysis. In W. Sieg, R. Sommer, and C. Talcott, Eds.,

Reflections on the Foundations of Mathematics, Association for Symbolic Logic, Natick, MA, 2002, 390–409.

- Sieg, W. On computability. In A. Irvine, Ed., *Handbook of the Philosophy of Mathematics*, Elsevier, 2009.
- Stannett, M. X-machines and the halting problem: Building a super-Turing machine. *Formal Aspects of Computing* 2, (1990), 331–341.
- Wegner, P. and Goldin, D. Computation beyond Turing machines. *Commun. ACM* 46, 4 (Apr. 2003), 100–102.

Jack Copeland (jack.copeland@canterbury.ac.nz) is a professor of Philosophy at the University of Canterbury, New Zealand, where he is also the director of the Turing Archive for the History of Computing.

Eli Dresner (dresner@post.tau.ac.il) is a member of the Gershon H. Gordon Faculty of Social Sciences at Tel Aviv University.

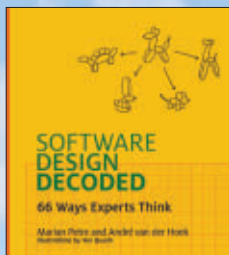
Diane Proudfoot (diane.proudfoot@canterbury.ac.nz) is an associate professor (reader) of Philosophy at the University of Canterbury, where she is also the co-director of the Turing Archive for the History of Computing.

Oron Shagrir (shagrir@cc.huji.ac.il) is a professor of Philosophy and Cognitive Science at the Hebrew University of Jerusalem.

Copyright held by authors.



The MIT Press



Software Design Decoded

66 Ways Experts Think
Marian Petre
and André van der Hoek
illustrations by Yen Quach

"This is the book I wish I'd had around throughout my journey as a software architect. It's charming, approachable, and full of wisdom—you'll learn things you'll come back to again and again."

—Grady Booch, IBM Fellow and Chief Scientist, IBM



Driverless

Intelligent Cars and the Road Ahead
Hod Lipson and Melba Kurman

"Driverless vehicles are poised to usher in a massive disruption of our transportation system, our urban landscapes, our economy—and quite possibly the very fabric of society. Anyone who wants to understand what's coming must read this fascinating book."

—Martin Ford, *New York Times* best-selling author of *Rise of the Robots*

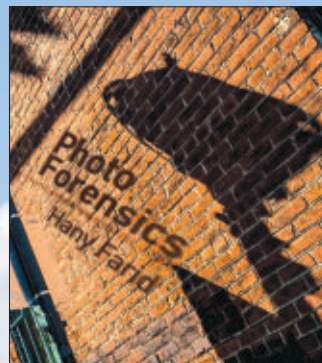


Photo Forensics

Hany Farid

"... a unique and delightful book, a tour of computational photography by a forgery sleuth. I loved it. Hany Farid, preeminent in this field, offers code, case studies, and advice to help readers find manipulated images."

—William T. Freeman, MIT; Research Scientist, Google



Men, Machines, and Modern Times

50th Anniversary Edition

Elting E. Morison
foreword by Rosalind Williams
with remarks by Leo Marx

"The most brilliant, original, and absorbing book in American history I have read for some time."

—Arthur Schlesinger, Jr.

mitpress.mit.edu

Viewpoint

Technology and Academic Lives

Considering the need to create new modes of interaction and approaches to assessment given a rapidly evolving academic realm.

I'M OVERCOMMITTED AT the moment. I find the university at the center of a huge work speedup.”
—Vice provost and dean

“The cult of busy-ness ... is really poisonous. People don’t want to be seen to have time for a leisurely intellectual conversation, because having time for that means they are insufficiently busy. They should be heading to the airport to go hustle grant money.”

—Department chair

From Dagstuhl to *Doonesbury*, we hear about the plight of academics. Curious, I consulted over a dozen computer and information science faculty, department chairs, deans, and other administrators from leading research universities. Details differed, but heightened demands and shifting evaluation criteria were consistently raised. What drove the changes? They mentioned budget pressure and rising expectations, but unanticipated consequences of new technologies appear to be at the heart of it. Technology has dramatically improved research and teaching, yet it can also complicate academic life.

Let’s consider how faculty were assessed in three worlds: technology-intensive university departments in 1975, when few had Internet access; in 1995, with the Internet but no Web services; and today. I was a graduate student in 1975, climbing the aca-



demic ladder in 1995, and am now an affiliate professor.^a

1975: The Ivory Tower Before Silicon

The world of 1975 is not easy to bring into focus, even for those of us who were there. No Internet, Web, mobile phones, Federal Express, or fax.^b Long-distance telephone was expensive. Conferences

were difficult to organize and few in number. Research labs and departments tended to be fiefdoms or small communities, each existing in a bubble.

We relied on people who were within walking distance. Faculty thoroughly trained their graduate students to be full collaborators. They invested in explaining their research carefully to departmental colleagues to get feedback, ideas, and possible collaborations. Departmental colloquia were well attended; faculty often commented cogently on talks outside their area. At UCSD, each graduate student presented a project to the department at the end

a My three snapshots of academic life are from graduate work at Stanford, MIT, and UCSD; rising from assistant to full professor at UC Irvine in the 1990s; and now an affiliate professor at the University of Washington.

b Facsimile machines existed but were not widely used. I first sent a fax in the late 1980s.

INTERACTIONS



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

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



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

Follow us on Facebook and Twitter  

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

Association for
Computing Machinery



of the first year, providing all faculty with a view into their colleagues' work, methods, and student selection and training. Faculty helped one another, building the department's reputation in an outside world that was known mainly through journal articles.

It wasn't idyllic—there were factional disputes, tenure anxiety, and 'publish or perish.' Nevertheless, high familiarity enabled faculty promotion cases to be handled effectively within a department and school.

The Internet Arrives

In late 1979, halfway through my Ph.D. journey, our lab connected to the ARPANET. Faculty and staff could access the computer and the ARPANET via modem from home. For students, the Internet precursor was largely a curiosity. For faculty, it turned out to be more significant.

Taking a postdoc in the U.K. in 1982 was like parachuting in with the clothes on my back. Communication with U.S. family, friends, and colleagues was epistolary and about one in 10 airmail letters went by sea and arrived a month late. On different occasions, the two senior professors from my UCSD lab visited and presented research co-authored with people outside our lab. I was stunned. This work had begun when I was a student but was never discussed in the lab. Prior to the ARPANET, everything was discussed. Suddenly, faculty could work with distant colleagues. They no longer had to educate departmental colleagues to get strong feedback. With faculty more focused on external discussions and collaborations, departmental meetings became less central to academic life.

1995: Assessment from Outside

After a stint in industry, I joined a remarkably democratic department: All faculty participated in all evaluations. Within months, as an assistant professor, I was voting on associate and full professor cases. Faculty were no longer well acquainted with one another's work; we relied extraordinarily heavily on external letters. This was not good news for those of us in sub-fields in which most professors had recently arrived from industry (such as IBM Research and Bell Labs), where refer-

ence letters were more balanced. The academic norm was lavish praise; readers looked for subtle negatives. My own tenure and full professor promotions encountered reference-letter turbulence but survived.

Being able to interact at a distance has phenomenal benefits, but it does reduce local interaction. This diminishing of community was reflected in the increased outsourcing of faculty assessment. It didn't seem ideal to weight the subjective opinions of outsiders so heavily, but it was manageable.

2015: The Information Age

As one of few full professors in HCI in the early 2000s, I was asked to write many letters for appointments and promotions. Over time, urgent personal requests gave way to form letters that often omit key information. External letters appeared to be losing their dominant role. This hypothesis was supported in my recent inquiries. A wealth of digital data is now available and often relied upon. One department chair wrote, "There is a pervasive atmosphere of pressure and obsessive quantification, and yes, it affects senior people, too. In part this is because as the senior ranks fill with people who came up through the obsessive quantification, the attitudes become entrenched ... People who don't buy into obsessive quantification get filtered out."

Budget-strapped legislatures demand that state universities show evidence of impact. Great teaching may not compensate for weak research, but bad teaching is tolerated less. Public universities with reduced state support and private universities facing higher costs count on rainmaking. "Funding is the first consideration for promotion to full, even though that is

A highly beneficial technology can have an undesirable side effect.

not written,” observed a Promotion and Tenure Committee chair. NSF adds to reporting requirements as it awards proportionally fewer grants. In addition to teaching evaluations and grant funding, performance measures include citations and downloads, journal and conference tiering, g- and h-indices, and students matriculated. Many universities use *Academic Analytics*. External letters remain a factor, especially in appointments for which the digital record is sparser, but their subjectivity seems alien in a world of ‘big data’ and metrics.

Faculty CVs sprout new categories, such as mass media notices. A dean said, “I don’t think much of the [research] literature is read anymore. Most of the communication is through coverage in national media (of which the research publication is the pretext).” Another dean said “A *New York Times* op-ed piece, that’s huge!”

Faculty assessment is back in local hands, but the assessment is not based on the direct familiarity that marked pre-Internet academic communities. Collaborating and gathering information across distance accelerates scholarship, yet it erodes the sense of community. For example, when faculty invested in and relied on a network of local colleagues, job-hopping exacted a high price. Today’s weaker local ties enable professors to play on a world stage, with benefits and costs. It is working, but stresses are felt.

Unintended Consequences

A highly beneficial technology can have an undesirable side effect. Automation of routine tasks is great, but it can put people out of work. Transparency is good, but it can facilitate unwanted surveillance. Collaborating over distances and with more people is fantastic, but it means less interaction with nearby colleagues and each collaborator on average.

Effects of technology on community extend beyond faculty assessment. Our conferences once focused on community building. As I have written previously,^{1,2} a likely indirect effect of two of our most valued technologies—word processing and digital archives—was that our conferences became arbiters of quality with high rejection rates that undermine cohesion.

If any technology has an irresistible trajectory, digital technology does.

We see the unquestioned value of technology but often underestimate the inextricably intertwined costs: reduced depth of social connectedness, work speedups, a troubled publication culture, lost jobs, privacy intrusions, cybercrime, distributed terrorist networks, and so on.

Addressing Indirect Consequences

Proposed solutions are often exhortations to roll back the clock: restrict academic assessments to a few high-quality publications, return journals to preeminence, build technical barriers to privacy incursions, and so on. Unfortunately, the underlying forces that brought us here are powerful, because they also bring tremendous benefits.

We smile at the story of King Canute placing his throne on the beach and commanding the incoming tide to halt. A technological tide is sweeping in that will never retreat. We can’t command away undesirable side effects by issuing policy statements. Perhaps to protect valued heritage we can build a massive seawall Netherlands-style—but only if we understand the tidal forces, decide what to save and what to let go, budget for the costs, and accept that unanticipated developments could render our efforts futile, as would a 10-centimeter rise in ocean levels.

An irresistible force—technology—meets an immovable object—our genetic constitution. Behaviors that we inherited from ancestors who lived in tightly knit communities do not stand in opposition to technology, but they shape our reactions. Can we control those tendencies, build seawalls to protect us when human nature risks interacting disadvantageously with useful technologies that it did not evolve alongside? Perhaps, if we recognize the forces at work. But we have little time to develop understanding in our world

of exponential growth, with the breath-taking wave that quickly follows the first perceptible effect.³

To assert that we are masters of our destiny is to set thrones on the beach. We cannot always control the consequences of introducing a new technology. Bows and arrows were doomed as weapons of war by the invention of a musket that anyone could load, point, and shoot. The introduction of literacy fundamentally changed previously oral cultures; money fundamentally changed barter cultures. These technologies brought advantages and disadvantages. Socrates famously railed against writing, but despite his eloquence, he couldn’t slow it down. If any technology has an irresistible trajectory, digital technology does. As it is woven ever more deeply into the fabric of our lives, it will be more difficult to link subtle effects to causes, and more challenging to address effects that are unexpected.

Conclusion

The close-knit communities of our academic predecessors are melting away. They won’t come back. We must create new modes of interaction and approaches to assessment that are less impersonal and stressful, for faculty members and job candidates, grant applicants and conference submitters, for students in MOOCs and traditional courses, and more broadly throughout society. Just as technology contributed to the problems, it will help address them. Software is versatile and we are innovative. The best metaphor may not be a seawall that strives to block the forces unleashed by technology and forestall change; rather, consider a martial art that enables us, when armed with a deep understanding of those forces, to redirect them to achieve positive outcomes. 

References

1. Grudin, J. Journal-conference interaction and the competitive exclusion principle. *ACM Interactions* 20, 1, (2013), 68–73.
2. Grudin, J. Technology, conferences, and community. *Commun. ACM*, 54, 2, (Feb. 2011), 41–43.
3. Grudin, J. The demon in the basement. *ACM Interactions* 13, 6 (2006), 50–53.

Jonathan Grudin (jgrudin@microsoft.com) is a principal researcher at Microsoft Research in Redmond, WA.

Copyright held by author.

**Expect to be constantly
and pleasantly befuddled.**

BY PAT HELLAND

The Power of Babble

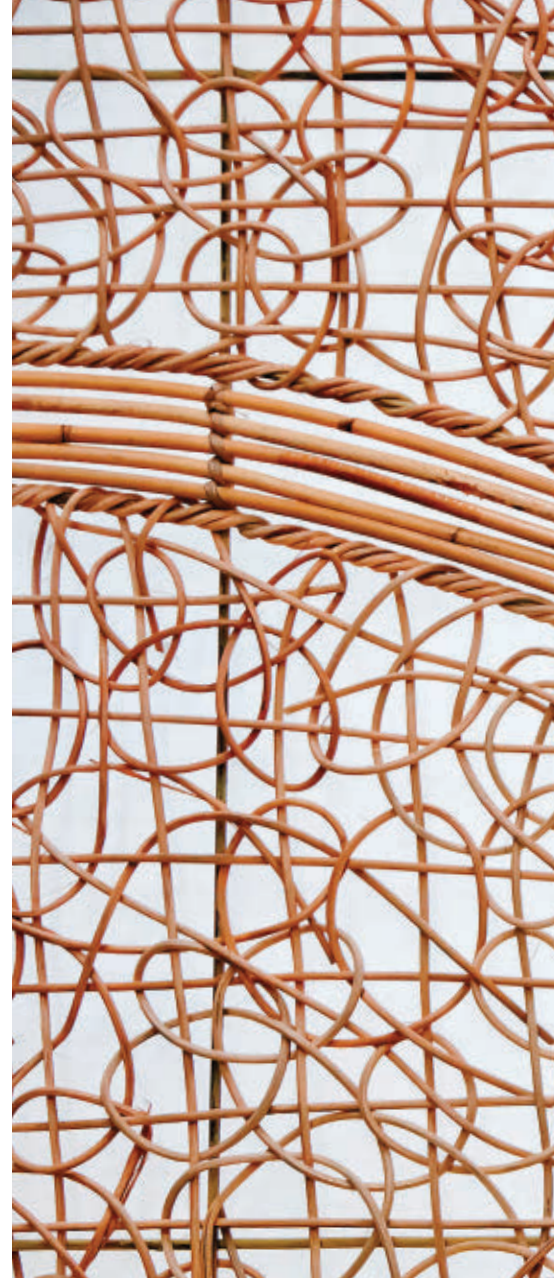
METADATA DEFINES THE shape, the form, and how to understand our data. It is following the trend taken by natural languages in our increasingly interconnected world. While many concepts can be communicated using shared metadata, no one can keep up with the number of disparate new concepts needed to have a common understanding.

English is the lingua franca of the world, yet there are many facets of humanity and the concepts held by different people that simply cannot be captured in English no matter how pervasive the language. In fact, English itself has nooks, crannies, dialects, meetups, and teenager slang that innovate and extend

its permutations with usages that usually do not converge. My personal idiolect shifts depending on whether I am speaking to a computer science audience, my team at work with its contextual usages, my wife, my grandkids, or the waiter at a local restaurant. Different communities of people extend English in different ways.

Computer systems have an emerging and increasing common metadata for interoperability. XML and now JSON fill similar roles by making the parsing of messages easy and common. It's great we are no longer arguing over ASCII versus EBCDIC, but that is hardly the most challenging problem of understanding.

As we move up the stack of understanding, new subtleties constantly emerge. Just when we think we understand, the other guy has some crazy new ideas!





As much as we would like to have complete understanding of each other, independent innovation is far more important than crisp and clear communication. Our economic future depends on the “power of babble.”

The Apocalypse of Two Elephants

To facilitate communications, the computing industry, various companies, and other organizations try to establish standard forms of communication. We see TCP, IP, Ethernet, and other communication standards as well as XML, JSON, and even ASCII making it easier to communicate. Above this, there are vertical specific standards (for example, healthcare and manufacturing standards). Many companies have internal communication standards as well.

Dave Clark of MIT observed that successful standards happen only if

they are lucky enough to slide into a trough of inactivity after a flurry of research and before huge investments in productization (see Figure 1). This observation is known as the Apocalypse of the Two Elephants (although Clark actually didn’t name it that).¹

Standards that happen in this trough are effective and experience little competition. If a standard doesn’t emerge here or the trough is squished by the two humps overlapping, it’s a much murkier road forward.

The best de jure standards are rubber stamps over de facto standards.

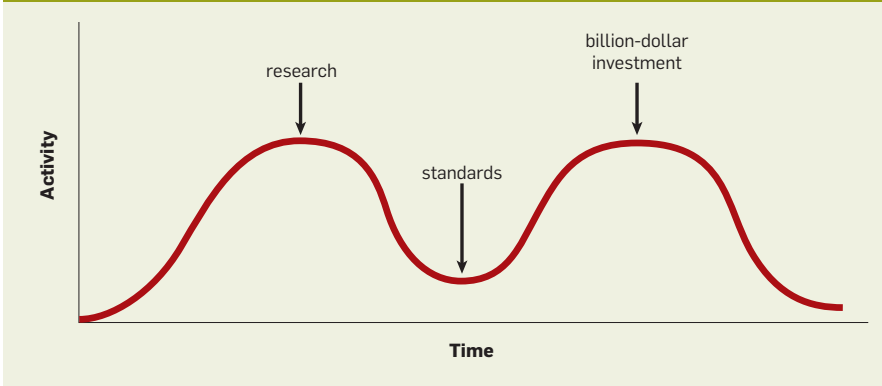
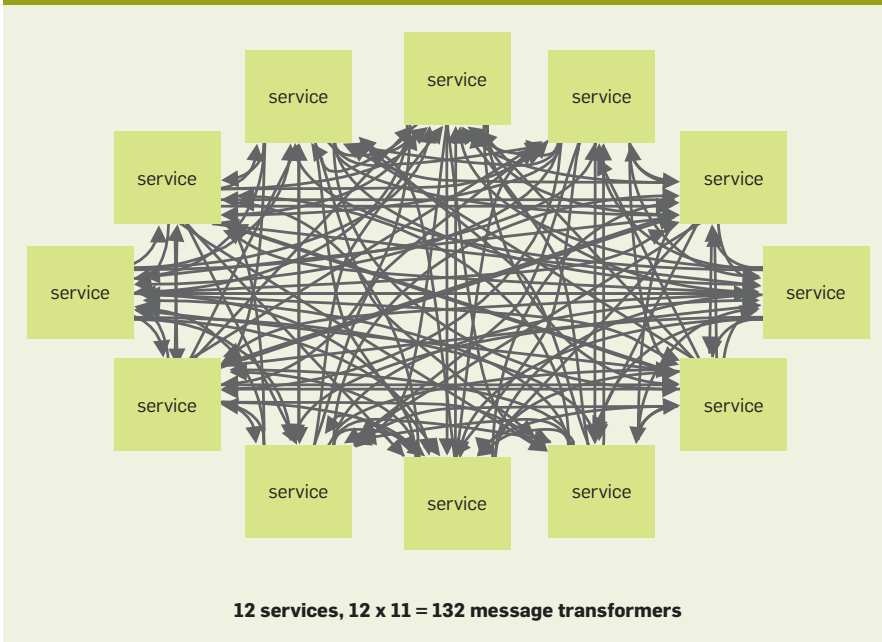
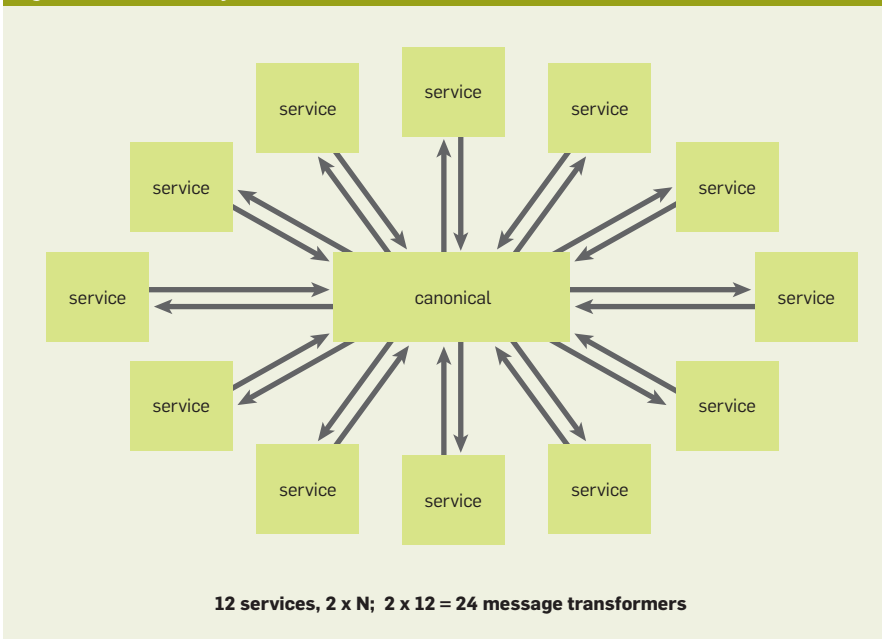
If there’s no de facto standard to start from, then the de jure standard typically contains the union of all ideas discussed by the committee. Natural selection relegates these standards and their clutter to history books.

The Dialects of the Business World

Computer systems and applications tend to be developed independently to support the special needs of their users. In the past, each system would be bespoke and support detailed specifications. Increasingly, shared application platforms are leveraged, either on premises or in the cloud. In these common apps, there is common metadata—at least as far as the apps have a common heritage.

When applications are independently developed, they have disparate concepts and representations. Many of these purchased applications are designed for extensions. As the specific customer gloms extensions onto the side of the app, this impacts the shape, form, and meaning of its internal and shared data.

When there’s a common application lineage, there’s a common understand-

Figure 1. The Apocalypse of Two Elephants.**Figure 2. Least-lossy conversion.****Figure 3. 'Double lossy' conversion.**

ing of its data. Popular ERP (enterprise resource planning), CRM (customer relationship management), and HRM (human resources management) applications have their ways of solving business problems, and different companies that have adopted these solutions may find it easier to interoperate.

Interneecine Interop

Still, challenges of understanding may exist even across departments or divisions of the same company. A large conglomerate may sell many products, including light bulbs, dishwashers, locomotives, and nuclear power plants. I would hazard a guess that it doesn't have a single canonical customer record type.

Of course, mergers and divestitures impact a company's metadata. I know from personal experience how difficult it is to change my mailing address with a bank or insurance company. They can't seem to track down all the systems that record my address even over the course of a year. It's not a big surprise that they have a hard time managing their metadata.

What'd You Say?

Whenever there are two representations of data, either somebody adapts or the fidelity of the translation suffers. In many cases, the adaptation is driven by economic power. When a manufacturer wants to sell something to a huge retailer, it may be told exactly the shape, form, and semantics of the messaging between the companies. To get the business, the manufacturer will figure it out!

The dog wags the tail. In any communication partnership, the onus to adapt rests on the side that most needs the relationship to work.

Translating between two data representations may very likely be lossy. Not all of the information in one form can be moved to the other form. It's highly likely that some stuff will be nulled out or possibly translated into a form that doesn't precisely map.

Each translation is lossy. By the time the translation occurs, a loss of knowledge has occurred. The best results will be from dedicated transformations designed to take exactly one source and translate it as best as possible to exactly one target. This is the least lossy

form of translation. Unfortunately, this results in a boatload of translators. Creating a specific conversion for each source and destination pair results in great conversion fidelity but also results in N^2 converters (see Figure 2).

What to do? Many times, we simply capture a canonical representation and do two data translations: first, a lossy translation into the canonical representation; then, a lossy translation from the canonical representation into the target representation. This is *double-lossy* and just doesn't supply as good a result.

Why do the translation to a canonical form? Because only $2 \times N$ translators are needed for N sources, and that is a heck of a lot fewer than N^2 , as N gets large. Using canonical metadata as a common translation reduces the number of converters but results in a double-lossy conversion (see Figure 3).

In most cases, people use canonical metadata to bound complexity but add specific source-to-target translators when the lossiness is too large.

What Color Are Your Rose-Colored Glasses?

We all see stuff couched in terms of a set of assumptions. This is a worldview that allows us to interpret incoming information. This interpretation may be right or wrong, but, more importantly, it is right or wrong for our subjective usage.

Computer systems are invariably designed for a certain company, department, or group. The data is typically cast into a meaning and use that are appropriate for one side but lose their deeper meaning through the translation.

Sometimes, the meaning and understanding of some data are deeply couched in cultural issues. Any translation to a new environment and culture simply loses all meaning. Reading about daily life in Medieval Europe doesn't help much unless you study the relationships between serfs and lord as well as between men and women. Only then can you understand the actions described in the book. Similarly, in any discussion of privacy, cultural expectations must be addressed. In North America and Europe, protecting against the damage that may result by disclosing a medical challenge is paramount. In India, the essential need to vet a prospective spouse for your child

is deemed more important than holding an illness private. Communication cannot take place without understanding the assumptions and interpreting through that lens.

The artificial language Esperanto was created in 1887 with the hope of achieving a common shared natural language for all people. Some folks grabbed hold and used it to write and share. Some say a few million speak it today.

The use of Esperanto has been waning, however. Each of the approximately 6,000 languages spoken by different communities in the world has its own flavor and nuance. You can say certain things in one language that you just can't say in another one.

Diverse and Homogeneous

The words and phrases people use and the metadata that applications use follows a similar pattern. With a common codebase DNA and history, some meanings are the same. As time, evolution, and commingling occur, it's more difficult to understand one another.

New software applications either in the cloud or on premises sometimes offer enough business benefit that enterprises adapt their ways of doing business to fit the application. The new user adopts the canonical representation of data and business processes by sheer hard work. When the business value of the software is high enough, mapping to it is cost effective. Now the enterprise is much more closely aligned to the new approach and to interoperating with other enterprises sharing the new data and process.

Next, the enterprise will begin to extend the system using extensibility features. These extensions can then become a source of misunderstanding, but they bring business value to the enterprise.

The U.S., Canada, and many other Western countries have tremendous diversity in their populations. New arrivals bring new customs. They work to understand the existing customs in their new home. While there are many differences at first, in a few short years the immigrants fit in. Their children are deeply ingrained in the new country, even though they still like some of that food their mom cooked at home. That food becomes as American (or English or German) as pizza, tacos, and falafel.

Similarly, the base metadata continues to move and adjust as it assimilates those new messages and fields that made no sense at all a short time ago.

Relishing Diversity

While not understanding another party is a pain, it probably means that innovation and growth have occurred. Economic forces will drive when and where it's worth the bother to invest in deeper understanding.

Playing loose with understanding allows for better cohesion, as exemplified by Amazon's product catalog and the search results from Google or Bing. Remember that in many cases, cultural and contextual issues will drive how something is interpreted. Extensible data does not have a prearranged understanding. Translating between representations is lossy and frequently involves a painful trade-off between expensive handcrafted translators and even lossier multiple translations.

Personally, as the years have gone by, I've gotten much more relaxed about the things I don't know and don't understand. A lot of stuff confuses me! As we interoperate across disparate boundaries, it would do us well to remember that the less stressed we are about perfect understanding and agreement, the better we will all get along. Moving forward, I expect to be constantly and pleasantly befuddled by the power of babble. 

Related articles on queue.acm.org

Immutability Changes Everything

Pat Helland

http://queue.acm.org/detail.cfm?id_2884038

Standardized Storage Clusters

Garth Goodson et al.

<http://queue.acm.org/detail.cfm?id+1317402>

Search Considered Integral

Ryan Barrows and Jim Traverso

<http://queue.acm.org/detail.cfm?id=1142068>

Reference

1. Clark, D. 2009. The Apocalypse of Two Elephants, or 'what I really said.' Advanced Network Architecture. MIT CSAIL; <http://groups.csail.mit.edu/ana/People/DDC/Apocalypse.html>.

Pat Helland has been implementing transaction systems, databases, application platforms, distributed systems, fault-tolerant systems, and messaging systems since 1978. He currently works at Salesforce.

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

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

Advanced synchronization methods can boost the performance of multicore software.

BY ADAM MORRISON

Scaling Synchronization in Multicore Programs

DESIGNING SOFTWARE FOR modern multicore processors poses a dilemma. Traditional software designs, in which threads manipulate shared data, have limited scalability because synchronization of updates to shared data serializes threads and limits parallelism. Alternative distributed software designs, in which threads do not share mutable data, eliminate synchronization and offer better scalability. But distributed designs make it challenging to implement features that shared data structures naturally provide, such as dynamic load balancing and strong consistency guarantees, and are simply not a good fit for every program.

Often, however, the performance of shared mutable data structures is limited by the synchronization methods in use today, whether lock-based or lock-free.

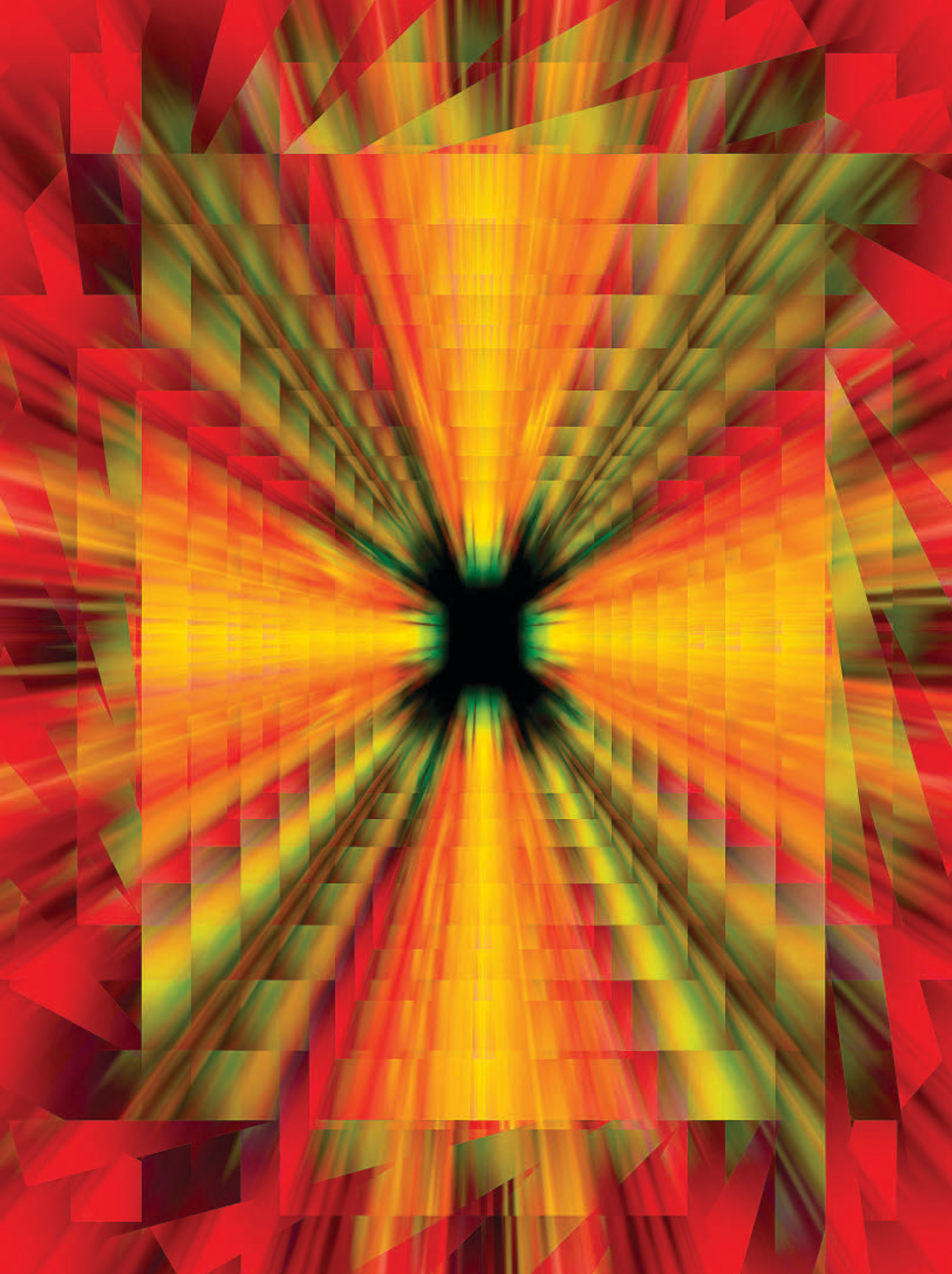
To help readers make informed design decisions, this article describes advanced (and practical) synchronization methods that can push the performance of designs using shared mutable data to levels that are acceptable to many applications.

To get a taste of the dilemmas involved in designing multicore software, let us consider a concrete problem: implementing a *work queue*, which allows threads to enqueue and dequeue work items—events to handle, packets to process, and so on. Issues similar to those discussed here apply in general to multicore software design.¹⁴

Centralized shared queue. One natural work queue design (depicted in Figure 1a) is to implement a centralized shared (thread-safe) version of the familiar FIFO (first in, first out) queue data structure—say, based on a linked list. This data structure supports enqueueing and dequeuing with a constant number of memory operations. It also easily facilitates dynamic load balancing: because all pending work is stored in the data structure, idle threads can easily acquire work to perform. To make the data structure thread-safe, however, updates to the head and tail of the queue must be synchronized, and this inevitably limits scalability.

Using locks to protect the queue serializes its operations: only one core at a time can update the queue, and the others must wait for their turns. This ends up creating a sequential bottleneck and destroying performance very quickly. One possibility to increase scalability is by replacing locks with *lock-free* synchronization, which directly manipulates the queue using atomic instructions,^{1,11} thereby reducing the amount of serialization. (Serialization is still a problem because the hardware cache coherence mechanism¹ serializes atomic instructions updating the same memory location.) In practice, however, lock-free synchronization often does not outperform lock-based synchronization, for reasons to be discussed later.

Partially distributed queue. Al-



ternative work-queue designs seek scalability by distributing the data structure, which allows for more parallelism but gives up some of the properties of the centralized shared queue. For example, Figure 1b shows a design that uses one SPMC (single-producer/multiple-consumer) queue per core. Each core enqueues work into its queue. Dequeues can be implemented in various ways—say, by iterating over all the queues (with the starting point

selected at random) until finding one containing work.

This design should scale much better than the centralized shared queue: enqueues by different cores run in parallel, as they update different queues, and (assuming all queues contain work) dequeues by different cores are expected to pick different queues to dequeue from, so they will also run in parallel.

What this design trades off, though,

is the data structure's *consistency guarantee*. In particular, unlike the centralized shared queue, the distributed design does not maintain the cause and effect relation in the program. Even if core P_1 enqueues x_1 to its queue after core P_0 enqueues x_0 to its queue, x_1 may be dequeued before x_0 . The design weakens the consistency guarantees provided by the data structure.

The fundamental reason for this weakening is that in a distributed design, it is difficult (and slow) to combine the per-core data into a *consistent* view of the data structure—one that would have been produced by the simple centralized implementation. Instead, as in this case, distributed designs usually weaken the data structure's consistency guarantees.^{5,8,14} Whether the weaker guarantees are acceptable or not depends on the application, but figuring this out—reasoning about the acceptable behaviors—complicates the task of using the data structure.

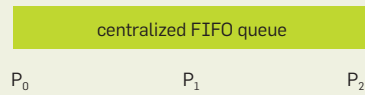
Despite its more distributed nature, this per-core SPMC queue design can still create a bottleneck when load is not balanced. If only one core generates work, for example, then dequeuing cores all swoop on its queue and their operations become serialized.

Distributed queue. To eliminate many-thread synchronization altogether, you can turn to a design such as the one depicted in Figure 1c, with each core maintaining one SPSC (single-producer/single-consumer) queue for each other core in the system, into which it enqueues items that it wishes its peer to dequeue. As before, this design weakens the consistency guarantee of the queue. It also makes dynamic load balancing more difficult because it chooses which core will dequeue an item in advance.

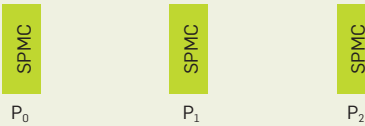
Motivation for improving synchronization. The crux of this discussion is that obtaining scalability by distributing the queue data structure trades off some useful properties that a centralized shared queue provides. Usually, however, these trade-offs are clouded by the unacceptable performance of centralized data structures, which obviates any benefit they might offer. This article makes the point that much of this poor performance is a result of inefficient synchronization methods. It surveys advanced synchronization

Figure 1. Possible designs for a work queue.

(a) all cores access a centralized shared queue



(b) each core has an SPMC queue



(c) each core shares an SPSC queue with every other core

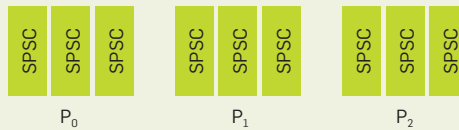
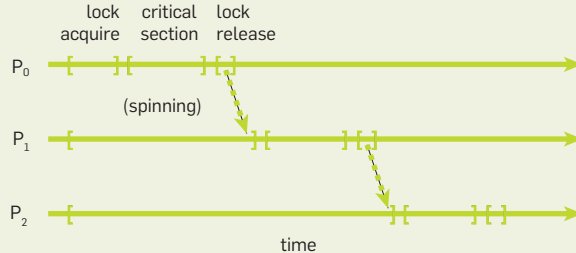
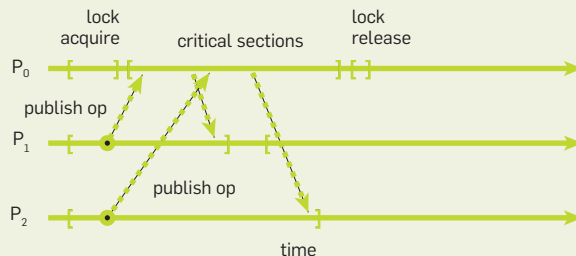


Figure 2. Critical path of lock-based code.

(a) queue-based lock



(b) delegation



methods that boost the performance of centralized designs and make them acceptable to more applications. With these methods at hand, designers can make more informed choices when architecting their systems.

Scaling Locking with Delegation

Locking inherently serializes executions of the critical sections it protects. Locks therefore limit scaling: the more cores there are, the longer each core has to wait for its turn to execute the critical section, and so beyond some number of cores, these waiting times dominate the computation. This scalability limit, however, can be pushed quite high—in some cases, beyond the scales of current systems—by *serializing more efficiently*. Locks that serialize more efficiently support more operations per second and therefore can handle workloads with more cores.

More precisely, the goal is to minimize the computation's *critical path*: the length of the longest series of operations that have to be performed sequentially because of data dependencies. When using locks, the critical path contains successful lock acquisitions, execution of the critical sections, and lock releases.

As an example of inefficient serialization that limits scalability, consider the *lock contention* that infamously occurs in simple spin locks. When many cores simultaneously try to acquire a spin lock, they cause its cache line to bounce among them, which slows down lock acquisitions/releases. This increases the length of the critical path and leads to a performance “melt-down” as core counts increase.²

Lock contention has a known solution in the form of scalable queue-based locks.^{2,10} Instead of having all waiting threads compete to be the next one to acquire the lock, queue-based locks line up the waiting threads, enabling a lock release to hand the lock to the next waiting thread. These hand-offs, which require only a constant number of cache misses per acquisition/release, speed up lock acquisition/release and decrease the length of the critical path, as depicted in Figure 2a: the critical path of a queue-based lock contains only a single transfer of the lock's cache line (dotted arrow).



The crux of this discussion is that obtaining scalability by distributing the queue data structure trades off some useful properties that a centralized shared queue provides.



Can lock-based serialization be made even more efficient? The *delegation* synchronization method described here does so: It eliminates most lock acquisition and releases from the critical path, and it speeds up execution of the critical sections themselves.

Delegation. In a delegation lock, the core holding the lock acts as a server and executes the operations that the cores waiting to acquire the lock wish to perform. Delegation improves scalability in several ways (Figure 2b). First, it eliminates the lock acquisitions and releases that otherwise would have been performed by waiting threads. Second, it speeds up the execution of operations (critical sections), because the data structure is hot in the server's cache and does not have to be transferred from a remote cache or from memory. Delegation also enables new optimizations that exploit the semantics of the data structure to speed up critical-section execution even further, as will be described.

Implementing delegation. The idea of serializing faster by having a single thread execute the operations of the waiting threads dates to the 1999 work of Oyama et al.¹³ but the overheads of their implementation overshadow its benefits. Hendler et al.⁶ in their *flat combining* work, were first to implement this idea efficiently and to observe that it facilitates optimizations based on the semantics of the executed operations.

In the flat combining algorithm, every thread about to acquire the lock posts the operation it intends to perform (for example, `dequeue` or `enqueue(x)`) in a shared *publication list*. The thread that acquires the lock becomes the server; the remaining threads spin, waiting for their operations to be applied. The server scans the publication list, applies pending operations, and releases the lock when done. To amortize the synchronization cost of adding records to the publication list, a thread leaves its publication record in the list and reuses it in future operations. Later work explored piggybacking the publication list on top of a queue lock's queue,⁴ and boosting cache locality of the operations by dedicating a core for the server role instead of having threads opportunistically become servers.⁹

Figure 3. Enqueue/dequeue throughput comparison.

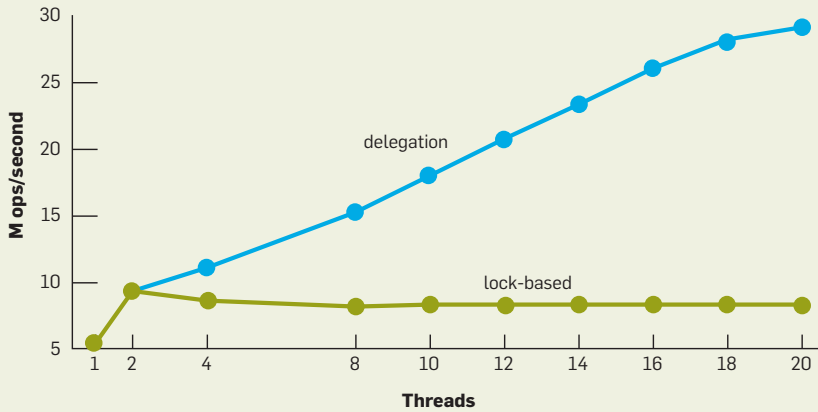


Figure 4. Lock-free linking of a node to the head of a linked list.

```

struct Node {
    struct Node* next;
    void* value;
}
// Pointer to head of the list
Node* head = NULL;

void enqueue (void* v) {
    Node *old, *new = malloc();
    new->value = v;
    while (true) {
        old = head;
        new->next = old;
        if ( CAS(&head, old, new) )
            return;
    }
}

```

Semantics-based optimizations. The server thread has a global view of concurrently pending operations, which it can leverage to optimize their execution in two ways:

► *Combining.* The server can combine multiple operations into one and thereby save repeated accesses to the data structure. For example, multiple counter-increment operations can be converted into one addition.

► *Elimination.* Mutually canceling operations, such as a counter increment and decrement, or an insertion and removal of the same item from a set, can be executed without modifying the data structure at all.

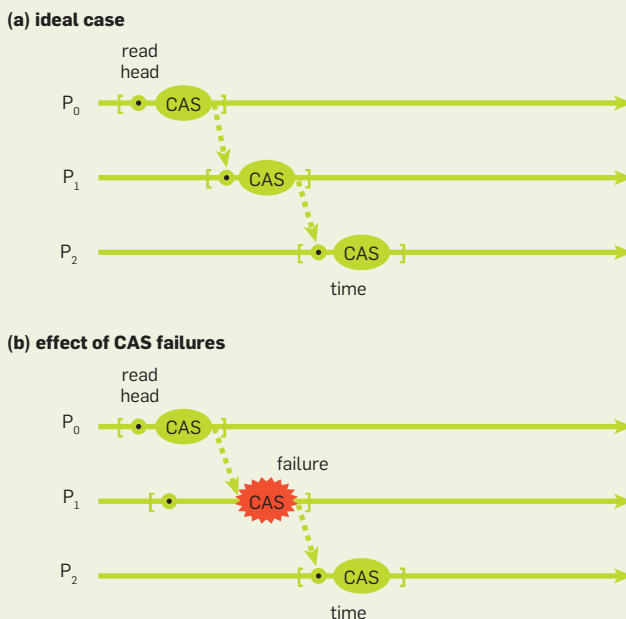
Deferring delegation. For operations that only update the data structure but do not return a value, such as enqueue(), delegation facilitates an optimization that can sometimes eliminate serialization altogether. Since these operations do not return a response to the invoking core, the core does not have to wait for the server to execute them; it can just log the requested operation in the publication list and keep running. If the core later invokes an operation whose return value depends on the state of the data structure, such as a dequeue(), it must then wait for the server to apply all its prior operations. But until such a time—which in update-

heavy workloads can be rare—all of its operations execute asynchronously.

The original implementation of this optimization still required cores to synchronize when executing these deferred operations, since it logged the operations in a centralized (lock-free) queue.⁷ Boyd-Wickizer et al.,³ however, implemented deferred delegation without any synchronization on updates by leveraging systemwide synchronized clocks. Their OpLog library logs invocations of responseless update operations in a per-core log, along with their invocation times. Operations that read the data structure become servers: they acquire the locks of all per-core logs, apply the operations in timestamp order, and then read the updated data-structure state. OpLog thus creates scalable implementations of data structures that are updated heavily but read rarely, such as LRU (least recently used) caches.

Performance. To demonstrate the benefits of delegation, let's compare a lock-based work queue to a queue implemented using delegation. The lock-based algorithm is Michael and Scott's *two-lock queue*.¹¹ This algorithm protects updates to the queue's head and tail with different locks, serializing operations of the same type but allowing enqueues and dequeues to run in parallel. Queue-based CLH (Craig, Landin, and Hagerstein) locks are used in the evaluated implementation of the lock-based algorithm. The delegation-based queue is Fatourou and Kallimanis's *CC-Queue*,⁴ that adds delegation to each of the two locks in the lock-based algorithm. (It thus has two serv-

Figure 5. Critical path of lock-free updating head of linked list.



ers running: one for dequeues and one for enqueues.)

Figure 3 shows enqueue/dequeue throughput comparison (higher is better) of the lock-based queue and its delegation-based version. The benchmark models a generic application. Each core repeatedly accesses the data structure, performing pairs of enqueue and dequeue operations, reporting the throughput of queue operations (that is, the total number of queue operations completed per second). To model the work done in a real application, a period of “think time” is inserted after each queue operation. Think times are chosen uniformly at random from 1 to 100 nanoseconds to model challenging workloads in which queues are heavily exercised. The C implementations of the algorithms from Fatourou and Kallimanis’s benchmark framework (<https://github.com/nkallima/sim-universal-construction>) are used, along with a scalable memory allocation library to avoid `malloc` bottlenecks. No semantics-based optimization is implemented.

This benchmark (and all other experiments reported in this article) was run on an Intel Xeon E7-4870 (Westmere EX) processor. The processor has 10 2.40-GHz cores, each of which multiplexes two hardware threads, for a total of 20 hardware threads.

Figure 3 shows the benchmark throughput results, averaged over 10 runs. The lock-based algorithm scales to two threads, because it uses two locks, but fails to scale beyond that amount of concurrency because of serialization. In contrast, the delegation-based algorithm scales and ultimately performs almost 30 million operations per second, which is more than 3.5 times that of the lock-based algorithm’s throughput.

Avoiding CAS Failures in Lock-Free Synchronization

Lock-free synchronization (also referred to as nonblocking synchronization) directly manipulates shared data using atomic instructions instead of locks. Most lock-free algorithms use the CAS (compare-and-swap) instruction (or equivalent) available on all multicore processors. A CAS takes three operands: a memory address `addr`, an `old` value, and a new value.

It atomically updates the value stored in `addr` from `old` to `new`; if the value stored in `addr` is not `old`, the CAS fails without updating memory.

CAS-based lock-free algorithms synchronize with a CAS loop pattern: A core reads the shared state, computes a new value, and uses CAS to update a shared variable to the new value. If the CAS succeeds, this read-compute-update sequence appears to be atomic; otherwise, the core must retry. Figure 4 shows an example of such a CAS loop for linking a node to the head of a linked list, taken from Treiber’s classic LIFO (last in, first out) stack algorithm.¹⁵ Similar ideas underlie the lock-free implementations of many basic data structures such as queues, stacks, and priority queues, all of which essentially perform an entire data-structure update with a single atomic instruction.

The use of (sometimes multiple) atomic instructions can make lock-free synchronization slower than a lock-based solution when there is no

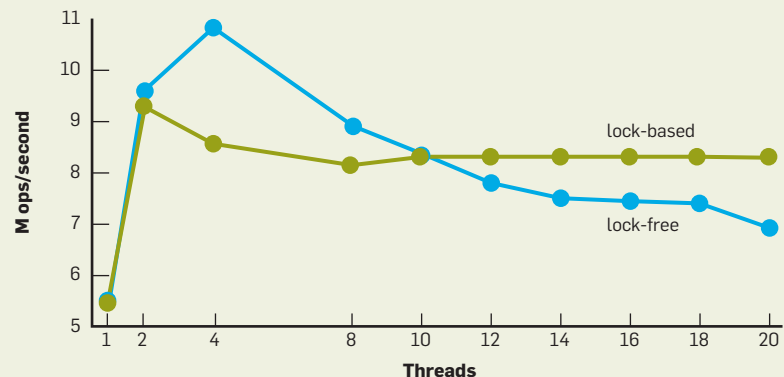
(or only light) contention. Under high contention, however, lock-free synchronization has the potential to be much more efficient than lock-based synchronization, as it eliminates lock acquire and release operations from the critical path, leaving only the data structure operations on it (Figure 5a).

In addition, lock-free algorithms guarantee that some operation can always complete and thus behave gracefully under high load, whereas a lock-based algorithm can grind to a halt if the operating system preempts a thread that holds a lock.

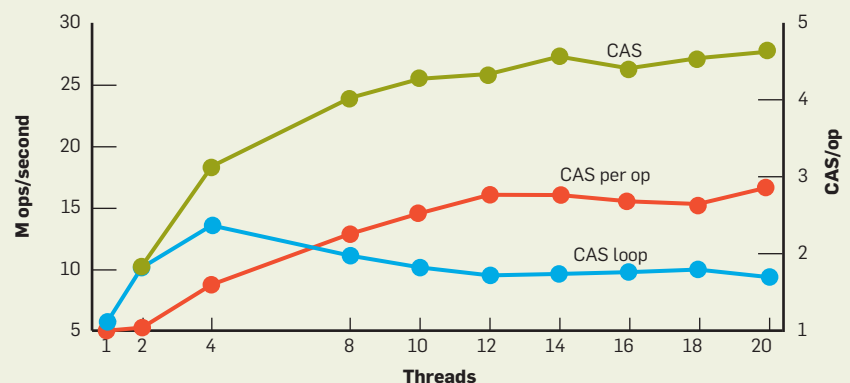
In practice, however, lock-free algorithms may not live up to these performance expectations. Consider, for example, Michael and Scott’s lock-free queue algorithm.¹¹ This algorithm implements a queue using a linked list, with items enqueued to the tail and removed from the head using CAS loops. (The exact details are not as important as the basic idea, which is similar in spirit to the example in Figure 4.) De-

Figure 6. Lock-free synchronization CAS failure problem.

(a) Lock-free vs. Lock-based Queue Throughput



(b) Effect of CAS Failures



spite this, as Figure 6a shows, the lock-free algorithm fails to scale beyond four threads and eventually performs worse than the two-lock queue algorithm.

The reason for this poor performance is CAS failure: as the amount of concurrency increases, so does the chance that a conflicting CAS gets interleaved in the middle of a core's read-compute-update CAS region, causing its CAS to fail. CAS

operations that fail in this way pile up less work on the critical path. Although these failing CASes do not modify memory, executing them still requires obtaining exclusive access to the variable's cache line. This delays the time at which later operations obtain the cache line and complete successfully (see Figure 5b, in which only two operations complete in the same time that three opera-

tions completed in Figure 5a).

To estimate the amount of performance wasted because of CAS failures, Figure 6b compares the throughput of successful CASes executed in a CAS loop (as in Figure 4) to the total CAS throughput (including failed CASes). Observe that the system executes contending atomic instructions at almost three times the rate ultimately observed in the data structure. If there were a way to make every atomic instruction useful toward completing an operation, you would significantly improve performance. But how can this be achieved, given that CAS failures are inherent?

The key observation to make is that the x86 architecture supports several atomic instructions that always succeed. One such instruction is FAA (fetch-and-add), which atomically adds an integer to a variable and returns the previous value stored in that variable. The design of a lock-free queue based on FAA instead of CAS is described next. The algorithm, named LCRQ (for linked concurrent ring queue),¹² uses FAA instructions to spread threads among items in the queue, allowing them to enqueue and dequeue quickly and in parallel. LCRQ operations typically perform one FAA to obtain their position in the queue, providing exactly the desired behavior.

The LCRQ Algorithm

This section presents an overview of the LCRQ algorithm; for a detailed description and evaluation, see the paper.¹² Conceptually, LCRQ can be viewed as a practical realization of the following simple but unrealistic queue algorithm (Figure 7). The unrealistic algorithm implements the queue using an *infinite* array, Q , with (unbounded) head and tail indices that identify the part of Q that may contain items. Initially, each cell $Q[i]$ is empty and contains a reserved value $\perp$ that may not be enqueued. The head and tail indices are manipulated using FAA and are used to spread threads around the cells of the array, where they synchronize using (uncontended) CAS.

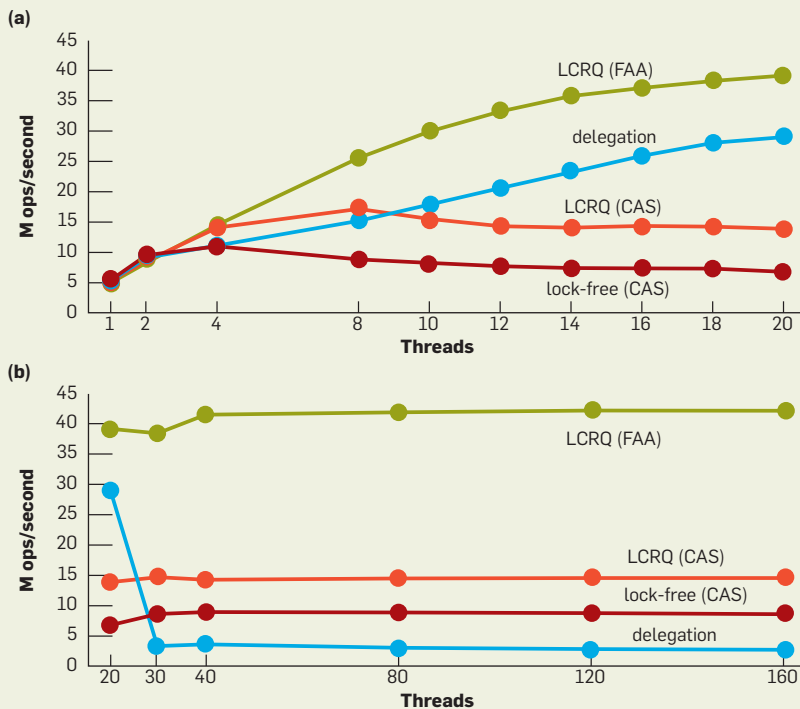
An enqueue (x) operation obtains a cell index t via an FAA on tail . The enqueue then atomically places x in $Q[t]$ using a CAS to update $Q[t]$ from $\perp$ to x . If the CAS succeeds, the enqueue operation completes; other-

Figure 7. Infinite array queue.

```
// The following defines a node
struct Cell {
    void* value;
}
// Queue is infinite array of nodes,
// with head and tail pointers.
Cell Q[] = {  $\perp$ ,  $\perp$ , ... };
int head = 0;
int tail = 0;

void enqueue(void* x) {
    while (true) {
        t = FAA(&tail, 1)
        if ( CAS(&Q[t],  $\perp$ , x) ) return
    }
}
void *dequeue() {
    while (true) {
        h = FAA(&head, 1)
        if ( !CAS(&Q[h],  $\perp$ ,  $\tau$ ) ) return Q[h]
        if ( tail  $\leq$  h+1 ) return NULL
    }
}
```

Figure 8. Enqueue/dequeue throughput comparison of all queues.



wise, it repeats this process.

A dequeue, D , obtains a cell index h using FAA on head. It tries to atomically CAS the contents of $Q[h]$ from $\perp$ to another reserved value $\top$. This CAS fails if $Q[h]$ contained some $x \neq \perp$, in which case D returns x . Otherwise, the fact that D stored $\top$ in the cell guarantees an enqueue operation that later tries to store an item in $Q[h]$ will not succeed. D then returns NULL (indicating the queue is empty) if $\text{tail} \leq h + 1$ (the value of head following D 's FAA is $h + 1$). If D cannot return NULL, it repeats this process.

This algorithm can be shown to implement a FIFO queue correctly, but it has two major flaws that prevent it from being relevant in practice: using an infinite array and susceptibility to livelock (when a dequeuer continuously writes $\top$ into the cell an enqueue is about to access). The practical LCRQ algorithm addresses these flaws.

The infinite array is first collapsed to a concurrent ring (cyclic array) queue—CRQ for short—of R cells. The head and tail indices still strictly increase, but now the value of an index modulo R specifies the ring cell to which it points. Because now more than one enqueue and dequeuer can concurrently access a cell, the CRQ uses a more involved CAS-based protocol for synchronizing within each cell. This protocol enables an operation to avoid waiting for the completion of operations whose FAA returns smaller indices that also point to the same ring cell.

The CRQ's crucial performance property is that in the common fast path, an operation executes only one FAA instruction. The LCRQ algorithm then builds on the CRQ to prevent the livelock problem and handle the case of the CRQ filling up. The LCRQ is essentially a Michael and Scott linked list queue¹¹ in which each node is a CRQ. A CRQ that fills up or experiences livelock becomes closed to further enqueues, which instead append a new CRQ to the list and begin working in it. Most of the activity in the LCRQ therefore occurs in the individual CRQs, making contention (and CAS failures) on the list's head and tail a nonissue.

Performance. This section compares the LCRQ to Michael and Scott's classic lock-free queue,¹¹ as well as to the delegation-based variant presented

previously. The impact of CAS failures is explored by testing LCRQ-CAS, a version of LCRQ in which FAA is implemented with a CAS loop.

Figure 8a shows the results. LCRQ outperforms all other queues beyond two threads, achieving peak throughput of ≈ 40 million operations per second, or about 1,000 cycles per queue operation. From eight threads onward, LCRQ outperforms the delegation-based queue by 1.4 to 1.5 times and the MS (Michael and Scott) queue by more than three times. LCRQ-CAS matches LCRQ's performance up to four threads, but at that point its performance levels off. Subsequently, LCRQ-CAS exhibits the throughput “meltdown” associated with CAS failures. Similarly, the MS queue's performance peaks at two threads and degrades as concurrency increases.

Oversubscribed workloads can demonstrate the graceful behavior of lock-free algorithms under high load. In these workloads the number of software threads exceeds the hardware-supported level, forcing the operating system to context-switch between threads. If a thread holding a lock is preempted, a lock-based algorithm cannot make progress until it runs again. Indeed, as Figure 8b shows, when the number of threads exceeds 20, the throughput of the lock-based delegation algorithm plummets by 15 times, whereas both LCRQ and the MS queue maintain their peak throughput.

Conclusion

Advanced synchronization methods can boost the performance of shared mutable data structures. Synchronization still has its price, and when performance demands are extreme (or if the properties of centralized data structures are not needed), then distributed data structures are probably the right choice. For the many remaining cases, however, the methods described in this article can help build high-performance software. Awareness of these methods can assist those designing software for multicore machines. **C**

Q Related articles
on queue.acm.org

Nonblocking Algorithms and Scalable Multicore Programming

Samy Al Bahra

<http://queue.acm.org/detail.cfm?id=2492433>

Maximizing Power Efficiency with Asymmetric Multicore Systems

Alexandra Fedorova et al.

<http://queue.acm.org/detail.cfm?id=1658422>

Software and the Concurrency Revolution

Herb Sutter and James Larus

<http://queue.acm.org/detail.cfm?id=1095421>

References

1. Al Bahra, S. Nonblocking algorithms and scalable multicore programming. *Commun. ACM* 56, 7 (July 2013), 50–61.
2. Boyd-Wickizer, S., Frans Kaashoek, M., Morris, R. and Zeldovich, N. Non-scalable locks are dangerous. In *Proceedings of the Ottawa Linux Symposium*, 2012, 121–132.
3. Boyd-Wickizer, S., Frans Kaashoek, M., Morris, R. and Zeldovich, N. OpLog: A library for scaling update-heavy data structures. Technical Report MIT-CSAIL-TR2014-019, 2014.
4. Fatourou, P. and Kallimanis, N.D. Revisiting the combining synchronization technique. In *Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2012, 257–266.
5. Haas, A., Lippautz, M., Henzinger, T.A., Payer, H., Sokolova, A., Kirsch, C.M. and Sezgín, A. Distributed queues in shared memory: multicore performance and scalability through quantitative relaxation. In *Proceedings of the ACM International Conference on Computing Frontiers*, 2013, 17:1–17:9.
6. Hendler, D., Incze, I., Shavit, N., Tzafrir, M. Flat combining and the synchronization-parallelism trade-off. In *Proceedings of the 22nd ACM Symposium on Parallelism in Algorithms and Architectures*, 2010, 355–364.
7. Klaftenegger, D., Sagonas, K. and Winblad, K. Delegation locking libraries for improved performance of multithreaded programs. In *Proceedings of the 20th International European Conference on Parallel and Distributed Computing*, 2014, 572–583.
8. Kulkarni, M., Pingali, K., Walter, B., Ramanarayanan, G., Bala, K., Chew, L.P. Optimistic parallelism requires abstractions. In *Proceedings of the 2007 ACM SIGPLAN Conference on Programming Language Design and Implementation*, 211–222.
9. Lozi, J.-P., David, F., Thomas, G., Lawall, J. and Muller, G. Remote core locking: migrating critical section execution to improve the performance of multithreaded applications. In *Proceedings of the 2012 USENIX Annual Technical Conference*, 65–76.
10. Mellor-Crummey, J.M. and Scott, M.L. Algorithms for scalable synchronization on shared-memory multiprocessors. *ACM Trans. Computer Systems* 9, 1 (1991), 21–65.
11. Michael, M.M. and Scott, M.L. Simple, fast, and practical non-blocking and blocking concurrent queue algorithms. In *Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing*, 1996, 267–275.
12. Morrison, A. and Afek, Y. Fast concurrent queues for x86 processors. In *Proceedings of the 18th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2013, 103–112.
13. Oyama, Y., Taura, K. and Yonezawa, A. Executing parallel programs with synchronization bottlenecks efficiently. In *Proceedings of International Workshop on Parallel and Distributed Computing for Symbolic and Irregular Applications*, 1999, 182–204.
14. Shavit, N. Data structures in the multicore age. *Comm. ACM* 54, 3 (Mar. 2011), 76–84.
15. Treiber, R.K. Systems programming: coping with parallelism. Technical Report RJ5118 (2006). IBM Almaden Research Center.

Adam Morrison works on making parallel and distributed systems simpler to use without compromising their performance. He is an assistant professor at the Blavatnik School of Computer Science, Tel Aviv University, Israel.

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

Expert-curated guides to the best of CS research for practitioners.

BY PETER BAILIS, CAMILLE FOURNIER,
JOY ARULRAJ, AND ANDREW PAVLO

Research for Practice: Distributed Consensus and Implications of NVM on Database Management Systems

IN THIS INSTALLMENT of Research for Practice, we provide highlights from two critical areas in storage and large-scale services: distributed consensus and non-volatile memory.

First, how do large-scale distributed systems mediate access to shared resources, coordinate updates to mutable state, and reliably make decisions in the presence of failures? Camille Fournier, a seasoned and experienced distributed-systems builder (and ZooKeeper PMC), has curated a fantastic selection on distributed consensus in practice. The history of consensus echoes many of the goals of RfP: For decades the study and use of consensus protocols were considered notoriously difficult to understand

and remained primarily academic concerns. As a result, these protocols were largely ignored by industry. The rise of Internet-scale services and demands for automated solutions to cluster management, failover, and sharding in the 2000s finally led to the widespread practical adoption of these techniques. Adoption proved difficult, however, and the process in turn led to new (and ongoing) research on the subject. The papers in this selection highlight the challenges and the rewards of making the theory of consensus practical—both in theory and in practice.

Second, while consensus concerns *distributed* shared state, our second selection concerns the impact of hardware trends on *single-node* shared state. Joy Arulraj and Andy Pavlo provide a whirlwind tour of the implications of NVM (non-volatile memory) technologies on modern storage systems. NVM promises to overhaul the traditional paradigm that stable storage (that is, storage that persists despite failures) be block-addressable (that is, requires reading and writing in large chunks). In addition, NVM's performance characteristics lead to entirely different design trade-offs than conventional storage media such as spinning disks.

As a result, there is an arms race to rethink software storage-systems architectures to accommodate these new characteristics. This selection highlights projected implications for recovery subsystems, data-structure design, and data layout. While the first NVM devices have yet to make it to market, these pragmatically oriented citations from the literature hint at the volatile effects of non-volatile media on future storage systems.

I believe these two excellent contri-

>> about rfp

Research for Practice combines the resources of the ACM Digital Library, the largest collection of computer science research in the world, with the expertise of the ACM membership. In every RfP column two experts share a short, curated selection of papers on a concentrated, practically oriented topic.

butions fulfill RfP's goal of allowing you, the reader, to become an expert in a weekend afternoon's worth of reading. To facilitate this process, as always, we have provided open access to the ACM Digital Library for the relevant citations from these selections so you can enjoy these research results in their full glory. Keep on the lookout for our next installment, and please enjoy! —Peter Bailis

Peter Bailis is assistant professor of computer science at Stanford University. His research in the Future Data Systems group (<http://futuredata.stanford.edu/>) focuses on the design and implementation of next-generation data-intensive systems.



Distributed Consensus

By Camille Fournier

"A distributed system is one in which the failure of a computer you didn't know existed can render your own computer unusable."

—Leslie Lamport

As Lamport predicted in this quote, the real challenges of distributed computing—not just communicating via a network, but communicating to unknown nodes in a network—has greatly intensified in the past 15 years. With the incredible scaling of modern systems, “we have found ourselves in a world where answering the question, what is running where?” is increasingly difficult. Yet, we continue to have requirements that certain data never be lost and that certain actions behave in a consistent and predictable fashion, even when some nodes of the system may fail. To that end, there has been a rapid adoption of systems that rely on consensus protocols to guarantee this consistency in a widely distributed world.

The three papers included in this selection address the real world of consensus systems: Why are they needed? Why are they difficult to understand? What happens when you try to implement them? Is there an easier way, something that more developers can understand and therefore implement?

The first two papers discuss the reality of implementing Paxos-based consensus systems at Google, focusing first on the challenges of correctly implementing Paxos itself, and second on the challenges of creating a system based on a consensus algorithm that

provides useful functionality for developers. The final paper attempts to answer the question—Is there an easier way?—by introducing Raft, a consensus algorithm designed to be easier for developers to understand.

Theory Meets Reality

Chandra, T. D., Griesemer, R., Redstone, J. et al.
Paxos made live—An engineering perspective.

Proceedings of the 26th Annual ACM Symposium on Principles of Distributed Computing, 2007, 398–407.
<http://queue.acm.org/rfp/vol14iss3.html>

Paxos as originally stated is a page of pseudocode. The complete implementation of Paxos inside of Google's Chubby lock service is several thousand lines of C++. What happened? “Paxos Made Live” documents the evolution of the Paxos algorithm from theory into practice.

The basic idea of Paxos is to use voting by replicas with consistent storage to ensure that, even in the presence of failures, there can be unilateral consensus. This requires a coordinator be chosen, proposals sent and voted upon, and finally a commit recorded. Generally, systems record a series of these consensus values to a sequence log. This log-based variant is called multi-Paxos, which is less formally specified.

In creating a real system, durable logs are written to disks, which have finite capacity and are prone to corruption that must be detected and taken into account. The algorithm must be run on machines that can fail, and to make it operable at scale you need to be able to change group membership dynamically. While the system was expected to be fault tolerant, it also needed to perform quickly enough to be useful; otherwise, developers would work around it and create incorrect abstractions. The team details their efforts to ensure the core algorithm is expressed correctly and is testable, but even with these conscious efforts, the need for performance optimizations, concurrency, and multiple developers working on the project still means that the final system is ultimately an extended version of Paxos, which is difficult to prove correct.

Hell Is Other Programmers

Burrows, M.
The Chubby lock service for loosely coupled distributed systems.

Proceedings of the 7th Symposium on Operating Systems Design and Implementation, 2006, 335–350.
<http://queue.acm.org/rfp/vol14iss3.html>

While “Paxos Made Live” discusses the implementation of the consensus algorithm in detail, this paper about the Chubby lock service examines the overall system built around this algorithm. As research papers go, this one is a true delight for the practitioner. In particular, it describes designing a system and then evolving that design after it comes into contact with real-world usage. This paper should be required reading for anyone interested in designing and developing core infrastructure software that is to be offered as a service.

Burrows begins with a discussion of the design principles chosen as the basis for Chubby. Why make it a centralized service instead of a library? Why is it a lock service, and what kind of locking is it used for? Chubby not only provides locks, but also serves small files to facilitate sharing of metadata about distributed system state for its clients. Given that it is serving files, how many clients should Chubby expect to support, and what will that mean for the caching and change notification needs?

After discussing the details of the design, system structure, and API, Burrows gets into the nitty-gritty of the implementation. Building a highly sensitive centralized service for critical operations such as distributed locking and name resolution turns out to be quite difficult. Scaling the system to tens of thousands of clients meant being smart about caching and deploying proxies to handle some of the load. The developers misused and abused the system by accident, using features in unpredictable ways, attempting to use the system for large data storage or messaging. The Chubby maintainers resorted to reviewing other teams' planned uses of Chubby and denying access until review was satisfied. Through all of this we can see that the challenge in building a consensus system goes far beyond implementing a correct algorithm. We are still building a system and must think as carefully about its design and the users we will be supporting.

Can We Make This Easier?

Ongaro, D., Ousterhout, J.

In search of an understandable consensus algorithm.

Proceedings of the Usenix Annual Technical Conference, 2014, 305–320.

<https://www.usenix.org/conference/atc14/technical-sessions/presentation/ongaro>

Finally we come to the question, have we built ourselves into unnecessary complexity by taking it on faith that Paxos and its close cousins are the only way to implement consensus? What if there was an algorithm that we could also show to be correct but was designed to be easier for people to comprehend and implement correctly?

Raft is a consensus algorithm written for managing a replicated log but designed with the goal of making the algorithm itself more understandable than Paxos. This is done both by decomposing the problem into pieces that can be implemented and understood independently and by reducing the number of states that are valid for the system to hold.

Consensus is decomposed into issues of leader election, log replication, and safety. Leader election uses randomized election timeouts to reduce the likelihood of two candidates for leader splitting the vote and requiring a new round of elections. It allows candidates for leader to be elected only if they have the most up-to-date logs. This prevents the need for transferring data from follower to leader upon election. If a follower's log does not match the expected state for a new entry, the leader will replay entries from earlier in its log until it reaches a point at which the logs match, thus correcting the follower. This also means that a history of changes is stored in the logs, providing a side value of letting clients read (some) historical entries, should they desire.

The authors then show that after teaching a set of students both Paxos and Raft, the students were quizzed on their understanding of each and scored meaningfully higher on the Raft quiz. Looking around the current state of consensus systems in industry, we can see this play out in another way: namely, several new consensus systems have been created since 2014 based on Raft, where previously there

were very few reliable and successful open-source systems based on Paxos.

Bottlenecks, Single Points of Failure, and Consensus

Developers are often tempted to use a centralized consensus system to serve as the system of record for distributed coordination. Explicit coordination can make certain problems much easier to reason about and correct for; however, that puts the consensus system in the position of the bottleneck or critical point of failure for the other systems that rely on it to make progress. As we can see from these papers, making a centralized consensus system production-ready can come at the cost of adding optimizations and recovery mechanisms that were not dreamed of in the original Paxos literature.

What is the way forward? Arguably, writing systems that do not rely on centralized consensus brokers to operate safely would be the best option, but we are still in the early days of coordination-avoidance research and development. While we wait for more evolution on that front, Raft provides an interesting alternative, an algorithm designed for readability and general understanding. The impact of having an easier algorithm to implement is already being felt, as far more developers are embedding Raft within distributed systems and building specifically tailored Raft-based coordination brokers. Consensus remains a tricky problem—but one that is finally seeing a diversity of approaches to reaching a solution.

Camille Fournier is a writer, speaker, and entrepreneur. Formerly the CTO of Rent the Runway, she serves on the technical oversight committee for the Cloud Native Computing Foundation, as a Project Management Committee member of the Apache ZooKeeper project, and a project overseer of the Dropwizard Web framework.



Implications of NVM on Database Management Systems

By Joy Arulraj and Andrew Pavlo



The advent of non-volatile memory (NVM) will fundamentally change the dichotomy between memory and durable storage in a database management system (DBMS). NVM is a broad class of tech-

nologies—including phase-change memory, memristors, and STT-MRAM (spin-transfer torque-magnetoresistive random-access memory)—that provide low-latency reads and writes on the same order of magnitude as DRAM (dynamic random-access memory), but with persistent writes and large storage capacity like an SSD (solid-state drive). Unlike DRAM, writes to NVM are expected to be more expensive than reads. These devices also have limited write endurance, which necessitates fewer writes and wear-leveling to increase their lifetimes.

The first NVM devices released will have the same form factor and block-oriented access as today's SSDs. Thus, today's DBMSs will use this type of NVM as a faster drop-in replacement for their current storage hardware.

By the end of this decade, however, NVM devices will support byte-addressable access akin to DRAM. This will require additional CPU architecture and operating-system support for persistent memory. This also means that existing DBMSs are unable to take full advantage of NVM because their internal architectures are predicated on the assumption that memory is volatile. With NVM, many of the components of legacy DBMSs are unnecessary and will degrade the performance of data-intensive applications.

We have selected three papers that focus on how the emergence of byte-addressable NVM technologies will impact the design of DBMS architectures. The first two present new abstractions for performing durable atomic updates on an NVM-resident database and recovery protocols for an NVM DBMS. The third paper addresses the write-endurance limitations of NVM by introducing a collection of write-limited query-processing algorithms. Thus, this selection contains novel ideas that can help leverage the unique set of attributes of NVM devices for delivering the features required by modern data-management applications. The common theme for these papers is that you cannot just run an existing DBMS on NVM and expect it to leverage its unique set of properties. The only way to achieve that is to come up with novel architectures, protocols, and algorithms that are tailor-made for NVM.

ARIES Redesigned for NVM

Coburn, J., et al.

From ARIES to MARS: Transaction support for next-generation, solid-state drives.

Proceedings of the 24th ACM Symposium on Operating Systems Principles, 2013, 197–212.
<http://queue.acm.org/rfp/vol14iss3.html>

ARIES is considered the standard for recovery protocols in a transactional DBMS. It has two key goals: first, it provides an interface for supporting scalable ACID (atomicity, consistency, isolation, durability) transactions; second, it maximizes performance on disk-based storage systems. In this paper, the authors focus on how ARIES should be adapted for NVM-based storage.

Since random writes to the disk whenever a transaction updates the database obviously decrease performance, ARIES requires that the DBMS first record a log entry in the write-ahead log (a sequential write) before updating the database itself (a random write). It adopts a *no-force* policy wherein the updates are written to the database lazily after the transaction commits. Such a policy assumes that sequential writes to non-volatile storage are significantly faster than random writes. The authors, however, demonstrate that this is no longer the case with NVM.

The MARS protocol proposes a new hardware-assisted logging primitive that combines multiple writes to arbitrary storage locations into a single atomic operation. By leveraging this primitive, MARS eliminates the need for an ARIES-style undo log and relies on the NVM device to apply the redo log at commit time. We are particularly fond of this paper because it helps in better appreciating the intricacies involved in designing the recovery protocol in a DBMS for guarding against data loss.

Near-Instantaneous Recovery Protocols

Arulraj, J., Pavlo, A., Dullloor, S.R.

Let's talk about storage and recovery methods for non-volatile memory database systems.

Proceedings of the ACM SIGMOD International Conference on Management of Data, 2015, 707–722.
<http://queue.acm.org/rfp/vol14iss3.html>

This paper takes a different approach to performing durable atomic up-

dates on an NVM-resident database than the previous paper. In ARIES, during recovery the DBMS first loads the most recent snapshot. It then replays the redo log to ensure that all the updates made by committed transactions are recovered. Finally, it uses the undo log to ensure that the changes made by incomplete transactions are not present in the database. This recovery process can take a lot of time, depending on the load on the system and the frequency with which snapshots are taken. Thus, this paper explores whether it is possible to leverage NVM's properties to speed up recovery from system failures.

The authors present a software-based primitive called *non-volatile pointer*. When a pointer points to data residing on NVM, and is itself stored on NVM, then it will remain valid even after the system recovers from a power failure. Using this primitive, the authors design a library of non-volatile data structures that support durable atomic updates. They propose a recovery protocol that, in contrast to MARS, obviates the need for an ARIES-style redo log. This enables the system to skip replaying the redo log, and thereby allows the NVM DBMS to recover the database almost instantaneously.

Both papers propose recovery protocols that target an NVM-only storage hierarchy. The generalization of these protocols to a multitier storage hierarchy with both DRAM and NVM is a hot topic in research today.

Trading Expensive Writes for Cheaper Reads

Viglas, S.D.

Write-limited sorts and joins for persistent memory.

Proceedings of the VLDB Endowment 7, 5 (2014), 413–424.
<http://www.vldb.org/pvldb/vol7/p413-viglas.pdf>

The third paper focuses on the higher write costs and limited write-endurance problems of NVM. For several decades algorithms have been designed for the random-access machine model where reads and writes have the same cost. The emergence of NVM devices, where writes are more expensive than reads, opens up the design space for new write-limiting algorithms. It will be fascinating to see researchers derive

new bounds on the number of writes that different kinds of query-processing algorithms must perform.

Viglas presents a collection of novel query-processing algorithms that minimize I/O by trading off expensive NVM writes for cheaper reads. One such algorithm is the *segment sort*. The basic idea is to use a combination of two sorting algorithms—external merge sort and selection sort—that splits the input into two segments that are then processed using a different algorithm. The selection-sort algorithm uses extra reads, and writes out each element in the input only once at its final location. By using a combination of these two algorithms, the DBMS can optimize both the performance and the number of NVM writes.

Game Changer for DBMS Architectures

NVM is a definite game changer for future DBMS architectures. It will require system designers to rethink many of the core algorithms and techniques developed over the past 40 years. Using these new storage devices in the manner prescribed by these papers will allow DBMSs to achieve better performance than what is possible with today's hardware for write-heavy database applications. This is because these techniques are designed to exploit the low-latency read/writes of NVM to enable a DBMS to store less redundant data and incur fewer writes. Furthermore, we contend that existing in-memory DBMSs are better positioned to use NVM when it is finally available. This is because these systems are already designed for byte-addressable access methods, whereas legacy disk-oriented DBMSs will require laborious and costly overhauls in order to use NVM correctly, as described in these papers. Word is bond. 

Joy Arulraj is a Ph.D. candidate at Carnegie Mellon University. He is interested in the design and implementation of next-generation database management systems.

Andy Pavlo is an assistant professor of databaseology in the Department of Computer Science at Carnegie Mellon University, Pittsburgh, PA.

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

DOI:10.1145/2934664

This open source computing framework unifies streaming, batch, and interactive big data workloads to unlock new applications.

BY MATEI ZAHARIA, REYNOLD S. XIN, PATRICK WENDELL, TATHAGATA DAS, MICHAEL ARMBRUST, ANKUR DAVE, XIANGRUI MENG, JOSH ROSEN, SHIVARAM VENKATARAMAN, MICHAEL J. FRANKLIN, ALI GHODSI, JOSEPH GONZALEZ, SCOTT SHENKER, AND ION STOICA

Apache Spark: A Unified Engine for Big Data Processing

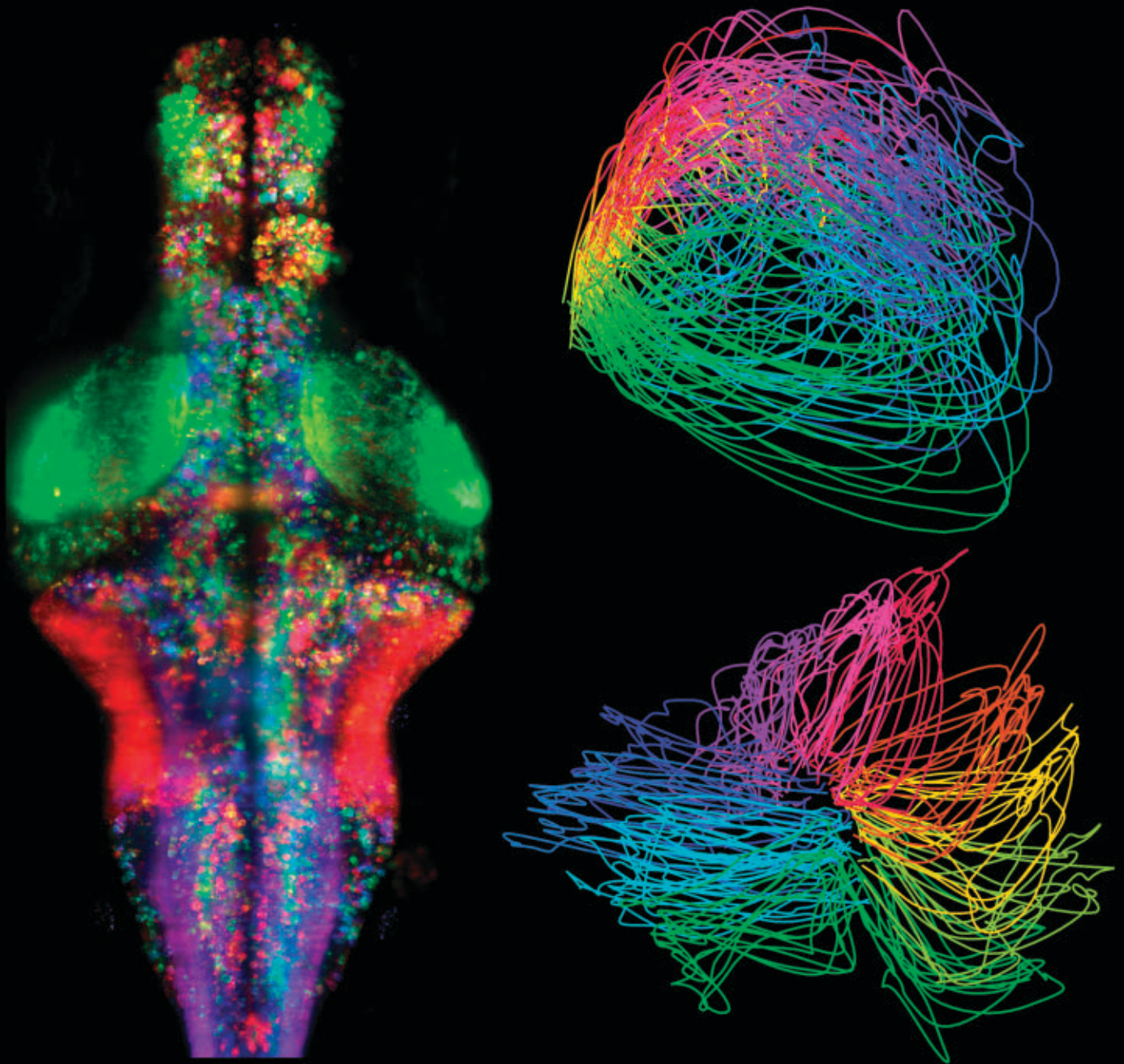
THE GROWTH OF data volumes in industry and research poses tremendous opportunities, as well as tremendous computational challenges. As data sizes have outpaced the capabilities of single machines, users have needed new systems to scale out computations to multiple nodes. As a result, there has been an explosion of new cluster programming models targeting diverse computing workloads.^{1,4,7,10} At first, these models were relatively specialized, with new models developed for new workloads; for example, MapReduce⁴ supported batch processing, but Google also developed Dremel¹³

for interactive SQL queries and Pregel¹¹ for iterative graph algorithms. In the open source Apache Hadoop stack, systems like Storm¹ and Impala⁹ are also specialized. Even in the relational database world, the trend has been to move away from “one-size-fits-all” systems.¹⁸ Unfortunately, most big data applications need to combine many different processing types. The very nature of “big data” is that it is diverse and messy; a typical pipeline will need MapReduce-like code for data loading, SQL-like queries, and iterative machine learning. Specialized engines can thus create both complexity and inefficiency; users must stitch together disparate systems, and some applications simply cannot be expressed efficiently in any engine.

In 2009, our group at the University of California, Berkeley, started the Apache Spark project to design a unified engine for distributed data processing. Spark has a programming model similar to MapReduce but extends it with a data-sharing abstraction called “Resilient Distributed Datasets,” or RDDs.²⁵ Using this simple extension, Spark can capture a wide range of processing workloads that previously needed separate engines, including SQL, streaming, machine learning, and graph processing^{2,26,6} (see Figure 1). These implementations use the same optimizations as specialized engines (such as column-oriented processing and incremental updates) and achieve similar performance but run as libraries over a common engine, making them easy and efficient to compose. Rather than being specific

» key insights

- A simple programming model can capture streaming, batch, and interactive workloads and enable new applications that combine them.
- Apache Spark applications range from finance to scientific data processing and combine libraries for SQL, machine learning, and graphs.
- In six years, Apache Spark has grown to 1,000 contributors and thousands of deployments.



Analyses performed using Spark of brain activity in a larval zebrafish: (left) matrix factorization to characterize functionally similar regions (as depicted by different colors) and (right) embedding dynamics of whole-brain activity into lower-dimensional trajectories. Source: Jeremy Freeman and Misha Ahrens, Janelia Research Campus, Howard Hughes Medical Institute, Ashburn, VA.

to these workloads, we claim this result is more general; when augmented with data sharing, MapReduce can emulate any distributed computation, so it should also be possible to run many other types of workloads.²⁴

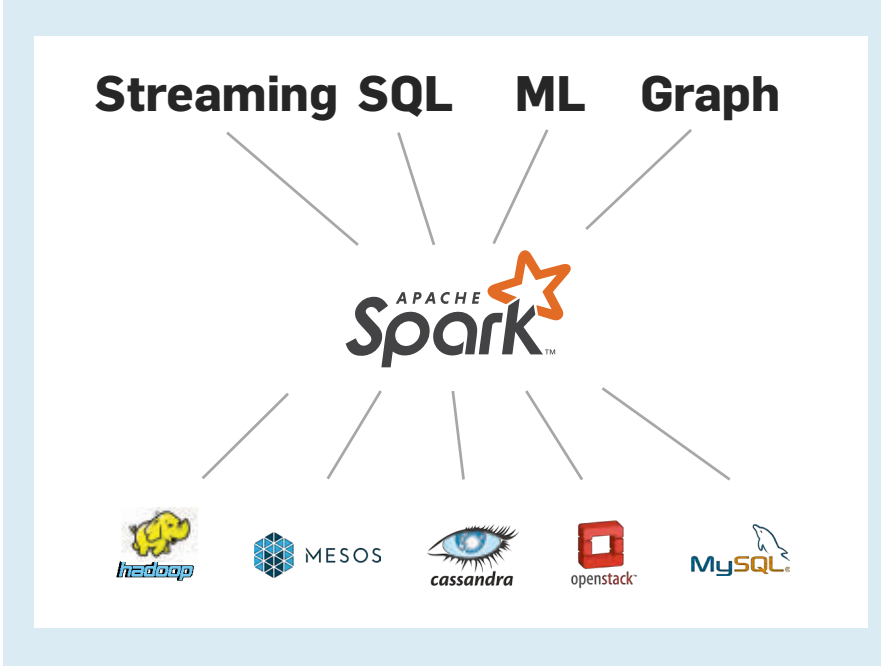
Spark's generality has several important benefits. First, applications are easier to develop because they use a unified API. Second, it is more efficient to combine processing tasks; whereas prior systems required writing the data to storage to pass it to another en-

gine, Spark can run diverse functions over the same data, often in memory. Finally, Spark enables new applications (such as interactive queries on a graph and streaming machine learning) that were not possible with previous systems. One powerful analogy for the value of unification is to compare smartphones to the separate portable devices that existed before them (such as cameras, cellphones, and GPS gadgets). In unifying the functions of these devices, smartphones enabled new

applications that combine their functions (such as video messaging and Waze) that would not have been possible on any one device.

Since its release in 2010, Spark has grown to be the most active open source project or big data processing, with more than 1,000 contributors. The project is in use in more than 1,000 organizations, ranging from technology companies to banking, retail, biotechnology, and astronomy. The largest publicly announced deployment has

Figure 1. Apache Spark software stack, with specialized processing libraries implemented over the core engine.



more than 8,000 nodes.²² As Spark has grown, we have sought to keep building on its strength as a unified engine. We (and others) have continued to build an integrated standard library over Spark, with functions from data import to machine learning. Users find this ability powerful; in surveys, we find the majority of users combine multiple of Spark's libraries in their applications.

As parallel data processing becomes common, the composability of processing functions will be one of the most important concerns for both usability and performance. Much of data analysis is exploratory, with users wishing to combine library functions quickly into a working pipeline. However, for “big data” in particular, copying data between different systems is anathema to performance. Users thus need abstractions that are general and composable. In this article, we introduce the Spark programming model and explain why it is highly general. We also discuss how we leveraged this generality to build other processing tasks over it. Finally, we summarize Spark's most common applications and describe ongoing development work in the project.

Programming Model

The key programming abstraction in Spark is RDDs, which are fault-tolerant collections of objects partitioned

across a cluster that can be manipulated in parallel. Users create RDDs by applying operations called “transformations” (such as `map`, `filter`, and `groupBy`) to their data.

Spark exposes RDDs through a functional programming API in Scala, Java, Python, and R, where users can simply pass local functions to run on the cluster. For example, the following Scala code creates an RDD representing the error messages in a log file, by searching for lines that start with `ERROR`, and then prints the total number of errors:

```
lines = spark.textFile("hdfs://...")
errors = lines.filter(
  s => s.startsWith("ERROR"))
println("Total errors: " + errors.count())
```

The first line defines an RDD backed by a file in the Hadoop Distributed File System (HDFS) as a collection of lines of text. The second line calls the `filter` transformation to derive a new RDD from `lines`. Its argument is a Scala function literal or closure.^a Finally, the last line calls `count`, another type of RDD operation called an “action” that

^a The closures passed to Spark can call into any existing Scala or Python library or even reference variables in the outer program. Spark sends read-only copies of these variables to worker nodes.

returns a result to the program (here, the number of elements in the RDD) instead of defining a new RDD.

Spark evaluates RDDs lazily, allowing it to find an efficient plan for the user's computation. In particular, transformations return a new RDD object representing the result of a computation but do not immediately compute it. When an action is called, Spark looks at the whole graph of transformations used to create an execution plan. For example, if there were multiple `filter` or `map` operations in a row, Spark can fuse them into one pass, or, if it knows that data is partitioned, it can avoid moving it over the network for `groupBy`.⁵ Users can thus build up programs modularly without losing performance.

Finally, RDDs provide explicit support for data sharing among computations. By default, RDDs are “ephemeral” in that they get recomputed each time they are used in an action (such as `count`). However, users can also persist selected RDDs in memory or for rapid reuse. (If the data does not fit in memory, Spark will also spill it to disk.) For example, a user searching through a large set of log files in HDFS to debug a problem might load just the error messages into memory across the cluster by calling

```
errors.persist()
```

After this, the user can run a variety of queries on the in-memory data:

```
// Count errors mentioning MySQL
errors.filter(s => s.contains("MySQL"))
    .count()
// Fetch back the time fields of errors that
// mention PHP, assuming time is field #3:
errors.filter(s => s.contains("PHP"))
    .map(line => line.split('\t')(3))
    .collect()
```

This data sharing is the main difference between Spark and previous computing models like MapReduce; otherwise, the individual operations (such as `map` and `groupBy`) are similar. Data sharing provides large speedups, often as much as 100×, for interactive queries and iterative algorithms.²³ It is also the key to Spark's generality, as we discuss later.

Fault tolerance. Apart from providing data sharing and a variety of paral-

lel operations, RDDs also automatically recover from failures. Traditionally, distributed computing systems have provided fault tolerance through data replication or checkpointing. Spark uses a different approach called “lineage.”²⁵ Each RDD tracks the graph of transformations that was used to build it and reruns these operations on base data to reconstruct any lost partitions. For example, Figure 2 shows the RDDs in our previous query, where we obtain the time fields of errors mentioning PHP by applying two filters and a map. If any partition of an RDD is lost (for example, if a node holding an in-memory partition of errors fails), Spark will rebuild it by applying the filter on the corresponding block of the HDFS file. For “shuffle” operations that send data from all nodes to all other nodes (such as `reduceByKey`), senders persist their output data locally in case a receiver fails.

Lineage-based recovery is significantly more efficient than replication in data-intensive workloads. It saves both time, because writing data over the network is much slower than writing it to RAM, and storage space in memory. Recovery is typically much faster than simply rerunning the program, because a failed node usually contains multiple RDD partitions, and these partitions can be rebuilt in parallel on other nodes.

A longer example. As a longer example, Figure 3 shows an implementation of logistic regression in Spark. It uses batch gradient descent, a simple iterative algorithm that computes a gradient function over the data repeatedly as a parallel sum. Spark makes it easy to load the data into RAM once and run multiple sums. As a result, it runs faster than traditional MapReduce. For example, in a 100GB job (see Figure 4), MapReduce takes 110 seconds per iteration because each iteration loads the data from disk, while Spark takes only one second per iteration after the first load.

Integration with storage systems. Much like Google’s MapReduce, Spark is designed to be used with multiple external systems for persistent storage. Spark is most commonly used with cluster file systems like HDFS and key-value stores like S3 and Cassandra. It can also connect with Apache Hive as a data catalog.

RDDs usually store only temporary data within an application, though some applications (such as the Spark SQL JDBC server) also share RDDs across multiple users.² Spark’s design as a storage-system-agnostic engine makes it easy for users to run computations against existing data and join diverse data sources.

Higher-Level Libraries

The RDD programming model provides only distributed collections of objects and functions to run on them. Using RDDs, however, we have built a variety of higher-level libraries on Spark, targeting many of the use cases of specialized computing engines. The key idea is that if we control the data structures stored inside RDDs, the partitioning of data across nodes, and the functions run on them, we can implement many of the execution techniques in other engines. Indeed, as we show in this section, these libraries often achieve state-of-the-art performance on each task while offering significant benefits when users combine them. We now discuss the four main libraries included with Apache Spark.

SQL and DataFrames. One of the most common data processing paradigms is relational queries. Spark SQL² and its predecessor, Shark,²³ implement such queries on Spark, using techniques similar to analytical databases. For example, these systems support columnar storage, cost-based optimization, and code generation for query execution. The main idea behind these systems is to use the same data layout as analytical databases—compressed columnar storage—inside RDDs. In Spark SQL, each record in an RDD holds a series of rows stored in binary format, and the system generates

Figure 2. Lineage graph for the third query in our example; boxes represent RDDs, and arrows represent transformations.

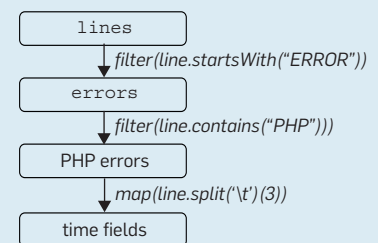


Figure 3. A Scala implementation of logistic regression via batch gradient descent in Spark.

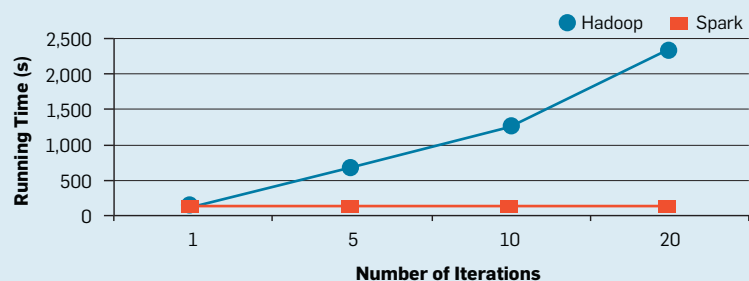
```

// Load data into an RDD
val points = sc.textFile(...).map(readPoint).persist()

// Start with a random parameter vector
var w = DenseVector.random(D)

// On each iteration, update param vector with a sum
for (i <- 1 to ITERATIONS) {
  val gradient = points.map { p =>
    p.x * (1 / (1 + exp(-p.y * (w.dot(p.x)))) - 1) * p.y
  }.reduce((a, b) => a + b)
  w -= gradient
}
  
```

Figure 4. Performance of logistic regression in Hadoop MapReduce vs. Spark for 100GB of data on 50 m2.4xlarge EC2 nodes.



code to run directly against this layout.

Beyond running SQL queries, we have used the Spark SQL engine to provide a higher-level abstraction for basic data transformations called DataFrames,² which are RDDs of records with a known schema. DataFrames are a common abstraction for tabular data in R and Python, with programmatic methods for filtering, computing new columns, and aggregation. In Spark, these operations map down to the Spark SQL engine and receive all its optimizations. We discuss DataFrames more later.

One technique not yet implemented in Spark SQL is indexing, though other libraries over Spark (such as IndexeDRDDs³) do use it.

Spark Streaming. Spark Streaming²⁶ implements incremental stream processing using a model called “discretized streams.” To implement streaming over Spark, we split the input data into small batches (such as every 200 milliseconds) that we regularly combine with state stored inside RDDs to produce new results. Running streaming computations this way has several benefits over traditional distributed streaming systems. For example, fault recovery is less expensive due to using lineage, and it is possible to combine streaming with batch and interactive queries.

GraphX. GraphX⁶ provides a graph computation interface similar to Pregel and GraphLab,^{10,11} implementing the same placement optimizations as these systems (such as vertex partitioning schemes) through its choice of partitioning function for the RDDs it builds.

MLlib. MLlib,¹⁴ Spark’s machine learning library, implements more than 50 common algorithms for distributed model training. For example, it includes the common distributed algorithms of decision trees (PLANET), Latent Dirichlet Allocation, and Alternating Least Squares matrix factorization.

Combining processing tasks. Spark’s libraries all operate on RDDs as the data abstraction, making them easy to combine in applications. For example, Figure 5 shows a program that reads some historical Twitter data using Spark SQL, trains a K-means clustering model using MLlib, and then applies the model to a new stream of tweets. The data tasks returned by each library (here the historic tweet RDD and the K-



Spark has a similar programming model to MapReduce but extends it with a data-sharing abstraction called “resilient distributed datasets,” or RDDs.



means model) are easily passed to other libraries. Apart from compatibility at the API level, composition in Spark is also efficient at the execution level, because Spark can optimize *across* processing libraries. For example, if one library runs a map function and the next library runs a map on its result, Spark will fuse these operations into a single map. Likewise, Spark’s fault recovery works seamlessly across these libraries, recomputing lost data no matter which libraries produced it.

Performance. Given that these libraries run over the same engine, do they lose performance? We found that by implementing the optimizations we just outlined within RDDs, we can often match the performance of specialized engines. For example, Figure 6 compares Spark’s performance on three simple tasks—a SQL query, streaming word count, and Alternating Least Squares matrix factorization—versus other engines. While the results vary across workloads, Spark is generally comparable with specialized systems like Storm, GraphLab, and Impala.^b For stream processing, although we show results from a distributed implementation on Storm, the per-node throughput is also comparable to commercial streaming engines like Oracle CEP.²⁶

Even in highly competitive benchmarks, we have achieved state-of-the-art performance using Apache Spark. In 2014, we entered the Daytona GraySort benchmark (<http://sortbenchmark.org/>) involving sorting 100TB of data on disk, and tied for a new record with a specialized system built only for sorting on a similar number of machines. As in the other examples, this was possible because we could implement both the communication and CPU optimizations necessary for large-scale sorting inside the RDD model.

Applications

Apache Spark is used in a wide range of applications. Our surveys of Spark

^b One area in which other designs have outperformed Spark is certain graph computations.^{12,16} However, these results are for algorithms with low ratios of computation to communication (such as PageRank) where the latency from synchronized communication in Spark is significant. In applications with more computation (such as the ALS algorithm) distributing the application on Spark still helps.

users have identified more than 1,000 companies using Spark, in areas from Web services to biotechnology to finance. In academia, we have also seen applications in several scientific domains. Across these workloads, we find users take advantage of Spark's generality and often combine multiple of its libraries. Here, we cover a few top use cases. Presentations on many use cases are also available on the Spark Summit conference website (<http://www.spark-summit.org>).

Batch processing. Spark's most common applications are for batch processing on large datasets, including Extract-Transform-Load workloads to convert data from a raw format (such as log files) to a more structured format and offline training of machine learning models. Published examples of these workloads include page personalization and recommendation at Yahoo!; managing a data lake at Goldman Sachs; graph mining at Alibaba; financial Value at Risk calculation; and text mining of customer feedback at Toyota. The largest published use case we are aware of is an 8,000-node cluster at Chinese social network Tencent that ingests 1PB of data per day.²²

While Spark can process data in memory, many of the applications in this category run only on disk. In such cases, Spark can still improve performance over MapReduce due to its support for more complex operator graphs.

Interactive queries. Interactive use of Spark falls into three main classes. First, organizations use Spark SQL for relational queries, often through business-intelligence tools like Tableau. Examples include eBay and Baidu. Second, developers and data scientists can use Spark's Scala, Python, and R interfaces interactively through shells or visual notebook environments. Such interactive use is crucial for asking more advanced questions and for designing models that eventually lead to production applications and is common in all deployments. Third, several vendors have developed domain-specific interactive applications that run on Spark. Examples include Tresata (anti-money laundering), Trifacta (data cleaning), and PanTera (large-scale visualization, as in Figure 7).

Stream processing. Real-time processing is also a popular use case, both in analytics and in real-time decision-

making applications. Published use cases for Spark Streaming include network security monitoring at Cisco, prescriptive analytics at Samsung SDS, and log mining at Netflix. Many of these applications also combine

streaming with batch and interactive queries. For example, video company Conviva uses Spark to continuously maintain a model of content distribution server performance, querying it automatically when it moves clients

Figure 5. Example combining the SQL, machine learning, and streaming libraries in Spark.

```
// Load historical data as an RDD using Spark SQL
val trainingData = sql(
  "SELECT location, language FROM old_tweets")

// Train a K-means model using MLlib
val model = new KMeans()
  .setFeaturesCol("location")
  .setPredictionCol("language")
  .fit(trainingData)

// Apply the model to new tweets in a stream
TwitterUtils.createStream(...)
  .map(tweet => model.predict(tweet.location))
```

Figure 6. Comparing Spark's performance with several widely used specialized systems for SQL, streaming, and machine learning. Data is from Zaharia²⁴ (SQL query and streaming word count) and Sparks et al.¹⁷ (alternating least squares matrix factorization).

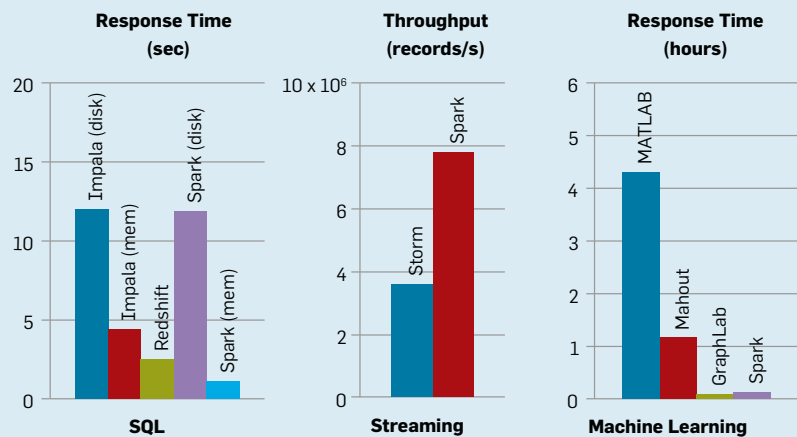
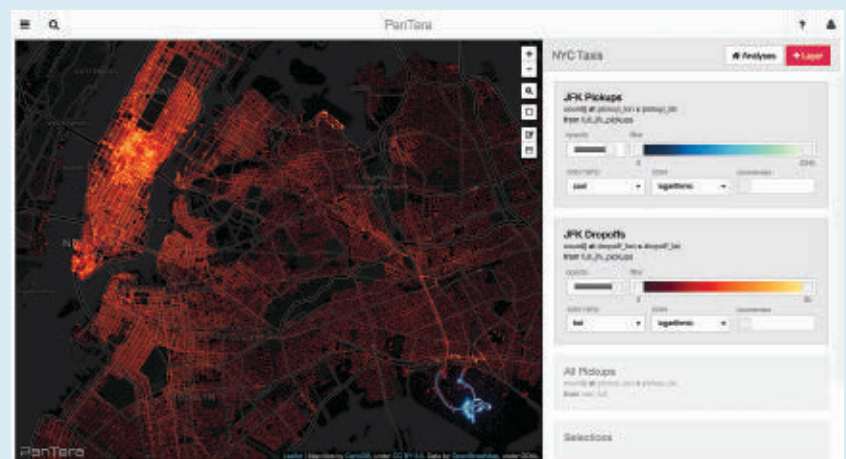


Figure 7. PanTera, a visualization application built on Spark that can interactively filter data.



Source: PanTera

across servers, in an application that requires substantial parallel work for both model maintenance and queries.

Scientific applications. Spark has also been used in several scientific domains, including large-scale spam detection,¹⁹ image processing,²⁷ and genomic data processing.¹⁵ One example that combines batch, interactive, and stream processing is the Thunder platform for neuroscience at Howard Hughes Medical Institute, Janelia Farm.⁵ It is designed to process brain-imaging data from experiments in real time, scaling up to 1TB/hour of whole-brain imaging data from organisms (such as zebrafish and mice). Using Thunder, researchers can apply machine learning algorithms (such as clustering and Principal Component Analysis) to identify neurons involved in specific behaviors. The same code can be run in batch jobs on data from previous runs or in interactive

queries during live experiments. Figure 8 shows an example image generated using Spark.

Spark components used. Because Spark is a unified data-processing engine, the natural question is how many of its libraries organizations actually use. Our surveys of Spark users have shown that organizations do, indeed, use multiple components, with over 60% of organizations using at least three of Spark's APIs. Figure 9 outlines the usage of each component in a July 2015 Spark survey by Databricks that reached 1,400 respondents. We list the Spark Core API (just RDDs) as one component and the higher-level libraries as others. We see that many components are widely used, with Spark Core and SQL as the most popular. Streaming is used in 46% of organizations and machine learning in 54%. While not shown directly in

Figure 9, most organizations use multiple components; 88% use at least two of them, 60% use at least three (such as Spark Core and two libraries), and 27% use at least four components.

Deployment environments. We also see growing diversity in where Apache Spark applications run and what data sources they connect to. While the first Spark deployments were generally in Hadoop environments, only 40% of deployments in our July 2015 Spark survey were on the Hadoop YARN cluster manager. In addition, 52% of respondents ran Spark on a public cloud.

Why Is the Spark Model General?

While Apache Spark demonstrates that a unified cluster programming model is both feasible and useful, it would be helpful to understand what makes cluster programming models general, along with Spark's limitations. Here, we summarize a discussion on the generality of RDDs from Zaharia.²⁴ We study RDDs from two perspectives. First, from an expressiveness point of view, we argue that RDDs can emulate any distributed computation, and will do so efficiently in many cases unless the computation is sensitive to network latency. Second, from a systems point of view, we show that RDDs give applications control over the most common bottleneck resources in clusters—network and storage I/O—and thus make it possible to express the same optimizations for these resources that characterize specialized systems.

Expressiveness perspective. To study the expressiveness of RDDs, we start by comparing RDDs to the MapReduce model, which RDDs build on. The first question is what computations can MapReduce itself express? Although there have been numerous discussions about the limitations of MapReduce, the surprising answer here is that MapReduce can emulate any distributed computation.

To see this, note that any distributed computation consists of nodes that perform local computation and occasionally exchange messages. MapReduce offers the map operation, which allows local computation, and reduce, which allows all-to-all communication. Any distributed computation can thus be emulated, perhaps somewhat inefficiently, by breaking down its work into timesteps,

Figure 8. Visualization of neurons in the zebrafish brain created with Spark, where each neuron is colored based on the direction of movement that correlates with its activity. Source: Jeremy Freeman and Misha Ahrens of Janelia Research Campus.

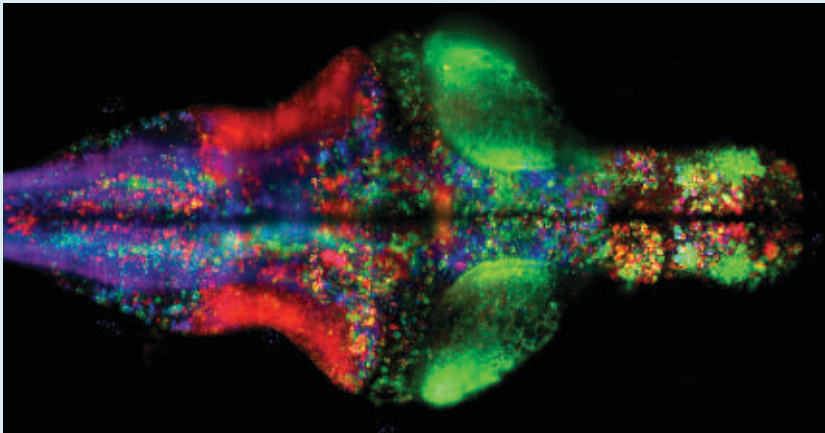
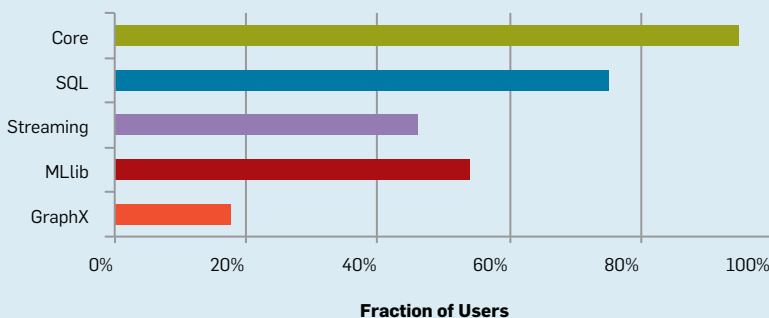


Figure 9. Percent of organizations using each Spark component, from the Databricks 2015 Spark survey; <https://databricks.com/blog/2015/09/24/>.



running maps to perform the local computation in each timestep, and batching and exchanging messages at the end of each step using a reduce. A series of MapReduce steps will capture the whole result, as in Figure 10. Recent theoretical work has formalized this type of emulation by showing that MapReduce can simulate many computations in the Parallel Random Access Machine model.⁸ Repeated MapReduce is also equivalent to the Bulk Synchronous Parallel model.²⁰

While this line of work shows that MapReduce can emulate arbitrary computations, two problems can make the “constant factor” behind this emulation high. First, MapReduce is inefficient at sharing data across timesteps because it relies on replicated external storage systems for this purpose. Our emulated system may thus become slower due to writing out its state after each step. Second, the latency of the MapReduce steps determines how well our emulation will match a real network, and most MapReduce implementations were designed for batch environments with minutes to hours of latency.

RDDs and Spark address both of these limitations. On the data-sharing front, RDDs make data sharing fast by avoiding replication of intermediate data and can closely emulate the in-memory “data sharing” across time that would happen in a system composed of long-running processes. On the latency front, Spark can run MapReduce-like steps on large clusters with 100ms latency; nothing intrinsic to the MapReduce model prevents this. While some applications need finer-grain timesteps and communication, this 100ms latency is enough to implement many data-intensive workloads, where the amount of computation that can be batched before a communication step is high.

In summary, RDDs build on MapReduce’s ability to emulate any distributed computation but make this emulation significantly more efficient. Their main limitation is increased latency due to synchronization in each communication step, but this latency is often not a factor.

Systems perspective. Independent of the emulation approach to characterizing Spark’s generality, we can take a systems approach. What are the

bottleneck resources in cluster computations? And can RDDs use them efficiently? Although cluster applications are diverse, they are all bound by the same properties of the underlying hardware. Current datacenters have a steep storage hierarchy that limits most applications in similar ways. For example, a typical Hadoop cluster might have the following characteristics:

Local storage. Each node has local memory with approximately 50GB/s of bandwidth, as well as 10 to 20 local disks, for approximately 1GB/s to 2GB/s of disk bandwidth;

Links. Each node has a 10Gbps (1.3GB/s) link, or approximately 40× less than its memory bandwidth and 2× less than its aggregate disk bandwidth; and

Racks. Nodes are organized into racks of 20 to 40 machines, with 40Gbps–80Gbps bandwidth out of each rack, or 2×–5× lower than the in-rack network performance.

Given these properties, the most important performance concern in many applications is the placement of data and computation in the network. Fortunately, RDDs provide the facili-

Figure 10. Emulating an arbitrary distributed computation with MapReduce.

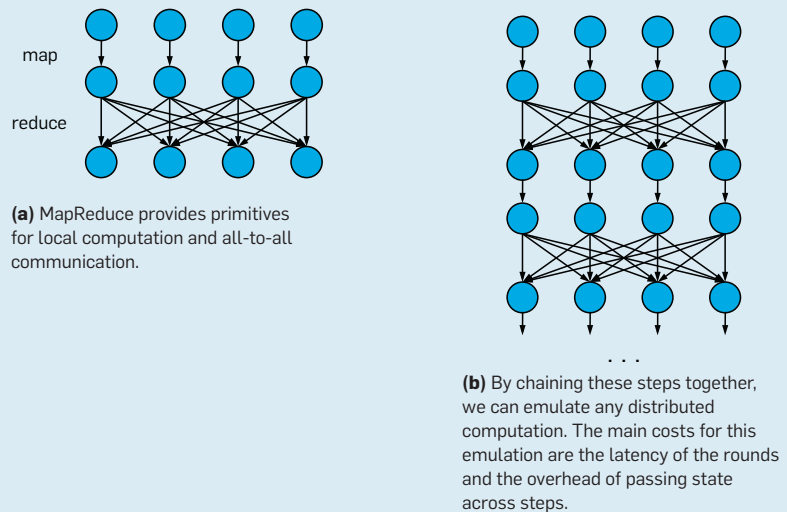


Figure 11. Example of Spark’s DataFrame API in Python. Unlike Spark’s core API, DataFrames have a schema with named columns (such as age and city) and take expressions in a limited language (such as `age > 20`) instead of arbitrary Python functions.

```
users.where(users["age"] > 20)
      .groupBy("city")
      .agg(avg("age"), max("income"))
```

Figure 12. Working with DataFrames in Spark’s R API. We load a distributed DataFrame using Spark’s JSON data source, then filter and aggregate using standard R column expressions.

```
people <- read.df(context, "./people.json", "json")

# Filter people by age
adults = filter(people, people$age > 20)

# Count number of people by country
summarize(groupBy(adults, adults$city), count=n(adults$id))
##      city      count
##1  Cambridge      1
##2 San Francisco    6
##3   Berkeley     4
```

ties to control this placement; the interface lets applications place computations near input data (through an API for “preferred locations” for input sources²⁵), and RDDs provide control over data partitioning and co-location (such as specifying that data be hashed by a given key). Libraries (such as GraphX) can thus implement the same placement strategies used in specialized systems.⁶

Beyond network and I/O bandwidth, the most common bottleneck tends to be CPU time, especially if data is in memory. In this case, however, Spark can run the same algorithms and libraries used in specialized systems on each node. For example, it uses columnar storage and processing in Spark SQL, native BLAS libraries in MLlib, and so on. As we discussed earlier, the only area where RDDs clearly add a cost is network latency, due to the synchronization at parallel communication steps.

One final observation from a systems perspective is that Spark may incur extra costs over some of today’s specialized systems due to fault tolerance. For example, in Spark, the map tasks in each shuffle operation save their output to local files on the machine where they ran, so reduce tasks can re-fetch it later. In addition, Spark implements a barrier at shuffle stages, so the reduce tasks do not start until all the maps have finished. This avoids some of the complexity that would be needed for fault recovery if one “pushed” records directly from maps to reduces in a pipelined fashion. Although removing some of these features would speed up the system, Spark often performs competitively despite them. The main reason is an argument similar to our previous one: many applications are bound by an I/O operation (such as shuffling data across the network or reading it from disk) and beyond this operation, optimizations (such as pipelining) add only a modest benefit. We have kept fault tolerance “on” by default in Spark to make it easy to reason about applications.

Ongoing Work

Apache Spark remains a rapidly evolving project, with contributions from both industry and research. The code-base size has grown by a factor of six since June 2013, with most of the activ-

ity in new libraries. More than 200 third-party packages are also available.^c In the research community, multiple projects at Berkeley, MIT, and Stanford build on Spark, and many new libraries (such as GraphX and Spark Streaming) came from research groups. Here, we sketch four of the major efforts.

DataFrames and more declarative APIs. The core Spark API was based on functional programming over distributed collections that contain arbitrary types of Scala, Java, or Python objects. While this approach was highly expressive, it also made programs more difficult to automatically analyze and optimize. The Scala/Java/Python objects stored in RDDs could have complex structure, and the functions run over them could include arbitrary code. In many applications, developers could get suboptimal performance if they did not use the right operators; for example, the system on its own could not push filter functions ahead of maps.

To address this problem, we extended Spark in 2015 to add a more declarative API called DataFrames² based on the relational algebra. Data frames are a common API for tabular data in Python and R. A data frame is a set of records with a known schema, essentially equivalent to a database table, that supports operations like filtering and aggregation using a restricted “expression” API. Unlike working in the SQL language, however, data frame operations are invoked as function calls in a more general programming language (such as Python and R), allowing developers to easily structure their program using abstractions in the host language (such as functions and classes). Figure 11 and Figure 12 show examples of the API.

Spark’s DataFrames offer a similar API to single-node packages but automatically parallelize and optimize the computation using Spark SQL’s query planner. User code thus receives optimizations (such as predicate push-down, operator reordering, and join algorithm selection) that were not available under Spark’s functional API. To our knowledge, Spark DataFrames are the first library to perform such

relational optimizations under a data frame API.^d

While DataFrames are still new, they have quickly become a popular API. In our July 2015 survey, 60% of respondents reported using them. Because of the success of DataFrames, we have also developed a type-safe interface over them called Datasets^e that lets Java and Scala programmers view DataFrames as statically typed collections of Java objects, similar to the RDD API, and still receive relational optimizations. We expect these APIs to gradually become the standard abstraction for passing data between Spark libraries.

Performance optimizations. Much of the recent work in Spark has been on performance. In 2014, the Databricks team spent considerable effort to optimize Spark’s network and I/O primitives, allowing Spark to jointly set a new record for the Daytona GraySort challenge.^f Spark sorted 100TB of data 3× faster than the previous record holder based on Hadoop MapReduce using 10× fewer machines. This benchmark was not executed in memory but rather on (solid-state) disks. In 2015, one major effort was Project Tungsten,^g which removes Java Virtual Machine overhead from many of Spark’s code paths by using code generation and non-garbage-collected memory. One benefit of doing these optimizations in a general engine is that they simultaneously affect all of Spark’s libraries; machine learning, streaming, and SQL all became faster from each change.

R language support. The SparkR project²¹ was merged into Spark in 2015 to provide a programming interface in R. The R interface is based on DataFrames and uses almost identical syntax to R’s built-in data frames. Other Spark libraries (such as MLlib) are also easy to call from R, because they accept DataFrames as input.

Research libraries. Apache Spark continues to be used to build higher-

^d One reason optimization is possible is that Spark’s DataFrame API uses lazy evaluation where the content of a DataFrame is not computed until the user asks to write it out. The data frame APIs in R and Python are eager, preventing optimizations like operator reordering.

^e <https://databricks.com/blog/2016/01/04/introducing-spark-datasets.html>

^f <http://sortbenchmark.org/ApacheSpark2014.pdf>

^g <https://databricks.com/blog/2015/04/28/>

^c One package index is available at <https://spark-packages.org/>

level data processing libraries. Recent projects include Thunder for neuroscience,⁵ ADAM for genomics,¹⁵ and Kira for image processing in astronomy.²⁷ Other research libraries (such as GraphX) have been merged into the main codebase.

Conclusion

Scalable data processing will be essential for the next generation of computer applications but typically involves a complex sequence of processing steps with different computing systems. To simplify this task, the Spark project introduced a unified programming model and engine for big data applications. Our experience shows such a model can efficiently support today's workloads and brings substantial benefits to users. We hope Apache Spark highlights the importance of composability in programming libraries for big data and encourages development of more easily interoperable libraries.

All Apache Spark libraries described in this article are open source at <http://spark.apache.org/>. Databricks has also made videos of all Spark Summit conference talks available for free at <https://spark-summit.org/>.

Acknowledgments

Apache Spark is the work of hundreds of open source contributors who are credited in the release notes at <https://spark.apache.org>. Berkeley's research on Spark was supported in part by National Science Foundation CISE Expeditions Award CCF-1139158, Lawrence Berkeley National Laboratory Award 7076018, and DARPA XData Award FA8750-12-2-0331, and gifts from Amazon Web Services, Google, SAP, IBM, The Thomas and Stacey Siebel Foundation, Adobe, Apple, Arimo, Blue Goji, Bosch, C3Energy, Cisco, Cray, Cloudera, EMC2, Ericsson, Facebook, Guavus, Huawei, Informatica, Intel, Microsoft, NetApp, Pivotal, Samsung, Schlumberger, Splunk, Virdata, and VMware. 

References

1. Apache Storm project; <http://storm.apache.org>
2. Armbrust, M. et al. Spark SQL: Relational data processing in Spark. In *Proceedings of the ACM SIGMOD/PODS Conference* (Melbourne, Australia, May 31–June 4). ACM Press, New York, 2015.
3. Dave, A. Indexedrd project; <http://github.com/amplab/spark-indexedrd>
4. Dean, J. and Ghemawat, S. MapReduce: Simplified data processing on large clusters. In *Proceedings of the Sixth OSDI Symposium on Operating Systems Design and Implementation* (San Francisco, CA, Dec. 6–8). USENIX Association, Berkeley, CA, 2004.
5. Freeman, J., Vladimirov, N., Kawashima, T., Mu, Y., Sofroniew, N.J., Bennett, D.V., Rosen, J., Yang, C.-T., Looger, L.L., and Ahrens, M.B. Mapping brain activity at scale with cluster computing. *Nature Methods* 11, 9 (Sept. 2014), 941–950.
6. Gonzalez, J.E. et al. GraphX: Graph processing in a distributed dataflow framework. In *Proceedings of the 11th OSDI Symposium on Operating Systems Design and Implementation* (Broomfield, CO, Oct. 6–8). USENIX Association, Berkeley, CA, 2014.
7. Isard, M. et al. Dryad: Distributed data-parallel programs from sequential building blocks. In *Proceedings of the EuroSys Conference* (Lisbon, Portugal, Mar. 21–23). ACM Press, New York, 2007.
8. Karloff, H., Suri, S., and Vassilvitskii, S. A model of computation for MapReduce. In *Proceedings of the ACM-SIAM SODA Symposium on Discrete Algorithms* (Austin, TX, Jan. 17–19). ACM Press, New York, 2010.
9. Kornacker, M. et al. Impala: A modern, open-source SQL engine for Hadoop. In *Proceedings of the Seventh Biennial CIDR Conference on Innovative Data Systems Research* (Asilomar, CA, Jan. 4–7, 2015).
10. Low, Y. et al. Distributed GraphLab: A framework for machine learning and data mining in the cloud. In *Proceedings of the 38th International VLDB Conference on Very Large Databases* (Istanbul, Turkey, Aug. 27–31, 2012).
11. Malewicz, G. et al. Pregel: A system for large-scale graph processing. In *Proceedings of the ACM SIGMOD/PODS Conference* (Indianapolis, IN, June 6–11). ACM Press, New York, 2010.
12. McSherry, F., Isard, M., and Murray, D.G. Scalability! But at what COST? In *Proceedings of the 15th HotOS Workshop on Hot Topics in Operating Systems* (Kartause Ittingen, Switzerland, May 18–20). USENIX Association, Berkeley, CA, 2015.
13. Melnik, S. et al. Dremel: Interactive analysis of Web-scale datasets. *Proceedings of the VLDB Endowment* 3 (Sept. 2010), 330–339.
14. Meng, X., Bradley, J.K., Yavuz, B., Sparks, E.R., Venkataraman, S., Liu, D., Freeman, J., Tsai, D.B., Amde, M., Owen, S., Xin, D., Xin, R., Franklin, M.J., Zadeh, R., Zaharia, M., and Talwalkar, A. MLlib: Machine learning in Apache Spark. *Journal of Machine Learning Research* 17, 34 (2016), 1–7.
15. Nothaft, F.A., Massie, M., Danford, T., Zhang, Z., Laserson, U., Yeksigian, C., Kottalam, J., Ahuja, A., Hammerbacher, J., Linderman, M., Franklin, M.J., Joseph, A.D., and Patterson, D.A. Rethinking data-intensive science using scalable analytics systems. In *Proceedings of the SIGMOD/PODS Conference* (Melbourne, Australia, May 31–June 4). ACM Press, New York, 2015.
16. Shun, J. and Blelloch, G.E. Ligra: A lightweight graph processing framework for shared memory. In *Proceedings of the 18th ACM SIGPLAN PPoPP Symposium on Principles and Practice of Parallel Programming* (Shenzhen, China, Feb. 23–27). ACM Press, New York, 2013.
17. Sparks, E.R., Talwalkar, A., Smith, V., Kottalam, J., Pan, X., Gonzalez, J.E., Franklin, M.J., Jordan, M.I., and Kraska, T. MLI: An API for distributed machine learning. In *Proceedings of the IEEE ICDM International Conference on Data Mining* (Dallas, TX, Dec. 7–10). IEEE Press, 2013.
18. Stonebraker, M. and Cetintemel, U. 'One size fits all': An idea whose time has come and gone. In *Proceedings of the 21st International ICDE Conference on Data Engineering* (Tokyo, Japan, Apr. 5–8). IEEE Computer Society, Washington, D.C., 2005, 2–11.
19. Thomas, K., Grier, C., Ma, J., Paxson, V., and Song, D. Design and evaluation of a real-time URL spam filtering service. In *Proceedings of the IEEE Symposium on Security and Privacy* (Oakland, CA, May 22–25). IEEE Press, 2011.
20. Valiant, L.G. A bridging model for parallel computation. *Commun. ACM* 33, 8 (Aug. 1990), 103–111.
21. Venkataraman, S. et al. SparkR; <http://dl.acm.org/citation.cfm?id=2903740&CFID=687410325&CFTOKEN=83630888>
22. Xin, R. and Zaharia, M. Lessons from running large-scale Spark workloads; <http://tinyurl.com/large-scale-spark>
23. Xin, R.S., Rosen, J., Zaharia, M., Franklin, M.J., Shenker, S., and Stoica, I. Shark: SQL and rich analytics at scale. In *Proceedings of the ACM SIGMOD/PODS Conference* (New York, June 22–27). ACM Press, New York, 2013.
24. Zaharia, M. An Architecture for Fast and General Data Processing on Large Clusters. Ph.D. thesis, Electrical Engineering and Computer Sciences Department, University of California, Berkeley, 2014; <https://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-12.pdf>
25. Zaharia, M. et al. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the Ninth USENIX NSDI Symposium on Networked Systems Design and Implementation* (San Jose, CA, Apr. 25–27, 2012).
26. Zaharia, M. et al. Discretized streams: Fault-tolerant streaming computation at scale. In *Proceedings of the 24th ACM SOSP Symposium on Operating Systems Principles* (Farmington, PA, Nov. 3–6). ACM Press, New York, 2013.
27. Zhang, Z., Barbary, K., Nothaft, N.A., Sparks, E., Zahn, O., Franklin, M.J., Patterson, D.A., and Perlmutter, S. Scientific Computing Meets Big Data Technology: An Astronomy Use Case. In *Proceedings of IEEE International Conference on Big Data* (Santa Clara, CA, Oct. 29–Nov. 1). IEEE, 2015.

Matei Zaharia (matei@cs.stanford.edu) is an assistant professor of computer science at Stanford University, Stanford, CA, and CTO of Databricks, San Francisco, CA.

Reynold S. Xin (rxin@databricks.com) is the chief architect on the Spark team at Databricks, San Francisco, CA.

Patrick Wendell (patrick@databricks.com) is the vice president of engineering at Databricks, San Francisco, CA.

Tathagata Das (tdas@databricks.com) is a software engineer at Databricks, San Francisco, CA.

Michael Armbrust (michael@databricks.com) is a software engineer at Databricks, San Francisco, CA.

Ankur Dave (ankurd@eecs.berkeley.edu) is a graduate student in the Real-Time, Intelligent and Secure Systems Lab at the University of California, Berkeley.

Xiangrui Meng (meng@databricks.com) is a software engineer at Databricks, San Francisco, CA.

Josh Rosen (josh@databricks.com) is a software engineer at Databricks, San Francisco, CA.

Shivaram Venkataraman (shivaram@cs.berkeley.edu) is a Ph.D. student in the AMPLab at the University of California, Berkeley.

Michael Franklin (mjfranklin@uchicago.edu) is the Liew Family Chair of Computer Science at the University of Chicago and Director of the AMPLab at the University of California, Berkeley.

Ali Ghodsi (ali@databricks.com) is the CEO of Databricks and adjunct faculty at the University of California, Berkeley.

Joseph E. Gonzalez (jegonzal@cs.berkeley.edu) is an assistant professor in EECS at the University of California, Berkeley.

Scott Shenker (shenker@icsi.berkeley.edu) is a professor in EECS at the University of California, Berkeley.

Ion Stoica (shenker@icsi.berkeley.edu) is a professor in EECS and co-director of the AMPLab at the University of California, Berkeley.

Copyright held by the authors.

Publication rights licensed to ACM. \$15.00



Watch the authors discuss their work in this exclusive *Communications* video. <http://cacm.acm.org/videos/spark>

DOI:10.1145/2934663

Enterprises that impose stringent password-composition policies appear to suffer the same fate as those that do not.

BY DINEI FLORÊNCIO, CORMAC HERLEY, AND PAUL C. VAN OORSCHOT

Pushing on String: The ‘Don’t Care’ Region of Password Strength

WE EXAMINE THE efficacy of tactics for defending password-protected networks from guessing attacks, taking the viewpoint of an enterprise administrator whose objective is to protect a population of passwords. Simple analysis allows insights on the limits of common approaches and reveals that some approaches spend effort in “don’t care” regions where added password strength makes no difference. This happens either when passwords do more than enough to resist online attacks while falling short of what is needed against offline attacks or when so many accounts have fallen that an attacker gains little from additional compromises.

Our review of tools available to improve attack-resistance finds that, for example, compelling returns are offered by password blacklists, throttling, and hash iteration, while current password-composition policies fail to provide demonstrable improvement in outcomes against offline guessing attacks.

Suppose a system administrator is tasked with defending a corporate, government, or university network or site. The user population’s passwords are targeted by attackers seeking access to network resources. Passwords can be attacked by guessing attacks—online or offline—and by capture attacks, that is, by non-guessing attacks. How to choose among password-attack mitigations is a practical question faced by millions of administrators. Little actionable guidance exists on how to do so in a principled fashion. Sensible policies must consider how to protect the population of user accounts. What is the best measure of the strength of a population of passwords? Is a good proxy for the overall ability of the network to resist guessing attacks the average, median, strongest, or weakest password? Slogans (such as suggesting that all passwords be “as strong as possible”) are too vague to guide action—and also suggest that infinite user effort is both available and achievable.

» key insights

- It has long been accepted that making users choose more complex passwords is the price they must pay to make their accounts safer; it turns out this line of thinking misunderstands how attacks actually work.
- Harder-to-guess passwords do not always reduce the likelihood of successful guessing attacks; in fact, in a large portion of the attack space, they make no difference at all.
- Enterprises should focus on users with the most easily guessed passwords; an attacker probably gets all the access needed just by compromising them, so improving other passwords denies the attacker very little.



The Compromise Saturation Point

To begin, consider an administrator who observes unusual behavior on his network and suspects that some accounts have been compromised. If he thinks it is just a few accounts, and can identify them, he might just block access to those. However, if he cannot be sure that an account is compromised until he sees suspicious activity, it is difficult to figure out the magnitude of the problem. Should he block access to all accounts and trigger a systemwide reset? If only a few accounts have been compromised, perhaps not; if half of them have been, he almost certainly should. What about a 1% compromise rate, or 5%, or 10%? At what point is global reset the right answer?

Suppose our hypothetical administrator resets all accounts. There still remains the question: How were the credentials obtained in the first place? If the door that led to the compromise remains open (such as undetected keyloggers on several machines) then nothing improves after a systemwide credential reset. On the other hand, if the credentials were compromised by guessing, then a reset (at least temporarily) helps, and a change in the policies that allowed vulnerable passwords might be in order. But even if he concludes that password guessing was the attack channel, was it online guessing? Or, somehow, did the attacker get hold of the password hash file and succeed with an offline guessing attack?

When attacks on passwords suc-

ceed, the specific attack channel is not necessarily clear; it is not obvious whether the compromised accounts had weak passwords, were spear-phished, or were drive-by download victims. Further, if it is not known which accounts have fallen, it may be best to reset all of them, even those not compromised. To facilitate more precise reasoning, let α be the fraction of credentials under attacker control—whether or not yet exploited. Thus, when $\alpha = 0.5$, half of the accounts (passwords) have already fallen to an attacker. At that point, would the administrator consider the network only 50% compromised, or fully overrun? The answer depends of course on the nature of the network in question. If a compromised account never has impli-

cations for any other account, then we might say the damage grows more-or-less linearly with α . However, for an enterprise network, a compromised account almost certainly has snowballing effects;³ a single credential might give access to many network resources, so the damage to the network grows faster than α (one possible curve is shown in Figure 1). At $\alpha = 0.5$, a system is arguably completely overrun. For many enterprise environments in this scenario, there would be few if any resources the attacker cannot access; in many social networks, the network value would approach zero, as spam would probably render things unusable; access to (probably well under) 50% of email inboxes likely yields a view to almost all company email, as an attacker requires access to only one of the sender(s) or recipient(s) of each message.

All password-based systems must tolerate some compromise; passwords will be keylogged, cross-site scripted, and spear-phished, and a network unable to handle this reality will not be able to function in a modern threat environment. As an attacker gains more and more credentials in an enterprise network, she naturally reaches some saturation point after which the impact of additional fallen credentials is negligible, having relatively little effect on the attacker's ability to inflict harm. The first credential gives initial access to the network, and the second, third, and fourth solidify the beachhead, but the benefit brought by each additional credential decreases steadily. The gain is substantial when k is small, less when large; the second password guessed helps a lot more than, say, password 102.

Let α_{sat} be the threshold value at which the attacker effectively has control of the password-protected part of the system, in the sense that there is negligible marginal gain from compromising additional credentials. That is, if an attacker had control of a fraction α_{sat} of account credentials, there are very few resources that she could not access; so the difference between controlling α_{sat} and $(\alpha_{\text{sat}} + \epsilon)$ is negligible. In what follows, our main focus is enterprise networks to consider possible values for α_{sat} .

There are a variety of tools attackers can use, once they have one set of



**The argument—
“stronger
passwords are
always better”—
while deeply
ingrained, thus
appears untrue.**



credentials, to get others. Phishing email messages that originate from an internal account are far more likely to deceive co-workers. Depending on the attacker's objective, a toehold in the network may be all she requires. The 2011 attack on RSA¹ (which forced a recall of all SecurID tokens) began with phishing email messages to “two small groups of employees,”¹³ none “particularly high-profile or high-value targets.” Dunagan et al.³ in examining a corporate network of more than 100,000 machines, found that 98.1% of the machines allowed an outward snowballing effect allowing compromise of 1,000 additional machines. Edward Snowden was able to compromise an enormous fraction of secrets on the NSA network starting from just one account.¹⁴ Given that RSA and the NSA (organizations we might expect to have above-average security standards) experienced catastrophic failures caused by handfuls of credentials in attackers' hands, we suggest a reasonable upper bound on the saturation point for a corporate or government network is $\alpha_{\text{sat}} \approx 0.1$; saturation likely occurs at much lower values.

It seems likely that enterprises will have the lowest values of α_{sat} ; at consumer Web services, compromise of one account has less potential to affect the whole network. As noted earlier, our focus here is on enterprise; nonetheless, we suggest that damage probably also grows faster than linearly at websites. For example, online accounts at a bank should have minimal crossover effects, but at 25% compromise all confidence in the legitimacy of transaction requests and the privacy of customer data is likely lost.

For guessing attacks, the most easily guessed passwords fall first. It is thus the weakest passwords that determine α_{sat} ; the number of guesses it takes to gather a cumulative fraction α_{sat} of accounts is what it takes to reach the saturation point. Since the attacker's ability to harm saturates once she reaches α_{sat} , the excess strength of the remaining $(1 - \alpha_{\text{sat}})$ of user passwords is wasted. For example, the strongest 50% of passwords might indeed be very strong, but from a systemwide viewpoint, that strength will come into play only when the other 50% of credentials has already been compromised—and

the attacker already has the run of everything that is password-protected in the enterprise network.

In summary, password strength, or guessing resistance, is not an abstract quantity to be pursued for its own sake. Rather, it is a tool we use to deny an attacker access to network resources. There is a saturation point, α_{sat} , where the network is so thoroughly penetrated that additional passwords gain the attacker very little; resistance to guessing beyond that point is wasted since it denies the attacker nothing. There is thus a “don’t care” region after that saturation level of compromise, and for enterprise networks, it appears quite reasonable to assume that α_{sat} is no higher than 0.1. At that level, it is not simply the case that the weakest 10% of credentials is most important, but that the excess strength of the remaining 90% is largely irrelevant to the administrator’s goal of systemwide defense. Of course there may be secondary benefits to individual users in having their password withstand guessing, even after α_{sat} has been exceeded. For example, if reused at another site, the user still has a significant stake in seeing the password withstand guessing. Since our focus is on the administrator’s goal of protecting a single site, this is not part of our model.

The Online-Offline Chasm

We next consider the difference between online and offline guessing. In online attacks, an attacker checks guesses against the defender’s server, that is, submitting guesses to the same server as legitimate users. In offline, she uses her own hardware resources, including networked or cloud resources and machines equipped with graphical processing units (GPUs) optimized for typical hash computations required to test candidate guesses. Offline attacks can thus test many-orders-of-magnitude more guesses than online attacks, whether or not online attacks are rate-limited by system defenses. We consider online and offline guessing attacks separately.

An online attack is always possible against a Web-facing service, assuming it is easy to obtain or generate valid account user IDs. An offline attack is possible only if the attacker gains access to the file of password hashes (in

order to verify correctness of guesses offline), and, in this case, an offline attack is necessary only if that file has been properly salted and hashed; otherwise, simpler attacks are possible⁶ (such as rainbow tables for unsalted hashed passwords).

There is an enormous difference between the strength required to resist online and offline guessing attacks. Naturally, the probability of falling either to an online or offline attack decreases

gradually with the number of guesses a password will withstand. One hundred guesses per account might be easy for an online attacker, but 1,000 is somewhat more difficult, and so on; at some point, online guessing is no longer feasible. Similarly, at some point the risk from an offline attack begins to gradually decrease. Let T_0 be a threshold representing the maximum number of guesses expected from an online attack and T_1 correspondingly the minimum

Figure 1. The fraction of network resources under attacker control grows faster than the fraction of credentials compromised.

For example, given half of a system’s credentials, an attacker likely effectively has access to all resources. For enterprise networks, we expect the saturation threshold, α_{sat} , where the network is completely penetrated, is likely under 0.1.

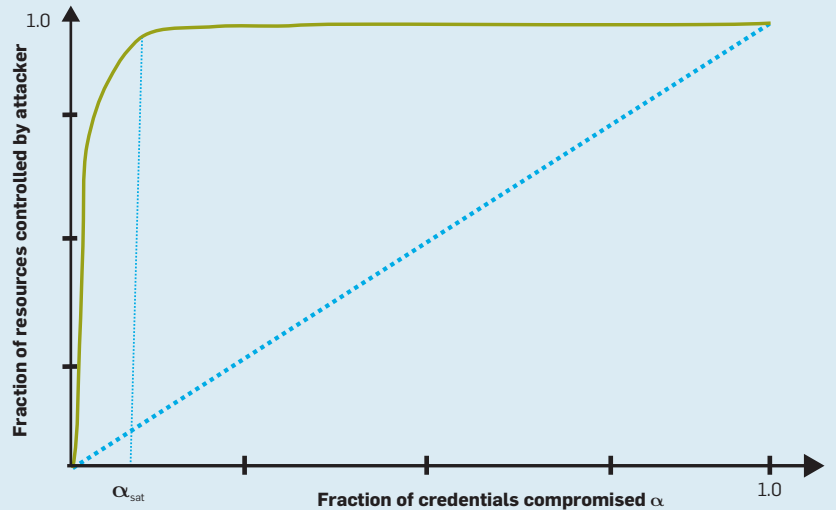
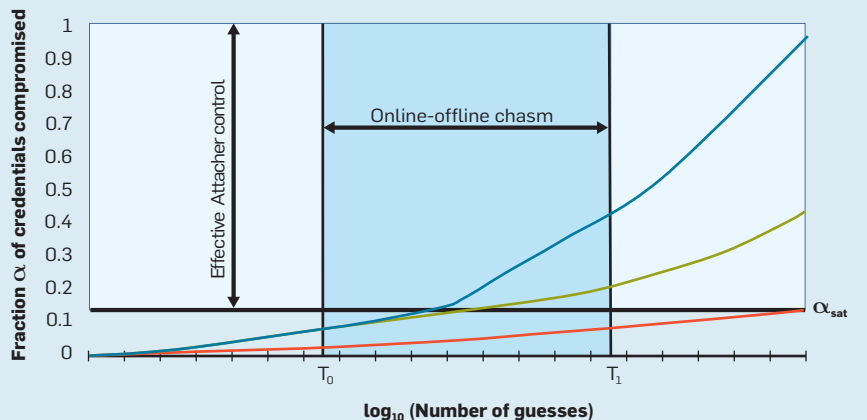


Figure 2. “Don’t care” regions where there is no return for increasing effort.

T_0 is the threshold above which online attacks cease to be a threat.
 T_1 is the threshold below which passwords almost surely will not survive credible offline attacks.
 α_{sat} is the threshold fraction of compromised accounts at which an attacker effectively has control of system resources. Examples for these parameters might be $T_0 = 10^6$, $T_1 = 10^{14}$, and $\alpha_{\text{sat}} = 0.1$.



number of guesses expected from a credible offline attack. (The asymmetry in these definitions is intentional to provide a conservative estimate in reasoning about the size of the gap.) A password withstanding T_0 guesses is then safe from online guessing attacks, while one that does not withstand T_1 guesses will certainly not survive offline attacks. Our own previous work⁶ suggests $T_0 \approx 10^6$ and $T_1 \approx 10^{14}$ are reasonable coarse estimates, giving a gap eight orders of magnitude wide (see Figure 2); we emphasize, however, that the arguments herein are generic, regardless of the exact values of T_0 and T_1 . While estimates for T_1 in particular are inexact, depending as they do on assumptions about attacker hardware and strategy (a previous estimate⁶ assumed four months of cracking against one million accounts using 1,000 GPUs, each doing one billion guesses per second), clearly T_1 vastly exceeds T_0 .

A critical observation is that for a password P whose guessing-resistance falls between T_0 and T_1 , incremental effort that fails to move it beyond T_1 is wasted in the sense that there is no guessing attack to which the original P falls, that the stronger password resists, since guessing attacks are either online or offline, with no continuum in between. Passwords in this online-

offline gap are thus in a “don’t care” region in which they do both too much and not enough—too much if the attack vector is online guessing and not enough if it is offline. Once a password is able to withstand T_0 guesses, to stop additional attacks, any added strength must be sufficient to move to the right of T_1 . While both online and offline attacks are countered by guessing-resistance, the amount needed varies enormously. In practical terms, distinct defenses are required to stop offline and online attacks. This online-offline chasm then gives us a second “don’t care” region, besides the one defined by α_{sat} .

The “Don’t Care” Region

The compromise saturation point and the online-offline chasm each imply regions where there is no return on effort. The marginal return on effort is zero for improving any password that starts in the zone greater than T_0 yet remains less than T_1 and for passwords with guessing-resistance above the α_{sat} threshold. Figure 2 depicts this situation, with shaded areas denoting the “don’t care” regions. A password withstanding $T_1 - \epsilon$ guesses has the same survival properties as one surviving $T_0 + \epsilon$; both survive online but fall to offline attack, that is, equivalent outcomes.

From the administrator’s point of view, the strongest password in the population, or having the greatest guessing-resistance (the password at $\alpha = 1.0$) is similar to one at $\alpha_{\text{sat}} + \epsilon$, in that both fall after the attacker’s ability to harm is already saturated.

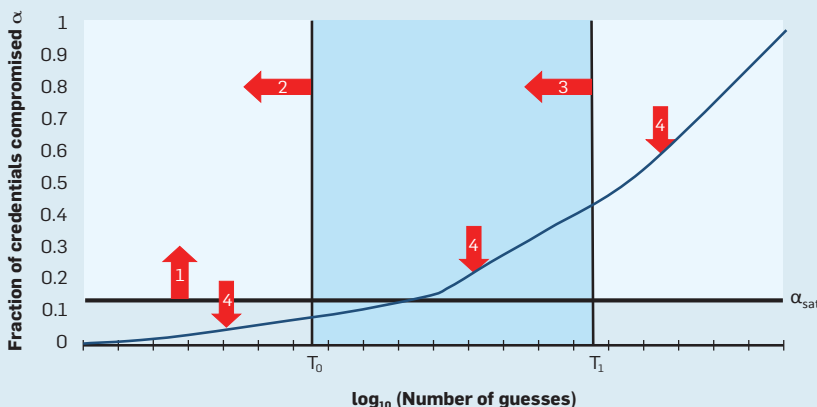
In summary, shaded regions denote these areas where there is no return-on-effort for “extra strength”: first, passwords whose guessing-resistance lies within the online-offline gap; and second, passwords beyond where an attacker gains little from additional credentials. The size of the “don’t care” region naturally depends on the particular values of α_{sat} , T_0 , and T_1 , but in all cases the shape of the password distribution (as defined by the colored curves in Figure 2) matters only in the areas below α_{sat} AND (left of T_0 OR right of T_1). An important observation is that, under reasonable assumptions, the “don’t care” regions cover a majority of the design space. The relatively small unshaded regions are shown in Figure 2; outside these regions, changes to the password distribution accomplish nothing, at least from the administrator’s viewpoint. To anchor the discussion, based on what we know of enterprise networks and attacker abilities, we have offered estimates of $\alpha_{\text{sat}} = 0.1$, $T_0 = 10^6$, and $T_1 = 10^{14}$; but choosing different values does not alter the conclusion that in large areas of the guess-resistance vs. credentials-compromised space, changing the distribution of user-chosen passwords improves little of use for the enterprise defender; it causes no direct damage but, like pushing on string, is ineffective and wasteful of user energy.

To understand the consequences of the “don’t-care” regions, Figure 2 also depicts guessing resistance of three hypothetical password distributions. The blue (top) and green (middle) curves diverge widely on the right side of the figure; for a fixed number of guesses, far fewer green-curve accounts will be compromised than accounts from the blue-curve distribution. The green curve might appear better since those passwords are much more guess-resistant than those from the blue curve. Nonetheless, they have identical attack survival outcomes since their divergence between T_0 and T_1 has minimal

Figure 3. Defensive elements aiming to improve guessing resistance.

- (1) α_{sat} (the point at which attacker control saturates) can be raised by implementing basic security principles (such as least-privilege and compartmentalization).
- (2) T_0 (the maximum number of online guesses) can be reduced by throttling mechanisms.
- (3) T_1 (the minimum number to be expected from an offline attack) can be reduced by iterated hash functions.
- (4) The cumulative fraction of accounts that have fallen at a given number of attacker guesses can be reduced (pushing the blue curve down) by improving the guessing-resistance of user-chosen passwords (such as to the left of T_0 by password blacklisting and by password composition policies generally).

Changing α_{sat} , T_0 and T_1 alter the size of white- and blue-shaded “don’t care” regions.



effect on performance against on-line or offline attacks, and divergence above α_{sat} happens only after the attacker's capacity for harm has already plateaued. The red (lower) curve shows a distribution that might survive an offline attack, as the curve lies below α_{sat} , even at the number of guesses an offline attacker might deliver. The enormous difficulty of getting users to choose passwords that will withstand offline guessing, and the waste that results unless almost all of them do, has also been argued by Tippet.¹¹ We reemphasize that α_{sat} , T_0 , and T_1 are site- and implementation-dependent variables. We next examine how an administrator can vary them to decrease the size of the “don't care” zone.

What Should an Administrator Optimize?

Ideally, a system's population of passwords would withstand both online and offline attacks. For this to be so, the fraction of accounts compromised must be lower than α_{sat} at T_1 guesses, and ideally much lower, since an offline attacker may be able to go well beyond the expected minimum T_1 . Unfortunately, recent work shows that user-chosen passwords do not even approach this level, even when stringent composition policies are enforced. For example, Mazurek et al.¹⁰ found that 48% of Carnegie Mellon University passwords (following a length-eight and four-out-of-four character sets policy) fell within 10^{14} guesses. On examining eight different password-creation policies, Kelley et al.⁸ found that none kept the cumulative fraction of accounts compromised below 10% by 10^{11} guesses. If we believe an attacker's control saturates by the time a fraction $\alpha_{\text{sat}} = 0.1$ of accounts is compromised, and an offline attack can mount at least $T_1 = 10^{11}$ guesses per account, there is thus little hope of resisting offline attack. That is, with these assumed values of α_{sat} and T_1 , an attacker who gains access to the hashed password file will have all the access she needs, no matter how far outside her reach the remaining fraction $(1 - \alpha_{\text{sat}})$ of passwords lie.

What then should an organization seek to do? It is ideal if passwords robustly withstand both online and offline attacks, as is the case for the lower



Password strength, which actually means guessing resistance, is not a universal good to be pursued for its inherent benefits; it is useful only to the extent it denies things to adversaries.



curve in Figure 2. However, striving for such resistance but falling short wastes user effort and accomplishes nothing. In Figure 2, all effort above and beyond that necessary to withstand online attack is completely wasted in the two upper curves. An organization that tries to withstand offline attacks but fails to have at least a fraction $1 - \alpha_{\text{sat}}$ of users survive T_1 guesses fares no better than one that never made the attempt and does worse if we assume user effort is a scarce resource.⁵ The evidence of recent work on cracking on real distributions^{8,10} suggests this is the fate of more or less all organizations that allow user-chosen passwords, unless we believe that $\alpha_{\text{sat}} \approx 0.1$ is unreasonably low or $T_1 \approx 10^{14}$ is unreasonably high. The argument—“stronger passwords are always better”—while deeply ingrained, thus appears untrue. Stronger passwords lower the cumulative fraction of accounts compromised at a given number of guesses, that is, push the curves in Figure 2 lower. However, changes that occur within the shaded “don't care” regions happen when it no longer matters and do not improve outcomes.

It follows that a reasonable objective is to maximize, within reasonable costs, guessing-resistance across a given system's set of passwords at the expected online and offline number of guesses, subject to the constraint that the fraction of compromised accounts stays below α_{sat} . That is, so long as $\alpha < \alpha_{\text{sat}}$, the lower α is at T_0 (respectively T_1) the better the resistance to online (respectively offline) attacks. For $\alpha > \alpha_{\text{sat}}$, improvements that do not reduce α below α_{sat} are unrewarded. Focusing as our model does on a single site, we remind readers that the additional benefit of withstanding guessing to users who have reused their password at other sites is not captured in this model.

What Can an Administrator Control?

What tools does an administrator have to reach these goals? The outcome of any administrator actions will be influenced by the values α_{sat} , T_0 , T_1 , and the shape of the cumulative password distribution. We show these various forces in Figure 3.

The value of the compromise saturation point α_{sat} is largely determined

by the network topology and might be relatively difficult to control or change in a given environment. Basic network hygiene and adherence to security principles (such as least privilege, isolation, and containment) can help minimize damage when intrusion occurs. These defenses are also effective against intrusions that do not involve password guessing. We assume these defenses will already be in force and in the rest of this section concentrate on measures that mostly affect password-guessing attacks.

Improving T_0 . There are few, if any, good reasons for an authentication system to allow hundreds of thousands of distinct guesses on a single account. In cases of actual password forgetting, it is unlikely that the legitimate user types more than one dozen or so distinct guesses. Mechanisms that limit the number of online guesses (thus reducing T_0) include various throttling mechanisms (rate limiting) and IP address blacklisting. The possibility of denial-of-service attacks can usually be dealt with by IP address whitelisting. (We mean, not applying the throttling triggered by new IP addresses to known addresses from which a previous login succeeded; wrong guesses from that known address should still be subject to throttling.) A simple, easily implemented throttling mechanism may suffice for many sites. When denial-of-service attacks are a possibility, more complex mechanisms may be necessary, perhaps including IP address white- and blacklisting, and methods requiring greater effort from site administrators. Such defensive improvements should come without additional burdens of effort and inconvenience to users. Together with password blacklisting (discussed in the following section), throttling may almost completely shut down generic online guessing attacks.

Improving T_1 . A password must withstand a relatively large number of guesses to have any hope of withstanding credible offline attacks. A lower bound T_1 on this number may vary depending on the defenses put in place and can be quite high. For example, if an attacker can make 10 billion guesses per second on each of 1,000 GPUs,⁷ then in a four-month period, she can try approximately $T_1 = 10^{14}$ guesses



If the hashed password file leaks, burdening users with complex composition policies does not alter the fact that a competent attacker likely gets access to all the accounts she could possibly need.



against each of one million accounts.⁶ Furthermore, technology advances typically aid attackers more than defenders; few administrators replace hardware every year, while attackers can be assumed to have access to the latest resources. This moves T_1 to the right—as computing speeds and technology advance and customized hardware gives attackers further advantages. Since the hardware is controlled by the attacker, little can be done to directly throttle offline guessing. However, an effective way to reduce the number of trials per second is to use a slow hash, that is, one that consumes more computation. The most obvious means is hash iteration;¹² recent research is also exploring the design of hash functions specifically designed to be GPU unfriendly. For example, ignoring the counteracting force of speed gains due to advancing technology, an iteration count of $n = 10^4$ reduces T_1 from 10^{14} to 10^{10} . Even with iteration it is difficult to move T_1 all the way left to T_0 to make the online-offline chasm disappear. Limits on further increasing n arise from the requirement that the time to verify legitimate users must be tolerable to both the users (wait time) and system hardware; for example, n might allow verifying 100 legitimate users/second (10ms per user). If 10ms is a tolerable delay, an attacker with access to 1,000 GPUs can compute a total of $1,000 \times 4 \times 30 \times 24 \times 60 \times 60 / 10^{-2} \approx 10^{12}$ guesses in four months. Directing this effort at 100 accounts would mean each would have to withstand a minimum of $T_1 = 10^{10}$ guesses. Since these are conservative assumptions, it appears challenging for a typical enterprise to decrease T_1 below this point.

Note that, as technology evolves, the number of hash iterations can easily be increased, invisibly to users and on the fly—by updating the stored password hashes in the systemside file to reflect new iteration counts.¹² Among the appealing aspects of iterated hashing, it is long known as an effective defensive tool, and costs are borne systemside rather than by user effort. However, hash iteration is not a miracle cure-all; for a password whose guess-resistance is 10^6 , online throttling is still important. An online attacker who could test one password every 10ms (matching the system rate noted earlier) will succeed in 10^4 seconds = 2 hours, 47minutes.

Eliminating offline attacks altogether. The emphasis by any parties who encourage users to choose stronger passwords can obscure the fact that offline attacks are only a risk when the password hash file “leaks,” a euphemism for “is somehow stolen,” or otherwise becomes available to an attacker. Any means or mechanisms that prevent the password hash file from leaking entirely remove the need for individual passwords to withstand an offline attack. Since we defined T_1 as the minimum number of guesses a password must withstand to resist an offline attack, any such mechanism effectively reduces T_1 to zero. We next discuss one particularly appealing such mechanism.

Hardware security modules (HSMs). A properly used HSM eliminates the risk of a hash file leaking⁶ or, equivalently, eliminates the risk of a decryption key (or backup thereof) leaking in the case that the means used to protect the information stored systemside to verify passwords is reversible encryption. In such a proper HSM architecture, rather than a file of the one-way hashes of salted passwords, what is stored in each file entry is a message authentication code (MAC) computed over the corresponding password using a secret key. When a password candidate is presented for verification, the candidate plus the corresponding MAC from the system file are provided as HSM inputs. The HSM holds the system secret key used to compute the MAC; importantly, this secret key is by design never available outside the HSM. Upon receiving the (MAC, candidate password) input pair, the HSM independently computes a MAC over the input candidate, compares it to the input MAC, and answers yes (if they agree) or no if they do not. Stealing the password hash file—in this case a password MAC file—is now useless to the offline attacker, because the HSM is needed to verify guesses; that is, offline attacks are no longer possible.

Another interesting scheme to mitigate offline attacks, proposed by Crescenzo et al.,² bounds the number of guesses that can be made by restricting the bandwidth of the connection between the authentication server and a specially constructed hash server that requires a very large collection of random bits. This approach limits an

attacker to online guessing, since, by design, the connection is too small to support the typical guessing rate an offline attacker needs, or to allow export of any file that would be useful to an offline attacker.

Improving the password distribution. Finally, we consider changes to the password distribution as a means of improving outcomes. Recall the curves in Figure 2 represent the cumulative fraction of accounts compromised as a function of the number of guesses per account. In general, it has been accepted without much second thought that the lower this cumulative fraction the better; a great deal of effort has gone into coercing users to choose supposedly “stronger” passwords, thus pushing the cumulative distribution curve downward in one or more of the three regions induced by T_0 and T_1 . However, as explained earlier, lower is of tangible benefit only outside the “don’t care” region; improvements to the curve inside the “don’t care” region have negligible effects on outcomes in any attack scenario.

First, note that tools to influence the cumulative distribution are mostly indirect; users choose passwords, not administrators. For example, by some combination of education campaigns, password policies, and password meters, administrators may try to influence this curve toward “better” passwords. However, the cumulative distribution is ultimately determined by user password choice; if users ignore advice, do the minimum to comply with policies, or are not motivated by meters, then efforts to lower the curve may have little impact on user choices.

Second, note that many policy and education mechanisms are unfocused in the sense that they cannot be targeted at the specific part of the cumulative distribution where they make the most difference—and away from the “don’t care” region where they make none. Even if they succeed, exhortations to “choose better passwords” are not concentrated at one part of the curve or another; if all users respond to such a request by improving their passwords marginally, the related effort of 90% of users is still wasted for an enterprise where $\alpha_{\text{sat}} = 0.1$. We now examine common approaches to influencing password choices in this light.

Password blacklisting. Attackers exploit the fact that certain passwords are common. Explicitly forbidding passwords known to be common can thus reduce risk. Blacklists concentrate at the head of the distribution, blocking the choice of most common passwords. For example, a user who attempts to choose “abcdefg” is informed this is not allowed and asked to choose another. Certain large services do this; for example, Twitter blacklisted 380 common passwords after an online guessing incident in 2009, and Microsoft applied a blacklist of several hundred to its online consumer properties in 2011. With improvements of password crackers and the recent wide availability of passwords lists, blacklists need to be longer.

A blacklist of, say, 10^6 common passwords may help bring guessing resistance to the 10^5 level. A natural concern with blacklists is that users may not understand why particular choices are forbidden. Kelley et al.⁸ examined the guessing resistance of blacklists of various sizes, but the question of how long one can be before the decisions appear capricious is an open one. Komanduri et al.⁹ pursue this question with a meter that displays, as a user types, the most likely password completion. A further unknown is the improvement achieved when users are told their password choice is forbidden. It appears statistically unlikely that all of the users who initially selected one of Twitter’s 380 blacklisted passwords would collide again on so small a list when making their second choice, but we are not aware of any measurements of the dispersion achieved. A promising recent practical password strength estimator is *zxcvbn*,¹⁵ a software tool that can also be of use for password blacklisting.

Observe that blacklists are both direct and focused: they explicitly prevent choices known to be bad rather than rely on indirect measures, and they target those users making bad choices, leaving the rest of the population unaffected. A blacklist appears to be one of the simplest measures to meaningfully improve the distribution in resisting online attacks.

Composition policies. Composition policies attempt to influence user-chosen passwords by mandating a

minimum length and the inclusion of certain character types; a typical example is “length at least eight characters, and use three of the four character sets {lowercase, uppercase, digits, special characters}.” Certain policies may help improve guess-resistance in the 10^5 to 10^8 range. However, for what we have suggested as the reasonable values $\alpha_{\text{sat}} = 0.1$ and $T_1 = 10^{14}$, the evidence strongly suggests that none of the password-composition policies in common use today or seriously proposed^{8,10} can help; for these to become relevant, one must assume that an attacker’s ability to harm saturates much higher than $\alpha_{\text{sat}} = 0.1$ or that the attacker can manage far fewer than $T_1 = 10^{14}$ offline guesses. Such policies thus fail to prevent total penetration of the network. Their ineffectiveness is perhaps the reason why a majority of large Web services avoid onerous policies.⁴

Note that composition policies are indirect: the constraints they impose are not themselves the true end objectives, but it is hoped they result in a more defensive password distribution. This problem is compounded by the fact that whether the desired improvement is indeed achieved is concealed from an administrator. The justifiably recommended practice of storing passwords as salted hashes means the password distribution is obscured, as are any improvements caused by policies. Composition policies are also unfocused in that they affect all users rather than being directed specifically where they may matter most. A policy may greatly affect user password choice and still have little effect on outcome (such as if all of the change in the cumulative distribution happens inside the “don’t care” region).

Conclusion

Password strength, which actually means guessing resistance, is not a universal good to be pursued for its inherent benefits; it is useful only to the extent it denies things to adversaries. When we consider a population of accounts, there are large areas where increased guessing resistance accomplishes nothing—either because passwords fall between the online and offline thresholds or because so many accounts have already fallen that attacker control has already saturated. If increases in password guess-

ing resistance were free this would not matter, but such increases are typically achieved at great cost in user effort; for example, there is a void of evidence that current approaches based on password composition policies significantly improve defensive outcomes and strong arguments that they waste much user effort. This situation thus creates risk of a false sense of security.

It is common to assume that users must choose passwords that will withstand credible offline attacks. However, if we assume an offline attacker can mount $T_1 = 10^{14}$ guesses per account and has all the access she needs by the time she compromises a fraction $\alpha_{\text{sat}} = 0.1$ of accounts, we must acknowledge that trying to stop offline attacks by aiming user effort toward choosing “better passwords” is unachievable in practice. The composition policies in current use seem so far from reaching this target that their use appears misguided. This is not to say that offline attacks are not a serious threat. However, it appears that enterprises that impose stringent password-composition policies on their users suffer the same fate as those that do not. If the hashed password file leaks, burdening users with complex composition policies does not alter the fact that a competent attacker likely gets access to all the accounts she could possibly need. Nudging users in the “don’t care” region (where most passwords appear to lie) is simply a waste of user effort.

The best investments to defend against offline attacks appear to involve measures transparent to users. Iteration of password hashes lowers the T_1 boundary; however, even with very aggressive iteration, we expect that at least 10^{10} offline guesses remain quite feasible for attackers. Use of MACs, that is, keyed hash functions, instead of regular (unkeyed) password hashes, provides effective defense against offline attacks that exploit leaked hash files—provided the symmetric MAC key is not also leaked. Use of HSMs is one method of protecting MAC keys, as discussed; though more expensive than software-only defenses, HSMs can eliminate offline attacks entirely. Online guessing attacks, in contrast, cannot be entirely eliminated, but effective defenses include password blacklists and throttling. There appear

few barriers to implementing these simple defenses. □

References

- Bright, P. RSA finally comes clean: SecurID is compromised. *Ars Technica* (June 6, 2011); <http://arstechnica.com/security/news/2011/06/rsa-finally-comes-clean-securid-is-compromised.ars/>
- Crescenzo, G.D., Lipton, R.J., and Walfish, S. Perfectly secure password protocols in the bounded retrieval model. In *Proceedings of the Theory of Cryptography Conference* (New York, Mar. 4–7). Springer-Verlag, 2006, 225–244.
- Dunagan, J., Zheng, A.X., and Simon, D.R. Heat-Ray: Combating identity snowball attacks using machine learning, combinatorial optimization and attack graphs. In *Proceedings of the ACM Symposium on Operating Systems Principles* (Big Sky, MT, Oct. 11–14). ACM Press, New York, 305–320.
- Florêncio, D. and Herley, C. Where do security policies come from? In *Proceedings of the SOUPS Symposium On Usable Privacy and Security* (Redmond, WA, July 14–16, 2010).
- Florêncio, D., Herley, C., and van Oorschot, P. Password portfolios and the finite-effort user: Sustainably managing large numbers of accounts. In *Proceedings of the 23rd USENIX Security Symposium* (San Diego, CA, Aug. 20–22). USENIX Association, Berkeley, CA, 2014, 575–590.
- Florêncio, D., Herley, C., and van Oorschot, P.C. An administrator’s guide to Internet password research. In *Proceedings of the USENIX LISA Conference* (Seattle, WA, Nov. 9–14). USENIX Association, Berkeley, CA, 2014, 35–52.
- Goodin, D. Why passwords have never been weaker and crackers have never been stronger. *Ars Technica* (Aug. 20, 2012); <http://arstechnica.com/security/2012/08/passwords-under-assault/>
- Kelley, P.G., Komanduri, S., Mazurek, M.L., Shay, R., Vidas, T., Bauer, L., Christin, N., Cranor, L.F., and Lopez, J. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *Proceedings of the IEEE Symposium on Security and Privacy* (San Francisco, May 20–23). IEEE Press, 2012, 523–537.
- Komanduri, S., Shay, R., Cranor, L.F., Herley, C., and Schechter, S. Telepathwords: Preventing weak passwords by reading users’ minds. In *Proceedings of the 23rd USENIX Security Symposium* (San Diego, CA, Aug. 20–22). USENIX Association, Berkeley, CA, 2014, 591–606.
- Mazurek, M.L., Komanduri, S., Vidas, T., Bauer, L., Christin, N., Cranor, L.F., Kelley, P., Shay, R., and Ur, B. Measuring password guessability for an entire university. In *Proceedings of the 20th ACM Conference on Computer and Communications Security* (Berlin, Germany, Nov. 4–8). ACM Press, New York, 2013.
- Tippett, P. Stronger passwords aren’t. *Information Security Magazine* (June 2001), 42–43.
- Provos, N. and Mazières, D. A future-adaptable password scheme. In *Proceedings of the 1999 USENIX Annual Technical Conference, FREENIX Track* (Monterey, CA, June 6–11). USENIX Association, Berkeley, CA, 1999, 81–91.
- RSA FraudAction Research Labs. Anatomy of a hack. RSA, Bedford, MA, Apr. 1, 2011; <https://blogs.rsa.com/anatomy-of-an-attack/>
- Toxen, B. The NSA and Snowden: Securing the all-seeing eye. *Commun. ACM* 57, 5 (May 2014), 44–51.
- Wheeler, D. zxcvbn: Low-budget password strength estimation. In *Proceedings of the 25th USENIX Security Symposium* (Austin, TX, Aug. 10–12). USENIX Association, Berkeley, CA, 2016.

Dinei Florêncio (dinei@microsoft.com) is a senior researcher in the Multimedia and Interactive Experiences group of Microsoft Research, Redmond, WA.

Cormac Herley (cormac@microsoft.com) is a principal researcher at Microsoft Research, Redmond, WA.

Paul C. van Oorschot (paulv@scs.carleton.ca) is a professor of computer science and Canada Research Chair in Authentication and Computer Security at Carleton University, Ottawa, Canada.

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

Actors linked to central others in networks are generally central, even as actors linked to powerful others are powerless.

BY ENRICO BOZZO AND MASSIMO FRANCESCHET

A Theory on Power in Networks

A “NETWORK” CONSISTS of a crowd of actors and a set of binary relations that tie pairs of actors. Networks are pervasive in the real world. Nature, society, information, and technology are supported by ostensibly different networks that in fact share an amazing number of interesting structural properties.

Networks are modeled in mathematics as “graphs,” with actors represented as points (also called nodes or vertices) and relations depicted as lines (also called edges or arcs) connecting pairs of points. In this article, we focus on undirected graphs, where the edges do not have a particular orientation. A meaningful question on networks is: Which are the most cen-

tral (important) nodes? Many measures have been proposed to address it. Among them, “eigenvector centrality” (or simply centrality in this article) states that an actor is central if it is connected with central actors. This circular definition is captured by an elegant recursive equation

$$\lambda x = Ax, \quad (1)$$

where x is a vector containing the sought centralities, A is a matrix encoding the network, and λ is a positive constant. Two actors in a network that are tied by an edge are said to be neighbors. Equation (1) claims two important properties of centrality: the centrality of an actor is directly correlated with the number of its neighbors and the centrality of its neighbors. Central actors are those with many ties or, for an equal number of ties, central actors are those connected with central others. This intriguing definition has been dis-

» key insights

- **Networks are everywhere; they are indeed the fabric of nature, society, information, technology, and sometimes even of art. All we need is an eye for them.**
- **The notion of centrality claims central actors are connected with central others, a mantra repeated over the past 70 years in econometrics, sociometry, bibliometrics, and information retrieval.**
- **Power, on the other hand, claims that powerful actors are connected with powerless others; it is meaningful in, say, bargaining situations, where it is favorable to negotiate with those with few options.**

covered and rediscovered many times over in different contexts. It has been investigated, in chronological order, in econometrics, sociometry, bibliometrics, Web information retrieval, and network science; see Franceschet¹² for an historical overview.

In some circumstances, however, centrality—the quality of being connected to central ones—has limited utility in predicting the locus of “power” in networks.^{2,8,11} Consider exchange networks, where the relationship in the network involves the transfer of valued items like information, time, money, or energy. A set of exchange relations is positive if exchange in one relation promotes exchange in others and negative if exchange in one relation inhibits exchange in others.⁷ In “negative exchange networks,” power comes from being connected to those with few options. Being connected to those with many possibilities reduces one’s power. Think of, for instance, a social network in which time is the exchanged value. Imagine every actor has a limited time to listen to others and that each actor divides its time among its neighbors. Exchange of time in one relation clearly precludes exchange of the same time in other relations. Which actors receive the most attention? They are the nodes that are connected to many neighbors with few options, since they receive almost full attention from all their neighbors. On the other hand, actors connected to few neighbors with many options receive little consideration because their neighbors are mostly busy with others.

In this article, we propose a theory on power in the context of networks. We start by this thesis: An actor is powerful if it is connected to powerless actors. We implement this circular thesis through this equation

$$x = Ax^{\dagger}, \quad (2)$$

where x is the sought power vector, A is a matrix encoding the network, and $x^{\dagger}$ is the vector whose entries are the reciprocal of those of x . Equation (2) states two important properties of power: the power of an actor is directly correlated with the number of its neighbors and is inversely correlated with the power of its neighbors. The first property seems reasonable;

Central actors are those with many ties or, for an equal number of ties, central actors are those connected with central others.

the more ties an actor has, the more powerful the actor is. The second property characterizes power; for an equal number of ties, actors linked to powerless others are powerful. On the other hand, actors tied to powerful others are powerless.

We investigate the existence and uniqueness of a solution for Equation (2), exploiting well-known results in combinatorial matrix theory. We study how to regain the solution when it does not exist, by perturbing the matrix representing the network. We formally relate the introduced notion of power with alternative notions and empirically compare them on the European natural gas pipeline network.

Motivating Example

In his seminal work on power-dependence relations, from 1962, Richard Emerson¹¹ claimed that power is a property of the social relation, not an attribute of the person: “X has power” is meaningless, unless we specify “over whom.” Power resides implicitly in others’ dependence, and dependence of an actor A upon actor B is directly proportional to A’s motivational investment in goals mediated by B and inversely proportional to the availability of those goals to A outside the A–B relation. The availability of such goals outside that relation refers to alternative avenues of goal achievement, most notably through other social relations.¹¹ This type of relational power is endogenous with respect to the network structures, meaning it is a function of the position of the node in the network. Exogenous factors (such as allure or charisma) external to the network structure might be added to endogenous power to complete the picture.

We begin with some small archetypal examples typically used in exchange-network theory to informally illustrate the notion of power and sometimes to distinguish it from the intersecting concept of centrality.¹⁰ Consider a two-node path

$$A-B.$$

The situation is perfectly symmetric, and a reasonable prediction is that both actors have the same power. In a three-node path

$A-B-C$,

much is changed. Intuitively, B is powerful and A and C are not. Indeed, both A and C have no alternative venues besides B (both depend on B), while B can exclude one of them by choosing the other.^a In a four-node path

$A-B-C-D'$

actors B and C hold power, while A and D are dependent on either B or C. Nevertheless, the power of B is less here than in the three-node path; in both cases, A depends on B, but in the three-node path, C also depends on B, while in the four-node path, C has an alternative, node D. Hence, node B is less powerful in the four-node path with respect to the three-node path since its neighbors are more powerful. Finally, the five-node path

$A-B-C-D-E$

is interesting since it discriminates power from centrality. All traditional central measures (eigenvector, closeness, betweenness) claim that C is the central one. Nevertheless, B and D are reasonably the powerful ones. Again, this is because they negotiate with weak partners (A and C or E and C), while C bargains with strong parties (B and D). This example is useful for illustrating an additional subtle aspect of power. Notice that in both the five-node path and the four-node path, B is surrounded by nodes (A and C) that are locally similar; for instance, they have the same degree in both paths. However, the power of C is reasonably less in the five-node path than in the four-node path; hence, we might expect the power of B is greater in the five-node path with respect to the four-node path. This separation is possible only if the notion of power spans beyond the local neighborhood of a node, if, say, power is recursively defined.

As a larger and more realistic example, consider Figure 1, which depicts the European natural gas pipeline

network. Nodes are European countries (country codes according to ISO 3166-1), and there is an undirected edge between two nations if a natural gas pipeline crosses the borders of the two countries. Data has been downloaded from the website of the International Energy Agency (<http://www.iea.org>). The original data corresponds to a directed, weighted multigraph, with edge weights corresponding to the maximum flow of the pipeline. We simplified and symmetrized the network, mapping the edge weights in a consistent way.

This is a negative exchange network because the exchange of gas with a country precludes the exchange of the same gas with others. Intuitively, powerful countries are those that are connected with states with few options for exchanging the gas. Suppose country B is connected to countries A and C, and B is the only connection for them, or $A-B-C$. Countries A and C can sell or buy gas only from B, while country B can choose between A and C. Reasonably, the bargaining power of B is greater, which translates in higher revenues or less expense for B in the gas negotiation.

A Theory on Power

Let G be an undirected, weighted graph. The graph G may contain “loops,” or edges from a node to itself. The edges of G are labeled with positive weights. Let A be the adjacency

matrix of G ; that is, $A_{i,j}$ is the weight of edge (i,j) if such edge exists and $A_{i,j} = 0$ otherwise. Hence, A is a square, symmetric, nonnegative matrix. Loops in G correspond to elements in the main diagonal of A .

The “centrality problem” is as follows: find a vector x with positive entries such that

$$\lambda x = Ax, \quad (3)$$

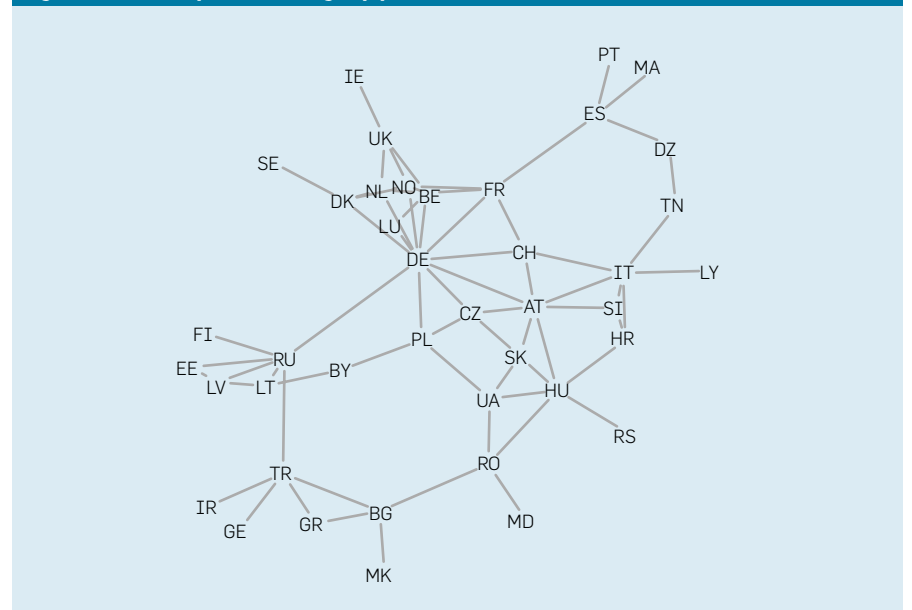
where $\lambda > 0$ is a constant. This means $\lambda x_i = \sum_j A_{i,j} x_j$; that is, the centrality of a node is proportional to the weighted sum of centralities of its neighbors. This is the main idea behind PageRank, Google’s original webpage ranking algorithm. PageRank determines the importance of a webpage in terms of the importance assigned to the pages that hyperlink to it. Besides Web information retrieval, this thesis has been successfully exploited in disparate contexts, including bibliometrics, sociometry, and econometrics.¹²

We define the “power problem” as follows: find a vector x with positive entries such that

$$x = Ax^\dagger, \quad (4)$$

where we denote with $x^\dagger$ the vector whose entries are the reciprocal of those of x . This means $x_i^\dagger = 1/x_i$; that is, the power of a node is equal to the weighted sum of reciprocals of power of its neighbors. Notice that if $\lambda x = Ax$, then, setting

Figure 1. The European natural gas pipeline network.



^a We assume here the so-called “1-exchange rule,” meaning each node may exchange with at most one neighbor. Likewise, we consider a negative exchange network in which the exchange in one relation inhibits exchange in others.

$y = \sqrt{\lambda}x$, we have that $y = Ay^\dagger$; hence, the proportionality constant λ is not necessary in the power equation. This notion of power is relevant on negative exchange networks.^{2,8} In these networks, when a value is exchanged between actors along a relation, it is consumed and cannot be exchanged along another relation. Hence, important actors are those in contact with many actors with few exchanging possibilities.

Finally, the “balancing problem” is the following: find a diagonal matrix D with positive main diagonal such that

$$S = DAD$$

is doubly stochastic; that is, all rows and columns of S sum to 1. The balancing problem is a fundamental question that is claimed to have first been used in the 1930s to calculate traffic flow⁴ and since then has been applied in many disparate contexts.¹⁴

It turns out that the power problem is intimately related to the balancing problem. Given a vector x , let D_x be the diagonal matrix whose diagonal entries coincide with those of x . We thus have the following result.

THEOREM 1. *The vector x is a solution of the power problem if and only if the diagonal matrix $D_{x^\dagger}$ is a solution for the balancing problem.*

PROOF. If DAD is doubly stochastic, then $DADe = e$ and $e^T DAD = e^T$, where e is a vector of all 1s. Actually, since A and D are symmetric, it holds that $DADe = e \Leftrightarrow e^T DAD = e^T$. If the vector x does not have zero entries, then D_x is invertible and $D_x^{-1} = D_{x^\dagger}$. We have that $x = Ax^\dagger \Leftrightarrow D_x e = AD_{x^\dagger} e \Leftrightarrow e = D_x^{-1} AD_{x^\dagger} e \Leftrightarrow e = D_{x^\dagger} AD_{x^\dagger} e$.

Existence and unicity of a solution. The link between the balancing problem and the power problem we established in Theorem 1 allows us to investigate a solution of the power problem (Equation 4) using the well-established theory of matrix balancing.

Recall that the “diagonal” of a square $n \times n$ matrix is a sequence of n elements that lies on different rows and columns of the matrix. A permutation matrix is a square $n \times n$ matrix that has exactly one entry equal to one in each row and each column, while all the other entries are equal to zero. Each diagonal clearly corresponds to a permutation matrix where the positions of the diagonal elements correspond to those of the unity entries of the permutation matrix. In particular, the identity matrix I is a permutation matrix, and the diagonal of A associated with I is called the main diagonal of A . A diagonal is positive if all its elements are greater than 0. A

matrix A is said to have “support” if it contains a positive diagonal and is said to have “total support” if $A \neq 0$ and every positive element of A lies on a positive diagonal. Total support clearly implies support.

A matrix is “indecomposable (irreducible)” if it is not possible to find a permutation matrix P such that

$$P^T AP = \begin{pmatrix} X & Y \\ 0 & Z \end{pmatrix},$$

where X and Z are both square matrices and 0 is a matrix of 0s; otherwise A is “decomposable (reducible).” A matrix is “fully indecomposable” if it is not possible to find permutation matrices P and Q such that

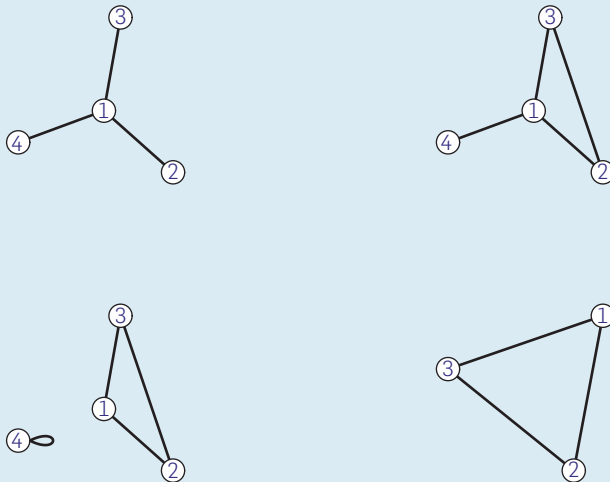
$$PAQ = \begin{pmatrix} X & Y \\ 0 & Z \end{pmatrix},$$

where X and Z are both square matrices; otherwise, A is “partly decomposable.” Clearly, a matrix (fully indecomposable) is also irreducible. It also holds that full indecomposability implies total support.⁵ Moreover, the adjacency matrix of a bipartite graph is never fully indecomposable, while the adjacency matrix of a non-bipartite graph is fully indecomposable if and only if it has total support and is irreducible.⁹ We say a graph has support, has total support, is irreducible, and is fully indecomposable if the corresponding adjacency matrix has these properties.

The combinatorial notions just outlined are rather terse. Fortunately, most of them have a simple interpretation in graph theory. It is known that irreducibility of the adjacency matrix corresponds to connectedness of the graph. Moreover, given an undirected graph G , let us define a “spanning cycle forest” of G a spanning subgraph of G whose connected components are single edges or cycles, including loops that are cycles of length 1. It is easy to realize that there exists a correspondence between diagonals in the adjacency matrix and spanning cycle forests in the graph. Hence, a graph has support if and only if it contains a spanning cycle forest and total support if and only if each edge is included in a spanning cycle forest. Four examples are given in Figure 2.

The following is a well-known necessary and sufficient condition for the solution of the balancing problem.^{9,17}

Figure 2. (top left) The graph has no support since a spanning cycle forest is missing. (top right). The graph has support formed by edges (1, 4) and (2, 3), but the support is not total; (edges (1, 3) and (1, 2) are not part of any spanning cycle graph. (bottom left) The graph has total support but is not irreducible, hence is not fully indecomposable. (bottom right) The graph has total support and is irreducible and not bipartite, so is fully indecomposable.



THEOREM 2. *Let A be a symmetric nonnegative square matrix. A necessary and sufficient condition for the existence of a doubly stochastic matrix S of the form DAD , where D is a diagonal matrix with positive main diagonal, is that A has total support. If S exists, then it is unique. If A is fully indecomposable, then matrix D is unique.*

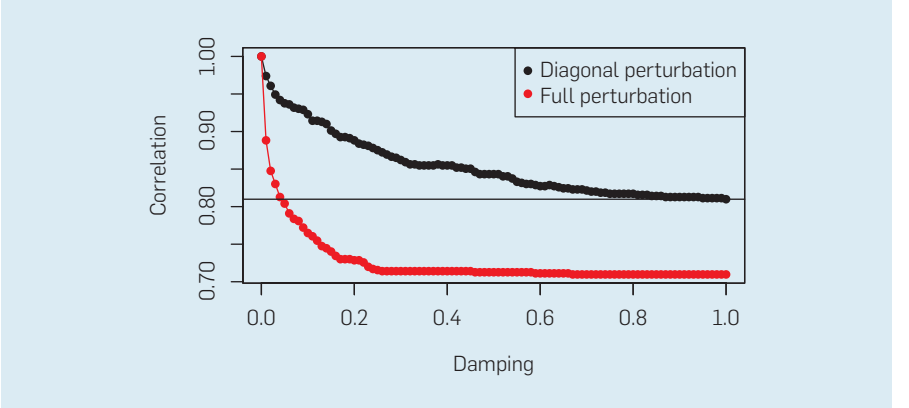
It follows that the power problem $x = Ax^\pm$ has a solution on the class of graphs that has total support. Moreover, if the graph is fully indecomposable, then the solution is also unique.

Perturbation (regaining the solution). What about the power problem on graphs whose adjacency matrix lacks total support? For such graphs, the power problem has no solution. Nevertheless, a solution can be regained by perturbing the adjacency matrix of the graph in a suitable way. We investigate two perturbations on the adjacency matrix A

1. *Diagonal perturbation:* $A_\alpha^D = A + \alpha I$, where $\alpha > 0$ is a damping parameter and I is the identity matrix.
2. *Full perturbation:* $A_\alpha^F = A + \alpha E$, where $\alpha > 0$ is a damping parameter and E is a full matrix of all 1s.

Matrix A_α^F is clearly fully indecomposable, has total support, and is irreducible. Hence, the power problem (as well as the centrality problem) on a fully perturbed matrix has a unique solution. On the other hand, matrix A_α^D has total support. Indeed, if $A_{i,j} > 0$ and $i = j$, then the main diagonal $A_{k,k}$ for $1 \leq k \leq n$ is positive and contains $A_{i,i}$. If $i \neq j$, then the diagonal $A_{i,i}, A_{j,j}, A_{k,k}$ for $1 \leq k \leq n$ and $k \neq i, j$ is positive and contains $A_{i,j}$. The power problem on a diagonally perturbed matrix thus has a solution. Moreover, the solution is unique if A is irreducible, since it is known that for a symmetric matrix A it holds that A is irreducible if and only if $A + I$ is fully indecomposable.⁶ Interestingly, the diagonal perturbation, besides providing convergence of the method, is useful for incorporating exogenous power in the model. By setting a positive value in the i th position of the diagonal, we are saying that node i has a minimal amount of power, or not a function of the position of the node

Figure 3. Correlation between original and perturbed powers varying the damping parameter from 0 to 1 on the largest biconnected component of the social network among dolphins (which has total support). The horizontal line corresponds to the correlation with diagonal perturbation and maximum damping. The correlation on the other networks is similar.



in the network. We can thus play with the diagonal of the adjacency matrix to assign nodes with potentially different entry levels of exogenous power.

Intuitively, the diagonal perturbation is less invasive than its full counterpart; the former modifies the diagonal elements only, and the latter touches all matrix elements. But how invasive is the perturbation with respect to the resulting power? To investigate this issue, we computed the correlation between original and perturbed power solutions. A simple and intuitive measure of the correlation between two rankings of size n is Kendall rank correlation coefficient k , which is the difference between the fraction of concordant pairs c (the number of concordant pairs divided by $n(n-1)/2$) and that of discordant pairs d in the two rankings: $k = c - d$. The coefficient runs from -1 to 1 , with negative values indicating negative correlation, positive values indicating positive correlation, and values close to 0 indicating independence. We used the following network datasets: a social network among dolphins, the Madrid train bombing terrorist network, a social network of jazz musicians, a network of friendships between members of a karate club, a collaboration network of scholars in the field of network science, and a co-appearance network of characters in the novel *Anna Karenina* by Lev Tolstoj.

The main outcomes of the current experiment are as follows (see Figure 3): as soon as the damping parameter is small, both diagonal and full perturbations do not significantly change the

original power; power with diagonal perturbation is closer to original power than power with full perturbation; and the larger the damping parameter, the lower the adherence of perturbed solutions to the original one.

Computing power. Due to the established relationship between the balancing problem and the power problem, we can use known methods for the former in order to solve the latter. The simplest approach for solving Equation (4) is to set up the iterative method

$$x_{k+1} = Ax_k^\pm, \quad (5)$$

known as the Sinkhorn–Knopp method (SKM).¹⁷ If we set $x_0 = e$, the vector of all 1s, then the first iteration $x_1 = Ae$; that is, $x_1(i) = \sum_j A_{i,j}$ is the degree d_i of i . The second iteration $x_2 = A(Ae)^\pm$; that is, $x_2(i) = \sum_j A_{i,j}/d_j$ is the sum of reciprocals of the degrees of the neighbors of i .

If A has total support, then the SKM converges, or, more precisely, the even and odd iterates of the method converge to power vectors that differ by a multiplicative constant. The convergence is linear with a rate of convergence that depends of the subdominant eigenvalue of the balanced matrix $S = DAD$ (see Theorem 2).¹⁷ In some cases, however, the convergence can be very slow. Knight and Ruiz¹⁴ proposed a faster algorithm based on Newton’s method (NM) that we now describe according to our setting and notations. In order to solve Equation (4), we apply NM for finding the zeros of the function $f: \mathbb{R}^n \rightarrow \mathbb{R}^n$ defined by $f(x) = x - Ax^\pm$. It is not difficult to check that

Table 1. Complexity of computation of power with different methods: PM (benchmark); SKM (SKM without perturbations for totally supported networks); SKM-D (SKM with diagonal perturbation and damping 0.15); SKM-F (SKM with full perturbation and damping 0.01); NM (NM without perturbations for totally supported networks); and NM-D (NM with diagonal perturbation and damping 0.15).

Network	PM	SKM	SKM-D	SKM-F	NM	NM-D
Dolphin	73	294	300	72	47	30
Madrid	28	416	320	78	46	27
Jazz	42	300	288	78	37	27
Karate	42	–	494	52	–	31
Collab	65	–	9740	30	–	33
Karenina	24	–	1006	32	–	32

Table 2. Correlation of power, as defined in this article, with degree (D), centrality (C), Bonacich power (B), Shapley power (S), and Nash power (N).

Network	D	C	B	S	N
Dolphin	0.81	0.35	0.89	0.91	0.72
Madrid	0.62	0.33	0.69	0.68	0.48
Jazz	0.85	0.62	0.91	0.85	0.17
Karate	0.77	0.36	0.74	0.96	0.75
Collaboration	0.77	0.05	0.77	0.85	0.60
Karenina	0.75	0.45	0.62	0.89	0.86

For the computation of power, we used diagonal perturbation (damping 0.15). For Bonacich power we used $\alpha = 1$ and $\beta = -0.85/r$, where r is the spectral radius of the graph.

$$\frac{\partial f_i}{\partial x_j}(x) = \delta_{i,j} + \frac{A_{i,j}}{x_j^2}, \quad i, j = 1, \dots, n,$$

where $\delta_{i,j} = 1$ if $i = j$ and $\delta_{i,j} = 0$ otherwise. We collect these partial derivatives in the Jacobian matrix of f that turns to be

$$J_f(x) = I + AD_{(x)}^2,$$

where the squaring of x is to be intended entrywise. Formally, the NM applied to the equation $f(x) = 0$ becomes

$$\begin{aligned} x_{k+1} &= x_k - J_f^{-1}(x_k) f(x_k) \\ &= J_f^{-1}(x_k) (J_f(x_k) x_k - f(x_k)) \\ &= J_f^{-1}(x_k) (x_k + AD_{(x_k)}^2 x_k - x_k - Ax_k^+) \\ &= 2J_f^{-1}(x_k) Ax_k^+. \end{aligned}$$

To apply NM precisely it is necessary to solve a linear system at each step, but this would be too expensive. Nevertheless, an approximate solution of the system obtained by means of an iterative method is sufficient, giving rise to an inner-outer iteration. This approach is appealing when the matrix that has to be balanced is symmetric and sparse, which is the case for the power problem on real networks.¹⁴

We experimentally assessed the complexity of computation of power on the real social networks; in fact, we used the largest biconnected component for the first three networks in order to also work with totally supported graphs. We use both SKM and NM. We consider the computation on the original matrix, as well as on the perturbed ones. We use as a benchmark the complexity of the computation of centrality using the power method (PM). The complexity is expressed as the overall number of matrix-vector product operations. If a matrix is sparse (the case for all tested networks), such operation has linear complexity in the number of nodes of the graph. The main empirical findings are summarized as follows (see Table 1): SKM on the original matrix is significantly slower than PM, and diagonal perturbation does not significantly change its speed; full perturbation significantly increases the speed of SKM, so the complexity of SKM with full perturbation and that of PM are comparable (moreover, the larger the damping parameter, the faster the method); NM on the original matrix is much faster than SKM: its complexity is comparable to that of fully perturbed SKM

and PM; and NM with diagonal perturbation is even faster than NM, and the larger the damping parameter, the faster the method.

Relationship with alternative power measures. Bonacich² proposed a family of parametric measures depending on two parameters: α and β . If A is the adjacency matrix of the graph, the Bonacich index x is defined as

$$x = \alpha Ae + \beta Ax. \quad (6)$$

The index for a node is the sum of two components: a first one (weighted by the parameter α) depends on the node's degree, and a second one (weighted by the parameter β) depends on the index on the node's neighbors. From Equation (6), under the condition that $I - \beta A$ is not singular, it is possible to obtain the following explicit representation of the proposed measure

$$x = \alpha (I - \beta A)^{-1} Ae = \alpha \left(\sum_{k=0}^{\infty} \beta^k A^{k+1} \right) e. \quad (7)$$

The equivalence with the infinite sum holds when $|\beta| < 1/r$, where $r = \max_i |\lambda_i|$, with λ_i the eigenvalues of A ; that is, r is the spectral radius of A . When the parameter β is positive, the index is a centrality measure. In particular, the measure approaches eigenvector centrality as a limit as β approaches $1/r$. On the other hand, when β is negative, the index is a power measure; it corresponds to a weighted sum of odd-length paths (with positive sign) and even-length paths (with negative sign).² Hence, powerful nodes correspond to nodes with many powerless neighbors. Finally, when $\beta = 0$, the measure boils down to degree centrality.

The difficulty with this measure is that it is parametric; that is, it depends on parameters α and β . While it is simple to set the parameter α , and it can be used to assign exogenous power to nodes, the choice for the parameter β is more delicate. In particular, the index makes sense when the parameter $|\beta| < 1/r$, hence the spectral radius r must be computed or at least approximated.

The precise relationship between Bonacich power (Bonacich index with negative β) and power defined in Equation (4) is explained as follows: If we set $x_0 = (1/\gamma)e$ in Newton's

iteration for the computation of power described earlier we obtain

$$x_1 = 2\gamma (I + \gamma^2 A)^{-1} A e.$$

But this first approximation is a member of the family of Bonacich's measures, with $\alpha = 2\gamma$ and $\beta = -\gamma^2$. Since β is negative, we indeed are facing a measure of power. Hence, Bonacich power can be considered as a first-order approximation of power using NM.

Bozzo et al.³ investigated power measures on sets of nodes. Given a node set T let $B(T)$ be the set of nodes whose neighbors all belong to T . Notice that nodes in $B(T)$ do not have connections outside T , hence are potentially at the mercy of nodes in T . We define a power function p such that $p(T) = |B(T)| - |T|$. Hence, a set T is powerful if it has potential control over a much larger set of neighbors $B(T)$. The power measure is interpreted as the characteristic function of a coalition game played on the graph and the "Shapley value" of the game; or the average marginal contribution to power carried by a node when it is added to any node set is proposed as a measure of power for single nodes. Interestingly, the discovered game-theoretic power measure corresponds to the second iteration of SKM for the computation of power as defined by Equation (4); that is, to the sum of reciprocals of neighbors' degrees.

The study of power has a long history in economics (in its recognition of bargaining power) and sociology (in its interpretation of social power).¹⁰ Consider the most basic case where just two actors, A and B , are involved in a negotiation over how to divide one unit of money. Each actor has an alternate option—a backup amount it can collect in case negotiations fail, say, α for A and β for B . A natural prediction, known as "Nash's bargaining solution,"¹⁵ is that the two actors will split the surplus $s = 1 - \alpha - \beta$, if any, equally between them; that is, if $s < 0$ no agreement between A and B is possible, since any division is worse than the backup option for at least one of the parties. On the other hand, if $s \geq 0$, then A and B will agree on $\alpha + s/2$ for A and $\beta + s/2$ for B .

A natural extension of the Nash bargaining solution from pairs of actors

The power of an actor somewhat inversely depends on the power of its neighbors.

to networks of actors was proposed in Cook et al.⁸ and Rochford¹⁶ and further investigated, particularly in Bayati et al.¹ and Kleinberg et al.¹³ In the following, we describe the dynamics that capture such an extension. Let A be the adjacency matrix of an undirected, unweighted graph G . Hence, $A_{ij} = 1$ if there is an edge (i, j) in G and $A_{ij} = 0$ otherwise. Negotiation among actors is possible only along edges; each pair of actors on an edge negotiates for a fixed amount of €1, and each actor may conclude a negotiation with at most one neighbor (one-exchange rule). For every edge (i, j) , define

- R_{ij} as the amount of "revenue" actor i receives in a negotiation with j .
- L_{ij} as the amount of revenue actor i receives in the best alternative negotiation, excluding the one with j .

Notice that matrices R and L have the same zero-non-zero pattern as A . More precisely, consider the following iterative process. We start with $R_{ij}^{(0)} = 1/2$ for all edges (i, j) and $R_{ij}^{(0)} = 0$ elsewhere. Let $N(i)$ be the set of neighbors of node i . For $t > 0$, the best alternative matrix $L^{(t)}$ at time t is

$$L_{ij}^{(t)} = \begin{cases} \max_{k \in N(i) \setminus j} R_{ik}^{(t-1)} & \text{if } A_{ij} = 1, \\ 0 & \text{otherwise.} \end{cases}$$

Let the surplus $S_{ij}^{(t)} = 1 - L_{ij}^{(t)} - L_{ji}^{(t)}$ be the amount for which actors i and j will negotiate at time t ; notice that actor i will never accept an offer from j less than his alternate option $L_{ij}^{(t)}$, and actor j will never accept an offer from i less than her alternate option $L_{ji}^{(t)}$. The profit matrix $R^{(t)}$ at time t is then

$$R_{ij}^{(t)} = \begin{cases} L_{ij}^{(t)} + s_{ij}^{(t)} / 2 & \text{if } A_{ij} = 1 \text{ and } s_{ij}^{(t)} \geq 0, \\ 1 - L_{ji}^{(t)} & \text{if } A_{ij} = 1 \text{ and } s_{ij}^{(t)} < 0, \\ 0 & \text{otherwise.} \end{cases}$$

Notice that $R_{ij}^{(t)} + R_{ji}^{(t)} = 1$; that is, $R_{ij}^{(t)}$ and $R_{ji}^{(t)}$ is the Nash's bargaining solution of a negotiation between actors i and j , given their alternate options $L_{ij}^{(t)}$ and $L_{ji}^{(t)}$. Let R be the fixpoint of the iterative process $R^{(t)}$ for growing time t . The "Nash power" x_i of node i is the best revenue of actor i among its neighbors; that is

Table 3. Matrix of correlations among power and centrality measures.

	S	B	P	N	C
S	1.00	0.82	0.90	0.69	0.41
B	0.82	1.00	0.84	0.61	0.46
P	0.90	0.84	1.00	0.72	0.47
N	0.69	0.61	0.72	1.00	0.36
C	0.41	0.46	0.47	0.36	1.00

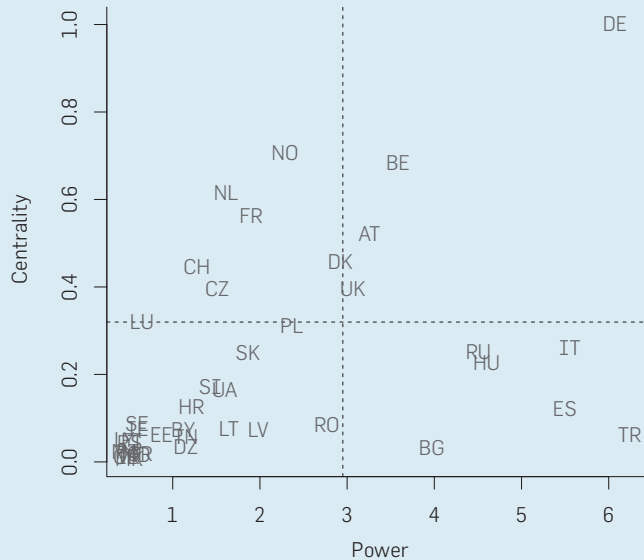
S, Shapley power; B, Bonacich power; P, power as defined in this article; N, Nash power; C, centrality.

Table 4. The top 10 powerful and central countries in the European natural gas exchange network.

P	TR	DE	IT	ES	HU	RU	BG	BE	AT	UK
	6.26	6.09	5.54	5.50	4.62	4.53	3.99	3.60	3.29	3.09
B	DE	IT	HU	TR	AT	RU	ES	BE	NO	BG
	7.07	4.58	4.06	3.73	3.41	3.37	3.29	3.23	2.92	2.76
S	TR	ES	IT	DE	RU	HU	BG	RO	UK	AT
	2.92	2.70	2.56	2.54	2.46	2.23	1.95	1.67	1.53	1.51
N	ES	TR	BG	RU	IT	HU	UK	RO	DK	LV
	1.00	1.00	1.00	0.87	0.83	0.83	0.75	0.75	0.75	0.75
C	DE	NO	BE	NL	FR	AT	DK	CH	CZ	UK
	1.00	0.71	0.68	0.62	0.56	0.52	0.46	0.45	0.40	0.39

P, power as defined in this article; B, Bonacich power; S, Shapley power; N, Nash power; C, centrality.

Figure 4. Scatterplot of power versus centrality. Vertical and horizontal lines correspond to third quartile.



$$x_i = \max_j R_{i,j}.$$

Among many other attractive results, Bayati et al.¹ showed that the dynamics always converge to a fixpoint solution.

Nash power bears some analogy with the one we propose and investigate here; in particular, both notions share the same recursive powerful-is-linked-with-powerless philosophy. Nash power

for an actor i depends directly on the revenues of i among its neighbors, which directly depend on the alternate options of i among its neighbors, which inversely depends on the revenues of neighbors of i , which determine the power of neighbors of i . Hence, power of an actor somewhat inversely depends on the power of its neighbors.

Using Kendall correlation, we assessed the overlapping of power, as defined in this article, with centrality and degree, as well as Bonacich power (Bonacich index with negative parameter β), Shapley power (the sum of reciprocals of neighbors' degrees), and Nash power on the social networks mentioned earlier. The main empirical outcomes are summarized in the following (see Table 2): as expected, both power and centrality are positively correlated with degree, but power is negatively correlated with centrality when the effect of degree is excluded (we used partial correlation); power is positively correlated with Bonacich power, and the association increases as the parameter β declines below 0 down to $-1/r$, with r the spectral radius of the adjacency graph matrix (moreover, the association is greater when the adjacency matrix is perturbed); power is positively correlated with Shapley power, and the association is generally stronger than with Bonacich power; and power is positively correlated with Nash bargaining network power, but the strength of the correlation is generally weaker than with Shapley power and Bonacich power. In particular, we noticed that the Nash-based method maps the power scores of the nodes of the surveyed networks into a small set of values, with very high frequency for values close to 0, 0.5, and 1. Hence, it is difficult to discriminate different gradations of power for nodes.

Motivating Example Reloaded

Here, we revisit examples from the “Motivating Example” section, using them as a benchmark to compare the different notions of power described in the “Relationship with Alternative Power Measures” section. When the graphs are not totally supported (all cases but the two-node path), we used diagonal perturbation with damping 0.15 to obtain a solution. Moreover,

we set Bonacich index parameters $\alpha = 1$ and $\beta = -0.85/r$, where r is the spectra radius of the graph.

In the two-node path, all methods agree to give identical power to both nodes. In the three-node path $A-B-C$, all methods agree B is the powerful one. Notably, Nash power assigns all power (1) to B and no power (0) to A and C, while the other methods say A and C hold a small amount of power. In the four-node path $A-B-C-D$, all methods claim B and C are the powerful ones. Moreover, all methods recognize that the power of B in this instance is less than its power in the three-node path. Finally, in the five-node path $A-B-C-D-E$, all methods discriminate B and D as the most powerful nodes, followed by C and finally A and E, with the only exception of Nash power, which assigns all power (1) to B and D and null power (0) to all other nodes; hence, the central node C has the same power as the peripheral nodes A and E, according to this method. All methods, with the exception of Shapley, notice that the power of B is greater in the five-node path with respect to the three-node path. This is because Shapley is a local method, while the others are global (recursive) methods.

Let us now revisit the natural gas pipeline example. We ranked all countries according to the following power and centrality measures: Shapley power (S), Bonacich power (B), power as defined in this article (P), Nash power (N), and eigenvector centrality (C). Table 3 shows the corresponding Kendall correlation matrix. As expected, P is well correlated with its approximations B and S. Moreover, P is positively correlated with N, but the correlation strength is weaker. Also, the association between P and C is positive but weak and mostly explained by the association with degree of both measures. Indeed, if we exclude the effect of degree, this correlation is negative.

These associations are mirrored in the top-10 rankings and ratings listed in Table 4, as well as in the scatter-plot comparing power and centrality in Figure 4. Notice how Germany (DE) is both powerful and central; Italy (IT) and Turkey (TR) are powerful but not central; Norway (NO) is central

but not powerful; and there are many countries that are neither powerful nor central outside the rankings. For instance, Italy contracts with nations that are both powerless and peripheral, namely Austria, Switzerland, Croatia, Tunisia, Libya, and Slovenia, with only Austria included in the top-10 power list and only Austria and Switzerland included in the top-10 centrality list (not in the first positions). The ranking according to Nash power is somewhat unusual if compared with the other power measures; for instance, Germany has bargaining power 0.5 and only in 14th position, tied with the other countries. It is fair to note that the generalized Nash bargaining solution was originally proposed in the context of assignment problems (such as in matching apartments to tenants and students to colleges) and was not suggested as a rating-and-ranking method for nodes in a network. For instance, in balanced matching over the gas network, Italy preferably negotiates with Libya and Turkey with Georgia. In fact, Cook and Yamagishi⁸ proposed using the negotiation values obtained by each node in such a solution as a structural power measure; see also Easley and Kleinberg¹⁰ (chapter 12) for a similar interpretation. According to the experiments we conducted for this article, this interpretation might seem opinable, but further investigation is necessary to gain a solid conclusion.

Conclusion

We proposed a theory on power in the context of networks. The philosophy underlying our notion of power maintains that an actor is powerful if it is connected with many powerless actors. This thesis has its roots and applications mainly in sociology and economics and traces an historical parallel with its celebrated linear counterpart, namely eigenvector centrality.¹²

The virtues of our definition of power are: it is a simple, elegant, and understandable measure; it is theoretically well-grounded and directly related to the well-studied balancing problem, making it possible to borrow results and techniques from this setting; the formulation is not

parametric; and it is global (the power of a node depends on the entire network) and can be approximated with a simple local measure—the sum of reciprocals of node degrees—that has a game-theoretic interpretation and can be efficiently computed on all networks. The definition has limitations as well, mainly that an exact solution exists only on the class of totally supported networks and is not immediately normalizable, so care is needed when comparing power values for nodes in different networks. ■

References

1. Bayati, M., Borgs, C., Chayes, J., Kanoria, Y., and Montanari, A. Bargaining dynamics in exchange networks. *J. Econ. Theory* 156 (2015), 417–454.
2. Bonacich, P. Power and centrality: A family of measures. *Am. J. Sociol.* 92, 5 (1987), 1170–1182.
3. Bozzo, E., Franceschet, M., and Rinaldi, F. Vulnerability and power on networks. *Network Sci.* 3, 2 (2015), 196–226.
4. Brown, J.B., Chase, P.J., and Pittenger, A.O. Order independence and factor convergence in iterative scaling. *Linear Algebra Appl.* 190 (1993), 1–38.
5. Brualdi, R.A. Matrices of 0s and 1s with total support. *J. Combin. Theory Ser. A* 28, 3 (1980), 249–256.
6. Brualdi, R.A. and Ryser, H.J. *Combinatorial Matrix Theory*, Vol. 39 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, Cambridge, U.K., 1991.
7. Cook, K.S., Emerson, R.M., Gillmore, M.R., and Yamagishi, T. The distribution of power in exchange networks: Theory and experimental results. *Am. J. Sociol.* 89, 2 (1983), 275–305.
8. Cook, K.S. and Yamagishi, T. Power in exchange networks: A power-dependence formulation. *Social Networks* 14 (1992), 245–265.
9. Csima, J. and Datta, B.N. The DAD theorem for symmetric nonnegative matrices. *J. Combin. Theory* 12, 1 (1972), 147–152.
10. Easley, D. and Kleinberg, J. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, New York, 2010.
11. Emerson, R.M. Power-dependence relations. *Am. Sociol. Rev.* 27, 1 (1962), 31–41.
12. Franceschet, M. PageRank: Standing on the shoulders of giants. *Commun. ACM* 54, 6 (2011), 92–101.
13. Kleinberg, J. and Tardos, E. Balanced outcomes in social exchange networks. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing* (2008), 295–304.
14. Knight, P.A. and Ruiz, D. A fast algorithm for matrix balancing. *IMA J. Numer. Anal.* 33, 3 (2013), 1029–1047.
15. Nash, J. The bargaining problem. *Econometrica* 18 (1950), 155–162.
16. Rochford, S.C. Symmetrically pairwise-bargained allocations in an assignment market. *J. Econ. Theory* 34, 2 (1984), 262–281.
17. Sinkhorn, R. and Knopp, P. Concerning nonnegative matrices and doubly stochastic matrices. *Pac. J. Math.* 21 (1967), 343–348.

Enrico Bozzo (enrico.bozzo@uniud.it) teaches numerical algorithms in the Department of Mathematics, Computer Science, and Physics at the University of Udine, Udine, Italy.

Massimo Franceschet (massimo.franceschet@uniud.it) teaches network science and generative art in the Department of Mathematics, Computer Science, and Physics at the University of Udine, Udine, Italy.

Looking at the mysteries of evolution from a computer science point of view yields some unexpected insights.

BY ADI LIVNAT AND CHRISTOS PAPADIMITRIOU

Sex as an Algorithm

The Theory of Evolution Under the Lens of Computation

LOOK AROUND YOU, and you will be stunned by the work of evolution. According to Nobel Laureate Jacques Monod, a strange thing about evolution is that all educated persons think they understand it fairly well, and yet very few—if any, one may grumble—actually do. Understanding evolution is essential: “Nothing in biology makes sense except in the light of evolution,” famously said the eminent 20th century biologist Theodosius Dobzhansky. And evolution is closer to home than black holes and other mysteries of science—it feels almost like your family history.

More so than most scientific fields, the theory of evolution has a sharp beginning: the publication

of Charles Darwin’s *The Origin of Species* in 1859.⁸ But of course nothing is that simple: during the first half of the 19th century, several scientists were convinced the diversity of life we see around us must be the result of evolution (see the sidebar on Charles Babbage and the accompanying figure). Darwin’s immense contribution lies in three things: His identification of natural selection as the engine of evolution; his articulation of the common descent hypothesis, stating that different species came from common ancestors, and further implying that all life came from a common source; and the unparalleled force of argument with which he empowered his theory. But of course *The Origin* was far from the last word on the subject: Darwin knew nothing about genetics, and had no clue about the role of sex in evolution, among several important gaps. On the ultimate reason for sex, for instance, he wrote, “the whole subject is as yet hidden in darkness.”

Mathematics has informed the theory early on. Mendel discovered genetics by appreciating mathematical patterns in the ratios of sibling pea plants exhibiting different characteristics and model building. When his laws were rediscovered 40 years later, their discrete nature was misunderstood as being at loggerheads with Darwin’s

» key insights

- Sexual reproduction is nearly ubiquitous in nature. Yet, despite a century of intense research, its evolutionary role and origin is still a mystery.
- Recent research at the interface of evolution and CS has revealed that evolution under sex possesses a surprising and multifaceted computational nature: It can be seen as a coordination game between genes played according to the powerful Multiplicative Weights Update Algorithm; or as a randomized algorithm for deciding whether genetic variants perform well across all possible genetic combinations; it allows mutation to process and transmit information from transient genetic combinations to future generations; and much more.
- Computational models and considerations are becoming an indispensable tool for unlocking the secrets of evolution.

40X

SEX

continuous perception of traits. A deep scientific crisis raged for two decades, and was eventually resolved with the help of mathematics: discrete alleles (see the glossary) can result, through cumulative contributions, in continuous phenotypic traits. During the 1920s and the next two decades, R.A. Fisher, J.B.S. Haldane, and S. Wright developed mathematical equations for predicting how the genetic composition of a population changes over the generations. This mathematical theory of population genetics is introduced briefly in the sidebar “The Equation that Reconciled Darwin and Mendel.” It is key to what is called the “modern evolutionary synthesis”—the 20th-century view of evolution, because it proposed one way of unifying Darwinism and Mendelism.

During the near-century since then, the study of evolution has flourished into a mature, comprehensive, and prestigious scientific discipline, while over the past two decades it has been inundated by a deluge of molecular data, a vast scientific gold mine that informs—and often challenges—its tenets. And yet, despite the towering accomplishments of modern evolutionary biology, there are several important questions that are beyond our current understanding:

► **What is the role of sex in evolution?**

Reproduction with recombination is almost ubiquitous in life (even bacteria exchange genetic material), while obligate asexual species appear to be rare evolutionary dead ends. And yet there is no agreement among the experts as to what makes sex so advantageous.

► **What exactly is evolution optimizing, if anything?** With all the evolution-inspired optimization heuristics coded by computer scientists (as we discuss next), this question—also very much contemplated by biologists over the decades—comes to the fore.

► **The paradox of variation.** Genetic variation in humans, and in many other species, is much higher than the theory predicted due to selection; and assuming the variation is neutral has its problems too.

► **Are mutations completely random?** We know they are far from uniformly distributed in the genome, but can they be the results of elaborate genetic mechanisms?



Not only is sex essentially universal, but it seems to be very much center stage in life, the basis of a fantastic variety of behavior and structure.



As we recount in this article, recent joint research by computer scientists and biologists, bringing ideas and concepts from computation into biology, has made quite unexpected progress in these questions. Additional background and literature in the online appendix is available in the ACM Digital Library (dl.acm.org) under Source Material.

Evolution and Computer Science

Over the past 70 years, computer scientists, starting with von Neumann,³⁷ have been inspired and intrigued by evolution. During the 1950s, computer scientists working in optimization developed local search heuristics: Start with a random solution and repeat the following step: If there is a “mutation” of this solution that is better than the current one, change to that, until a local optimum is discovered. By “mutation” (much more often the word “neighbor” is used), we mean a solution differing from the present one in a very small number of features; in the traveling salesman problem, for example, a mutation could change two, or three, edges of the tour to form a new tour. This process is repeated many times from random starts, a stratagem that can be seen as a sequential way of maintaining a population (see Papadimitriou and Steiglitz³² for a survey on the 1980s).

This basic idea of local search was enhanced in the 1980s by a thermodynamic metaphor,²⁰ to help the algorithm escape local optima and barriers: *Simulated annealing*, as this variant of local search is called, allows the adoption of even a disadvantageous mutation, albeit with a probability decreasing with the disadvantage, and with time. A further variant called *go with the winners*¹ is closer to evolution in that it keeps a population of solutions, teleporting the individuals that are stuck at local optima to the more promising spots. Notice that all these heuristics are inspired by asexual evolution (no recombination between solutions happens); heuristics of this genre have been used successfully in many realms of practice, and there are several practically important hard problems, such as graph partitioning, for which such heuristics are competitive with the best known.

During the 1970s, a different family of heuristics called genetic algorithms was proposed by John Holland.¹⁴ A population of solutions evolves through mutations *and* recombination. Recombination is much more difficult to apply to optimization problems, because it presupposes a genetic code, mapping the features of the problem to recombinable genes (to see the difficulty, think of the traveling salesman problem: tours can mutate, but they cannot recombine). The evolutionary fitness of each individual solution in the population (that is, the number of children it will have) is proportional to the quantity being maximized in the underlying problem. After many generations, the population will presumably include some excellent solutions. Holland's idea had instant appeal and immense following, and by now there is a vast bibliography on the subject (see, for example, Mitchell²⁸ and Goldberg¹²). The terms *evolutionary algorithms* and *evolutionary computation* are often used as rough synonyms of “genetic algorithms,” but often they describe more general concepts, such as the very interesting algorithmic work—also categorized as research in artificial life—whose purpose is not to find good solutions to a practical optimization problem, but instead to understand evolution in nature by exposing novel, complex evolutionary phenomena in silico, for an example, see Jong.¹⁶

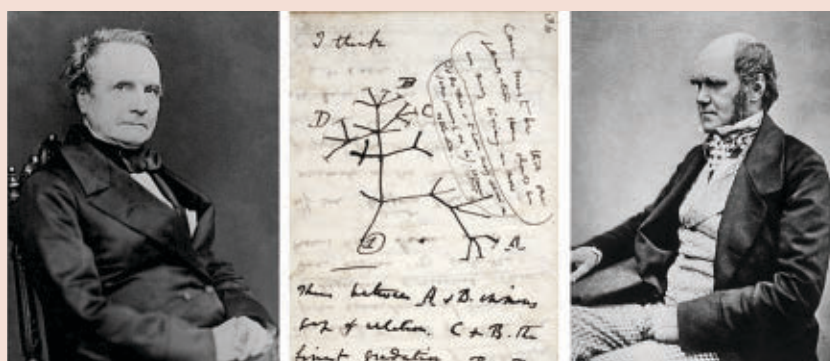
Coming back to the genetic algorithms, several successes in solving actual optimization problems have been reported in the literature, but the general impression seems to prevail that genetic algorithms are far less successful in practice than local search and simulating annealing. If this is true, it is quite remarkable—a great scientific mystery—because in nature the exact opposite happens: Recombination is successful and ubiquitous, while obligate asexual reproduction is extremely rare and struggling.

The authors started collaborating on a computational understanding of evolution in 2006, precisely in order to investigate this mystery, and we recount our findings in this article. At exactly the same year, Leslie Valiant first wrote on his theory of evolvability, another attempt at understanding evolution in computational terms.³⁶ Valiant sees evolution as an approximation of an

Charles Babbage on Evolution

Charles Babbage, the grandfather of computer science, was undoubtedly the first to think of evolution in algorithmic terms. In 1837—22 years before the publication of *The Origin of Species* and the same year that Darwin jotted down a sketch of the tree of life in his notebook (see the figure below)—Babbage wrote the Ninth Bridgewater Treatise.⁴ The treatise was ninth in a series of philosophical essays on the subject of the deity, in which he refers extensively to his Difference Engine and how it can be programmed to go on for days calculating unimaginably large integers, with no further intervention by the programmer. Babbage used this image to argue that a benevolent creator can design not necessarily species, but an algorithm (he did not use the word) for creating species.

Portraits of Charles Babbage (left) and Charles Darwin (right). In the middle is Darwin's sketch showing the principle of common descent leading to a branching pattern.



Glossary

Allele: Variant of a gene. Some are known to affect our susceptibility to diseases.

Diploidy: The state of having two instances of each chromosome, and thus each gene, in a cell. Organisms with one instance of each chromosome and gene are called haploid.

Fitness: The ability of an individual to survive and reproduce in its natural environment, manifested in its expected number of surviving offspring.

Gene: A unit of heredity and a region of the DNA that encodes a functional product. It is thought that humans have more than 20,000 of these. However, now that coding is known to be far more complex than originally thought, it is no longer clear how to define these units and their boundaries.

Genotype: The whole or a part of the genetic make-up of an individual or that is common to a group of individuals.

Heterozygous: An individual having two different alleles of a certain gene.

Mutation: A change in the hereditary material.

Phenotype: The characteristics of an individual other than its genetic code.

Recombination: In sexual reproduction, taken broadly, this term means that the genetic material of an offspring is a bricolage of the genetic material of its parents, due to both the independent assortment of chromosomes during the halving of chromosome numbers and the crossing over of chromosomes—the shuffling of segments between two corresponding chromosomes (in humans, typically 2–3 per chromosome)—that occur in the generation of gametes. More generally, DNA recombination in the sense of exchange of genetic material between two DNA molecules or between segments of the same DNA molecule is an important part of mutational mechanisms.

The Equation that Reconciled Darwin and Mendel

After Fisher showed how a continuous trait can result from adding up the contributions of multiple discrete Mendelian factors, he, Wright, and Haldane established the classic population genetic framework for studying how a population of genotypes evolves from one generation to the next. A central idea is fitness, a mathematical articulation of Darwin's conception of evolutionary advantage, a useful summary of an organism's phenotype. Formally, this fitness of a genotype can be thought of as the number of surviving offspring an organism with this genotype will have, in expectation. For simplicity, imagine a haploid organism (one copy of each gene) in which fitness depends only on two genes, and these genes come in three alleles each. The fitness function of this species is perhaps captured by this 3×3 table and denoted w_{ij} : $i = 1; 2; 3; j = 1; 2; 3$:

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	1.05	1.00	0.96
$i = 2$	1.01	1.03	1.02
$i = 3$	0.94	1.02	0.99

Suppose that at some point in time, in the t th generation of this species, the nine genotypes have these frequencies, denoted P_{ij}^t :

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	0.1111	0.1111	0.1111
$i = 2$	0.1111	0.1111	0.1111
$i = 3$	0.1111	0.1111	0.1111

Assume reproduction by recombination (sex); this means that a child inherits each allele from either of its parents; assume that it inherits it from either with equal probability, independently for the two genes (this holds, for example, if the two genes reside in different chromosomes). Then the genotype frequencies in the next generation, P_{ij}^{t+1} , to the fourth decimal place, will be as shown in this table:

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	0.1164	0.1109	0.1064
$i = 2$	0.1120	0.1142	0.1131
$i = 3$	0.1042	0.1131	0.1098

while the frequencies in 10 and 100 generations will be as follows:

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	0.1210	0.1226	0.0919
$i = 2$	0.1263	0.1469	0.1171
$i = 3$	0.0825	0.1085	0.0831

in 10 generations

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	0.0848	0.1409	0.0163
$i = 2$	0.2320	0.4424	0.0542
$i = 3$	0.0086	0.0185	0.0022

in 100 generations

The formula for going from the P_{ij}^t s to the P_{ij}^{t+1} s is the following:

$$P_{ij}^{t+1} = \frac{w_{ij}}{\bar{w}} \left(\frac{1}{2} P_{ij}^t + \frac{1}{2} \sum_i P_{ii}^t \sum_k P_{kj}^t \right),$$

where $\bar{w}$ is the sum of the numerators for all ij s. Of course, this equation is based on certain crucial assumptions: It is assumed that generations do not overlap (as if all members of one generation procreate simultaneously just before death), that the population is so large that it can be thought of as being infinite, that mating is completely random and without separate sexes, and that expectations can be added and divided with impunity. However, for the sake of simplicity, these assumptions have been used very widely and successfully in evolutionary theory for many decades. So far we have been assuming sexual reproduction. If this species were asexual, the next generation, the 10th and 100th hence would be like this:

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	0.1164	0.1109	0.1064
$i = 2$	0.1120	0.1142	0.1131
$i = 3$	0.1042	0.1131	0.1098

next generation

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	0.1693	0.1039	0.0691
$i = 2$	0.1148	0.1397	0.1267
$i = 3$	0.0560	0.1267	0.0940

in 10 generations

	$j = 1$	$j = 2$	$j = 3$
$i = 1$	0.7767	0.0059	0.0001
$i = 2$	0.0160	0.1135	0.0428
$i = 3$	0.0000	0.0428	0.0022

in 100 generations

and the equation is much simpler:

$$P_{ij}^{t+1} = \frac{w_{ij}}{\bar{w}} P_{ij}^t,$$

with $\bar{w}$ playing the same role. Naturally, much of evolution involves events not entering these equations at all: mutations creating new alleles, speciation events (the creation of a new branch in the tree of life with its own equations), and changes in the environment. These equations should be thought of as a mathematical description of a piece of the puzzle, still studied after nearly a century of research.

ideal fitness function by a polynomially large population of genotypes in polynomially many generations³⁵ through learning by mutations. Notably, there is no recombination (sex) in his theory, even though it can be added for a modest advantage.¹⁷ Several natural classes of functions are evolvable in this sense; in fact, functions susceptible to a limited weak form of learning called *statistical learning*.¹⁸ In the section “A Game between Genes,” we discuss another interesting connection between machine learning algorithms and evolution.

The Role of Sex

Sex is nearly universal in life: it occurs in animals and plants by the coming together of sperm and egg, in fungi by the fusion of hyphae, and even in bacteria.³⁴ Two bacterial cells can pair up, for example, and build a bridge between them through which genes are transferred. Many species engage in asexual reproduction, or in selfing, at some times, but also engage in sexual reproduction at other times, keeping their genotypes well shuffled. In contrast, species that do not exchange genes in any form or manner, called “obligate asexuals,” are extremely rare, inhabiting sparse, recent twigs of the tree of life, coming from sexual ancestral species that lost their sexuality, and heading toward eventual extinction without producing daughter species.²²

Not only is sex essentially universal, but it seems to be very much center stage in life, the basis of a fantastic variety of behavior and structure: from bacterial conjugation to the intense molecular machinery of meiosis (cell division producing gametes), from flower coloration to bird courtship dances, from stag fights all the way to the drama of human passion, much of life seems to revolve around sex. So why? What role might sex play in evolution?

One common answer is that sex generates vast genetic diversity, and hence it must help evolution. But, just as sex puts together genetic combinations, it also breaks them down: a highly successful genotype will be absent from the next generation, as children inherit half their genes from each parent. To say the role of sex is to create particular, highly favorable genetic combinations is like watching a man catch fish only to toss them back

to sea, and concluding that he wants to bring food to his family's table. (Incidentally, the designers of genetic algorithms are well aware of this downside of sex, and often allow the most successful individuals into the next generation, a stratagem known as “elitism,” which however cannot be easily imitated by nature.)

Evolutionary theorists have labored for about a century to find other explanations for the role of sex in evolution, but all 20th century explanations are valid only under specific conditions, contradicting the prevalence of sex in nature.^{9,a}

This is not a small problem. Imagine, for example, that even though much of the terrestrial world is green, we had no clue why leaves exist. That would have been a pretty big gap in our understanding of nature. Not knowing the role of sex is an even bigger gap, because far more life forms exchange genes than photosynthesize. It is no wonder the role of sex has been called “the queen of problems” in evolutionary biology.⁶

Since sex breaks down genetic combinations, it has been mainly thought in evolution that effective selection acts on individual alleles,³⁸ that is, each (non-neutral) allele is either beneficial or detrimental on its own. According to this line of thinking, two main forces drive allele frequencies: selection acting on alleles as independent actors (where alleles are often assumed to be making additive contributions to fitness), and random genetic drift (chance sampling effects on allele frequency, as discussed previously). The interaction between alleles, within and between loci—even though it has been of interest in population genetics from the start^{10,39}—has played a secondary role, often being treated as a mere correction to the above, under the term “epistasis.” A few years ago, while working with biologists Marcus Feldman and Jonathan Dushoff and computer scientist Nicholas Pippenger, we asked whether interactions between alleles could be crucial to the understanding of the role of sex in a yet unexplored manner.^{23–26} Based on the

standard equations used to describe how genotype frequencies change over generations (see the Darwin and Mendel sidebar), we demonstrated an important difference between sex and asex: In asexual evolution, the best combination of alleles always prevails. In the presence of sex, however, natural selection favors “mixable” alleles, those alleles that, even though they may not participate in any truly great genetic combinations, perform adequately across a wide variety of different combinations.^{23–26} To put it differently, in the hypothetical three-by-three fitness landscape in the sidebar, the winner of asexual evolution will be the largest entry of the fitness matrix (in this case, 1.05). In contrast, sexual evolution will favor, roughly speaking, those alleles (rows and columns) with larger average value; where “average” takes into account the prevalence of these genotypes in the population, as we will explore.^b

Sex as a Randomized Algorithm

One of the most central and striking themes of algorithms research in the past few decades has been the surprising power of randomization.²⁹ Paradoxically, evoking chance is often the safest and most purposeful and effective way of solving a computational problem. For one, it helps avoid the worst case, as in Quicksort. Second, sampling from a distribution D helps decide between competing hypotheses about D : Randomized algorithms for software testing, as well as for testing primality, or validity of polynomial identities, are like this.

Evolution under sex can be seen as an instance of a randomized algorithm of the latter type. Suppose we want to design a hypothetical evolutionary experiment for determining whether a new allele of a particular gene performs better than its alternative, across all genetic combinations. If the population is asexual, this could be done by inserting this mutation in the genome of one individual, and gauging the lineage thus founded to see if it thrives. This kind of sampling

a See the appendix available in the ACM Digital Library (dl.acm.org) under Source Material for a more extensive bibliography on this and other subjects.

b The original paper²⁴ refers to the unweighted average fitness as mixability, instead of the more natural average weighted by genotype frequency.

is very inefficient, because we sample from a small pool (the genotypes that happen to be available in the population), and must repeat the insertion many times—in many individuals. But if the population is sexual, then by inserting the mutation once, after $\log n$ generations, where n is the number of genes with which this particular allele interacts,³³ we will be sampling from all *possible* genetic combinations that could in principle be constructed. Sex enables evolution to sample quickly from the entire space of genetic combinations, in the distribution under which they appear in the population. What is more, evolution under sex not only decides among the competing hypotheses (which allele performs better), but also *implements* this decision (eventually, and with high probability, it will fix the winner).

Finally, for yet another take on explaining the ubiquity of sex in life, in terms and concepts familiar to computer scientists, view The Red-Blue Tree Theorem sidebar.

A Game between Genes

The search in population genetics for a quantity that is optimized by natural selection has a long history. Fisher wanted a theory for evolution with a mathematical law as clean and central as the second law of thermodynamics,¹⁰ while Wright pointed out that the frequency of an allele in a diploid locus changes in the direction that increases the population's mean fitness.⁴⁰ Later, investigators tried their hand again at looking for a Lyapunov function that will describe evolution, albeit with little success.

Our search for an analytical maximization principle involving mixability ended with a surprise: We did not answer the question “What is evolution optimizing?” but, perhaps more interestingly, we identified the quantity that each gene seems to be optimizing during evolution under sex. Together with Erick Chastain and Umesh Vazirani,⁷ we focused on the standard equations described in the Darwin Mendel sidebar, in a particular evolutionary regime known as *weak selection*.³⁰ Weak selection is the widely held assumption that fitness differences between genotypes are small. The fitness of a genotype g in this regime

is written $F_g = 1 + s \Delta_g$, where s is small and Δ_g is the *differential fitness* of the genotype, ranging in $[-1, 1]$.

Working with the weak selection assumption, and after some algebraic manipulation, we noticed the equations of the evolution of a population under sex are mathematically equivalent to a novel process, which entails an entirely different way of looking at evolution:

- The process is a game between the genes of the organism. Recall that a game is a mathematical model of the interaction between several players, each player having a set of available actions or pure strategies, and a utility function: the objective the player is trying to maximize in the game, a mapping from choices of actions, one for each player, to a real number. The game here is repeated, that is, the same precise game is played over and over for many rounds, with each round corresponding to a generation of the population.

- The available actions of each gene/player correspond to the gene's alleles.

- At every generation, each gene chooses and plays a mixed strategy: it randomizes over its actions, that is to say, its alleles. The probability assigned to each allele in this mixed strategy corresponds to the frequency of the allele in the population during this generation.

- The intricacy of game theory is mostly due to conflicts between the objectives of the players. But in the game played by the genes there are no conflicts—this is not Dawkins's selfish gene metaphor: All players have the same utility function, which is the fitness of the genotype resulting from the players' choices. Games of identical utilities are called coordination games in game theory, and are the simplest possible kind. They are of interest only in cases in which the players are cognitively weak, or cannot communicate effectively.

- In a repeated game, the players must update their mixed strategies from one round to the next, based on the outcome of the previous rounds. Here lies the biggest surprise: The update rule used by the genes in this game is identical to a venerable learning algorithm, well known to computer scientists for its prowess in work-

ing surprisingly well in a multitude of difficult problems and contexts: the *multiplicative weights* update algorithm (MWU),² also known in machine learning as “no-regret learning” or “hedge” (see more in box “The Experts Problem” in the online appendix). MWU changes the frequencies x_i of the i -th allele of the gene as follows

$$x_i \leftarrow x_i (1 + s m_i), \quad (1)$$

where m_i is the expected differential fitness, positive or negative, of the i -th allele in the current gene pool. This quantity m_i is a measure of what we have called the *mixability* of allele i , its ability to form fit combinations with alleles of other genes in the current genetic mix. To summarize, at each generation in sexual evolution, each gene boosts the frequency of each of its alleles by a factor that increases with the mixability of this allele in the current generation. Naturally, the quantities resulting from the equation are normalized appropriately so as to add to one.

This is a completely new way of looking at evolution. And it is a productive view, because it gets more interesting: Let us look back at the update Equation 1 and ask once again the question: This choice of the new probabilities for the alleles by the gene, is it optimizing something? For once, the answer is very clean: Yes, the choice of allele frequencies by the gene shown in Equation 1 optimizes the following function, specific to this gene, of the allele frequencies:

$$\Phi(x) = \sum_i x_i M_i - \frac{1}{s} \sum_i x_i \ln x_i. \quad (2)$$

Here M_i denotes the cumulative relative fitness of allele i , that is, the sum of the m_i in Equation 1 over all generations up to and including $t - 1$. It is easy to notice that Φ is a strictly concave function, and thus has one maximum, and this maximum can be checked by routine calculation to be exactly the new frequencies as updated in Equation 1! Now notice that the second term of Φ is plainly the entropy of the distribution x , a well-known measure of a distribution's diversity.

There is much that is unexpected and evocative here, but perhaps most surprising of all is that this radically new interpretation of evolution was lurking for almost a century so close to the surface of these well-trodden equa-

tions. That an algorithm as effective as MWU is involved in evolution under sex is also significant. It was pointed out in a commentary on our paper⁵ that the MWU is also present in asex. Indeed, asexual evolution can be trivially thought of as MWU helping nature select genotypes. However, our point is that in sexual evolution, the picture is far more sophisticated and organic, occurring deeper in the hierarchy of life: Individual genes interact, each “managing its investments in alleles” using MWU, in a context created by the other genes and, of course, by the environment.

The function Φ is a rare explicit optimization principle in evolution. The second term, and especially its large constant coefficient (recall that the selection strength s is small, and $|m_i| \leq 1$) suggests that attention to diversity is an important ingredient of this mechanism, a remark that may be relevant to the question of how genetic diversity is maintained. But there is a mystery in the non-diversity terms: the cumulative nature of the fitness coefficients M suggests that performance during any previous generation is as important as the current generation for the determination of the genetic make-up of the next generation. How can this be?

This surprising connection between evolution and algorithms, through game theory and machine learning, as well as the maximization principle Φ , are very recent, and their full interpretation is a work in progress.

The insights about sex as we have discussed shed some light on the mystery of genetic algorithms, as explained previously. There is a mismatch between heuristics and evolution. Heuristics should strive to create populations that contain outstanding individuals. In contrast, evolution under sex seems to excel at something markedly different: at creating a “good population.” So, it is small wonder that genetic algorithms are not the best heuristics around. On the other hand, these insights also suggest that genetic algorithms may be valuable when the robustness of solutions is sought, or when the true objective is unknown or uncertain or subject to change.

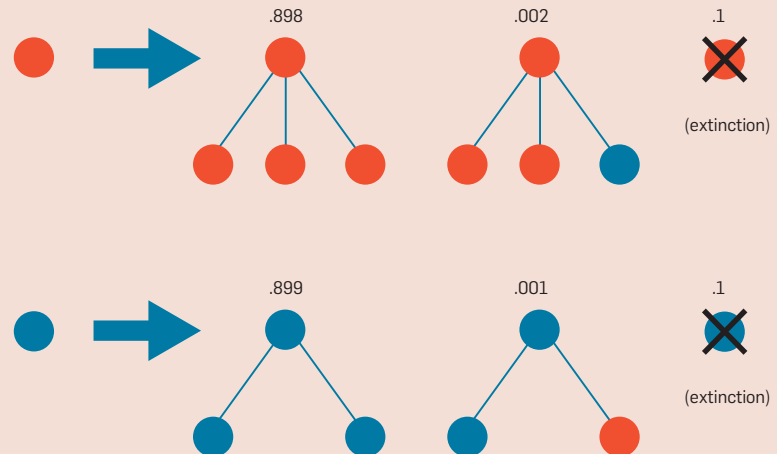
Are Mutations Random?

What is a mutation? Point mutations (changes in a single base such as an A turning into a C) are only part of the sto-

The Red-Blue Tree Theorem

A completely different take on the ubiquity of sex in evolution, based on unpublished work, uses concepts and images very familiar to computer scientists.

Imagine a tree with nodes of two kinds: red and blue. The tree starts from a root, which is blue, and is generated through a random process. At each step (depth of the tree), the nodes at this depth are replaced as shown here, each independently of the others according to the probabilities as shown below.



This is a very crude model for generating the tree of life (the probabilities here are indicative, and not supposed to reflect some biological reality). Red nodes are sexual species, and blue nodes are distinct asexual types (the term “species” does not perfectly apply to the latter). While there are reasons to believe that life started *sexually*,²² and not with a well-delineated species as exist today, we make the assumption that the tree begins with a blue node, which for our current purposes is conservative. According to the chosen parameters, sexuals and asexuals become extinct with equal probability. If extinction does not happen, sexual species branch out 50% more. With small probability, one of the new species can be of the other kind.

How would the leaves of the tree (that is, the life we see around us) look like, many steps down? After only 50 steps, the expected ratio of asexuals to sexuals is roughly 2:1000 (the probability of the top middle event).

Asexual nodes are rare! And, to a computer scientist, this stands to reason: We know that branching factors (in trees, computer virus propagation, recursive algorithms, among others) can dwarf in ultimate significance all other ingredients of a situation. Interestingly, this simple model immediately explains the main characteristics of the distribution of sexuality: It predicts that asexual forms are rare, sparsely distributed across the tree of life, and recently derived from sexual ancestors. Empirical evidence shows that all three points are true.²²

The precise relevant theorem is the following²³ (it is a consequence of well known properties of branching, see Athreya and Ney³):

Consider a continuous random process of generating a red-blue tree in which the rates of the events $B \rightarrow BB$, $B \rightarrow BR$, $B \rightarrow \text{extinction}$, $R \rightarrow RR$, $R \rightarrow BR$ and $R \rightarrow \text{extinction}$ are, respectively, σ_A , β , ε_A , σ_S , α , and ε_S . Let $\delta_S = \sigma_S - \varepsilon_S$, $\delta_A = \sigma_A - \varepsilon_A$, and $\delta_S > \delta_A$. Suppose that $\alpha > 0$ and $\beta > 0$ be each small in comparison to $\delta_S - \delta_A$. Then the asymptotic ratio of asexual to sexual species at the leaves is $\approx \alpha / (\delta_S - \delta_A)$.

This argument suggests that, conceivably, a key advantage of sex might be its “branching factor” in the tree of life. Even if sex were a burden for species and organisms (which is at least a part of the truth), the mere fact that it promotes diversification and an exploration of new evolutionary niches might be enough for this strange trait to dominate the planet.

ry, as many important mutations are rearrangements of small stretches as well as large swaths of DNA: duplications, deletions, insertions, inversions, among others.¹³ For a long time it has been believed that mutations are the results of accidents such as radiation damage or replication error. But by now we have a

deluge of evidence pointing to involved biological mechanisms that bring about and affect mutations.²²

We know, for example, that the chance of a mutation varies from one region of the genome to another and is affected by both local and remote DNA.²² Nearly a quarter of all point mutations in humans

happen at a C base which comes before a G after that C is chemically modified (*methylated*);¹¹ methylation is known to be the result of complex enzymatic processes. As to rearrangement mutations, there are powerful agents of mutagenicity (*creation of mutations*) in the genome, such as transposable elements: DNA sequences prone to “jump” from one place of the genome to another, carrying other DNA sequences with them.¹³ A key step in mammalian pregnancy (*decidualization*), for instance, was the result of massive evolutionary rewiring of about 1,500 genes mediated in part by transposable elements.²⁷ Furthermore, the genetic sequences that participate in a rearrangement mutation are likely to be functionally related, since they are likely to be close to each other in 3D space and to bear sequence similarity, both of which allow interaction through recombination-based mutational mechanisms (see Livnat²² and references therein). Indeed, the same machinery that effects sexual recombination is also involved in mutations, and in fact produces different types of rearrangement mutations, depending on the genetic sequences that are present.¹³ Finally, different human populations undergo different kinds of mutations resulting in the same favorable effect, such as malaria resistance, suggesting that genetic differences between populations cause differences in mutation origination.²²

The idea that mutations may be non-accidental is still confronted with suspicion due to the legacy of Lamarck, who believed around 1800 that organisms can sense, through interaction with the environment, what is needed for an evolutionary improvement, and are able to make the correct heritable change. Since in the light of modern biology this seems impossible—to a computer scientist it sounds like reversing a one-way function, or a hash function—the accidental mutation notion prevailed. But in science one must not assume that the only relevant alternatives are the familiar, inside-the-box ones. Mutations are random—but it may be more productive to think of them as random in the same way that the outputs of randomized algorithms are random. Indeed, mutations are biological processes, and as such they must be affected by the interactions between genes. This new conception of heredity is exciting, because it creates



There is a mismatch between heuristics and evolution. Heuristics should strive to create populations that contain outstanding individuals. Evolution under sex seems to excel at something markedly different: creating a “good population.”



an image of evolution that is even more explicitly algorithmic. It also means that genes interacting in one organism can leave hereditary effects on the organism’s offspring.²² It no longer matters that a lucky genetic combination created by sex is doomed to vanish from the face of the earth (that the fisherman of our earlier metaphor throws the fish away): It may have achieved a lasting effect on the population through mutagenicity. Finally, the biological mechanisms affecting mutations may themselves evolve.

On the Preservation of Variation

If there is one idea that permeates all the various aspects of computational thinking about evolution, as explained in the past sections, it is this: Interactions between genes are crucial for understanding evolution. Gene interactions also come up in our recent work on the following question: How is variation within a species preserved?

Classic data indicates that, for a large variety of plants and animals taken together, the percentage of protein-coding loci that are polymorphic (in the sense that more than one protein variant exists that appears in more than 1% of the individuals in a population), and the percentage of such loci that are heterozygous in an individual, average around 30% and 7% respectively.³¹ Such a number is far greater than could be explained by traditional selection-based theories.²¹ Genetic variation is fed by mutations, and, according to the equations of populations genetics, it is decreased by fixation, the eventual triumph of one allele and the extinction of all other alleles of the same gene. In much of the discourse on the subject, selection is assumed to act on individual genes, and therefore fitness is additive. The equations tell us that fixation will happen after $O(\frac{1}{\Delta})$ generations, where Δ is the difference in relative fitness between the most fit and the second most fit allele. Fixation looks rather speedy.

In the spring of 2014, the Simons institute brought to Berkeley 60 biologists and computer scientists to exchange ideas on evolution. It was during that time the authors, with Costis Daskalakis and Albert Wu, explored how our understanding of the speed of fixation would be affected if one takes into account gene interactions. We approached the subject through some decades-old work on the

complexity of local search;¹⁵ this theory examines how difficult it is for a local search process to reach a local optimum, and the conclusion has in many cases been: “pretty hard.” By applying this point of view to selection on interacting genes, we showed that there are n -gene systems, in which the fitness is the sum total of contributions of certain pairs of alleles—that is, the next step beyond selection on single alleles—for which fixation takes a number of generations proportional to 2^n to happen. A stronger result can also be obtained under the well-accepted complexity assumption that local search is intractable in general (see Johnson et al.¹⁵ for details). The implication is that, if gene interactions are taken into account, fixation may take *much* longer than in the regime of selection on individual genes.

Does this insight explain the mystery of variation? Not yet, because our analysis so far has been disregarding two other powerful forces in evolution, besides mutation and selection, acting on variation: the finiteness of the population, and heterozygosity (a diploid organism carrying two different alleles of a gene.).

First, finiteness. Because the number of individuals carrying the alleles in question is finite, say N , the number of individuals carrying each allele at each generation evolves as in a kind of a random walk within the confines of $[0, N]$, and, ignoring selection, this results in fixation after $O(N)$ generations.¹⁹ Second, diploidy introduces the possibility of overdominance, in which organisms with two different alleles of a gene are more fit than organisms with two copies of one allele or two copies of the other. In overdominance, the equations of selection point to stable variation, with both alleles enjoying stably high frequency in the population.

How these three effects, of finite population, of heterozygosity, and of selection acting on combinations of alleles across loci, interact with one another is an important subject for further research.

Epilogue

A computer scientist marvels at the brilliant ways in which evolution has achieved so much: Systems with remarkable resource efficiency, reliability and survivability, adaptability to exogenous circumstances, let alone ingenious and

pristine solutions to difficult problems such as communication, cooperation, vision, locomotion, and reasoning, among so many more. One is tempted to ask: *What algorithm could create all this in just 10^{12} steps?* The number 10^{12} —one trillion—comes up because this is believed to be the number of generations since the dawn of life $3.5 \cdot 10^9$ years ago (notice that most of our ancestors could not have lived for much more than a day). And it is not a huge number: cellphone processors do many more steps in an hour.

Over the past decade, computer scientists and evolutionary biologists working together have come up with new insights about central open problems surrounding evolution—including, rather surprisingly, a proposed answer to the “algorithm” question—by looking at evolution from a computational point of view. And, of course many more questions, inviting similar investigation, were opened up in the process. 

Additional background and literature appears in an online appendix available with this article in the ACM Digital Library (dl.acm.org) under Source Material.

References

1. Aldous, D. and Vazirani, U. ‘Go with the winners’ algorithms. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science* (1994), 492–501.
2. Arora, S., Hazan, E. and Kale, S. The multiplicative weights update method: A meta-algorithm and applications. *Theory of Computing* 8, 1 (2012), 121–164.
3. Athreya, K. and Ney, P. *Branching Processes*. Springer, 1972.
4. Babbage, C. *The Ninth Bridgewater Treatise*. 2nd edn. John Murray, London, 1838.
5. Barton, N.H., Novak, S. and Paixão, T. Diverse forms of selection in evolution and computer science. In *Proceedings of the National Academy of Sciences* 111, 29 (2014), 10398–10399.
6. Bell, G. *The Masterpiece of Nature: The Evolution and Genetics of Sexuality*. University of California Press, Berkeley, CA, 1982.
7. Chastain, E., Livnat, A., Papadimitriou, C. and Vazirani, U. Algorithms, games, and evolution. In *Proceedings of the National Academy of Sciences* 111, 29 (2014), 10620–10623.
8. Darwin, C. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. Murray, London, 1859.
9. Feldman, M.W., Otto, P. and Christiansen, F.B. Population genetic perspectives on the evolution of recombination. *Annual Review of Genetics* 30 (1997), 261–295.
10. Fisher, R.A. *The Genetical Theory of Natural Selection*. The Clarendon Press, Oxford, U.K., 1930.
11. Fryxell, K.J. and Moon, W.-J. CpG mutation rates in the human genome are highly dependent on local GC content. *Molecular Biology and Evolution* 22 (2005), 650–658.
12. Goldberg, D. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.
13. Graur, D. and Li, W.-H. *Fundamentals of Molecular Evolution*. Sinauer Associates, Sunderland, MA, 2000.
14. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. U Michigan Press, 1975.
15. Johnson, D.S., Papadimitriou, C.H., and Yannakakis, M. How easy is local search? *J. Computer and System Sciences* 37, 1 (1988), 79–100.
16. Jong, K.A.D. *Evolutionary Computation: A Unified Approach*. MIT Press, Cambridge MA, 2006.
17. Kanade, V. Evolution with recombination. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, (2011), 837–846.
18. Kearns, M. Efficient noise-tolerant learning from statistical queries. *J. ACM* 45, 6 (1998), 983–1006.
19. Kimura, M. and Ohta, T. The average number of generations until fixation of a mutant gene in a finite population. *Genetics* 61, 3 (1969), 763.
20. Kirkpatrick, S., Gelatt, Jr., C.D. and Vecchi, M.P. Optimization by simulated annealing. *Science* 220 (1983), 671–680.
21. Lewontin, R.C. and Hubby, J.L. A molecular approach to the study of genic heterozygosity in natural populations; amount of variation and degree of heterozygosity in natural populations of *Drosophila pseudoobscura*. *Genetics* 54 (1966), 595–609.
22. Livnat, A. Interaction-based evolution: How natural selection and nonrandom mutation work together. *Biology Direct* 8, 1 (2013), 24.
23. Livnat, A., Feldman, M.W., Papadimitriou, C. and Pippenger, N. On the advantage to sexual species in diversification rates. Unpublished manuscript.
24. Livnat, A., Papadimitriou, C., Dushoff, J. and Feldman, M.W. A mixability theory for the role of sex in evolution. In *Proceedings of the National Academy of Sciences* 105, 50 (2008), 19803–19808.
25. Livnat, A., Papadimitriou, C. and Feldman, M.W. An analytical contrast between fitness maximization and selection for mixability. *J. Theoretical Biology* 273, 1 (2011), 232–234.
26. Livnat, A., Papadimitriou, C., Pippenger, N. and Feldman, M.W. Sex, mixability, and modularity. In *Proceedings of the National Academy of Sciences* 107, 4 (2010), 1452–1457.
27. Lynch, V.J., Leclerc, R.D., May, G. and Wagner, G.P. Transposon-mediated rewiring of gene regulatory networks contributed to the evolution of pregnancy in mammals. *Nature Genetics* 43 (2011), 1154–1159.
28. Mitchell, M. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, 1996.
29. Motwani, R. and Raghavan, P. *Randomized Algorithms*. Cambridge University Press, 1995.
30. Nagylaki, T., Hofbauer, J. and Brunovsky, P. Convergence of multilocus systems under weak epistasis or weak selection. *J. Mathematical Biology* 38, 2 (1999), 103–133.
31. Nevo, E., Beiles, A. and Ben-Shlomo, R. The evolutionary significance of genetic diversity: Ecological, demographic and life history correlates. *Lecture Notes in Biomathematics* 53 (1984), 13–213.
32. Papadimitriou, C. and Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity*. Dover, 1998.
33. Rabani, Y., Rabinovich, Y. and Sinclair, A. A computational view of population genetics. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, (1995), 83–92.
34. Stearns, S.C. and Hoekstra, R.F. *Evolution: An Introduction*. Oxford University Press, New York, 2005.
35. Valiant, L. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books, 2013.
36. Valiant, L.G. Evolvability. *J. ACM* 56, 1 (2009), 3.
37. Von Neumann, J. and A.W. Burks, A.W. Theory of self-reproducing automata. *IEEE Transactions on Neural Networks* 5, 1 (1966), 3–14.
38. Williams, G.C. *Adaptation and Natural Selection*, 8th edition. Princeton University Press, 1996.
39. Wright, S. Evolution in Mendelian populations. *Genetics* 16 (1931), 97–159.
40. Wright, S. The distribution of gene frequencies in populations. In *Proceedings of the National Academy of Sciences of the United States of America*, 23, 6 (1937), 307–320.

Adi Livnat (alivnat@univ.haifa.ac.il) is a Senior Lecturer in the Department of Evolutionary and Environmental Biology, and Institute of Evolution at the University of Haifa, Israel.

Christos Papadimitriou (christos@cs.berkeley.edu) is the C. Lester Hogan Professor in the Computer Science Division of the University of California at Berkeley.

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



Watch the authors discuss their work in this exclusive *Communications* video.
<http://cacm.acm.org/videos/sex-as-an-algorithm>

The future success of these systems depends on more than a Netflix challenge.

BY DIETMAR JANNACH, PAUL RESNICK,
ALEXANDER TUZHILIN, AND MARKUS ZANKER

Recommender Systems—Beyond Matrix Completion

THE USE OF recommender systems has exploded over the last decade, making personalized recommendations ubiquitous online. Most of the major companies, including Google, Facebook, Twitter, LinkedIn, Netflix, Amazon, Microsoft, Yahoo!, eBay, Pandora, Spotify, and many others use recommender systems (RS) within their services.

These systems are used to recommend a whole range of items, including consumer products, movies, songs, friends, news articles, restaurants and various others. Recommender systems constitute a mission-critical technology in several companies. For example, Netflix reports that at least 75% of its downloads and rentals come from their RS, thus making it of strategic importance to the company.^a

In some ways, the systems that produce these recommendations are remarkable. They incorporate

a variety of signals about characteristics of the users and items, including people's explicit or implicit evaluations of items. The systems process these signals at a massive scale, often under real-time constraints. Most importantly, the recommendations are of significant quality on average. In empirical tests, people choose the suggested items far more often than they choose suggested items based on unpersonalized benchmark algorithms that are based on overall item popularity.

In other ways, the systems that produce these recommendations are sometimes remarkably bad. Occasionally, they make recommendations that are embarrassing for the system, such as recommending to a faculty member an introductory book from the "for dummies" series on a topic she is expert in. Or, they continue recommending items the user is no longer interested in. Shortcomings like these motivate ongoing research both in industry and academia, and recommender systems are a very active field of research today.

To provide an understanding of the state of the art of recommender systems, this article starts with a bit of history, culminating in the million-dollar Netflix challenge. That challenge led to a formulation of the recommendation problem as one of matrix completion: Given a matrix of users by items, with item ratings as cells, how well can an algorithm predict the values in some

» key insights

- Recommender systems have become a ubiquitous part of our daily online user experience and support users in a variety of domains.
- Today, the scientific community operationalizes the research problem mainly on principles from information retrieval and machine learning, leading to a well-defined but narrow problem characterization.
- We briefly review the history of the field, report on the recent advances, and propose a more comprehensive research approach that considers both the consumer's and the provider's perspective.

a <http://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5-stars.html>



cells that are deliberately held out? However, algorithms with maximum accuracy at the matrix completion task are not sufficient to make the best recommendations in many practical settings. We will describe why, review some of the approaches that current research is taking to do better, and finally sketch ways of approaching the recommendation problem in a more comprehensive way in the future.

A Brief History

Many fields have contributed to recommender systems research, including information systems, information retrieval (IR), machine learning (ML), human-computer interaction (HCI), and even more distant disciplines like marketing and physics. The common starting point is that recommenda-

tions must be personalized or adapted to the user's situation, with different people typically getting different item suggestions. That implies maintaining some kind of user history or model of user interests.

Building user profiles: Information filtering roots. In many application domains, for example, in news recommendation, recommenders can be seen as classic information filtering (IF) systems that scan and filter text documents based on personal user preferences or interests. The idea of using a computer to filter a stream of incoming information according to the preferences of a user dates back to the 1960s, when first ideas were published under the term “selective dissemination of information.”¹⁷ Early systems used explicit keywords that were pro-

vided by the users to rank and filter documents, for example, based on keyword overlap counts. Later on, more elaborate techniques like weighted term vectors (for example, TF-IDF vectors) or more sophisticated document analysis methods like latent semantic indexing (LSI) were applied to represent documents, with corresponding representations of user interests stored as user models. Recommender systems based on these techniques are typically called “content-based filtering” approaches.

Leveraging the opinions of others. As early as 1982, then ACM president Peter J. Denning complained about “electronic (email) junk” and advocated the development of more intelligent systems that help to organize, prioritize, and filter the incoming

Recommendation as Matrix Completion

The recommendation problem viewed as a matrix completion problem as done in the Netflix Prize.

1. Given a sparse matrix M , create a matrix M' by randomly hiding a subset H of the known ratings.
2. Predict the values (H^*) of the hidden ratings using M' .
3. Assess the difference between the predicted and the true ratings using the Root Mean Square Error (RMSE).

RMSE: Given a vector of predictions H^* of length n and the vector containing the true values H , the RMSE is computed as

$$\sqrt{\frac{1}{n} \sum_{i=1}^n (H_i^* - H_i)^2}$$

M	Users/Items	I1	I2	I3		I100000		
	U1	4	3			2	x	Known ratings
	U2		2	5				Missing ratings
	U3	3	5			2		
	...							
	U10000	2	2	5				

M'	Users/Items	I1	I2	I3		I100000		
	U1	4	3			2	x	Known ratings
	U2		2	?				Missing ratings
	U3	3	5			?		Withheld ratings
	...							
	U10000	2	?	5				

H

Common techniques. The goal of *matrix factorization* techniques in RS is to determine a low-rank approximation of the user-item rating matrix by decomposing it into a product of (user and item) matrices of lower dimensionality (latent factors).

The idea of *ensemble methods* is to combine multiple alternative machine learning models to obtain more accurate predictions.

streams of information.⁸ One of his proposals included the idea to use “trusted authorities” that assess document quality; receivers would only read documents that surpass some defined quality level. In 1987, the “Information Lens” personal mail processing system was proposed.²⁵ The system was mainly based on manually defined filtering rules but the authors already envisaged a system where email receivers could endorse other people whose opinions they value. The number and strength of the endorsements would then prioritize incoming messages. The Tapestry email filtering system at Xerox PARC¹⁴ adopted a similar approach of employing user-specified rules. It also introduced the idea that some readers could classify (rate) messages and other readers could access this information, which was called “collaborative filtering” (CF).

In 1994, Resnick et al.³³ presented the GroupLens system, which contin-

ued the ideas of Tapestry and introduced a system component, the “Better Bit Bureau,” which made automated predictions about which items people would like based on a nearest-neighbor scheme. Other research groups working independently developed similar ideas.^{18,36} The idea was “in the air” that opinions of other people were a valuable resource and the race was on to turn the idea into practical results.

It works in e-commerce! In 1999, only five years after the first CF methods were proposed, Shafer et al.³⁵ reported on several industrial applications of recommender systems technology in e-commerce; for example, for the recommendation of books, movies, or music. Amazon.com is mentioned as one of the early adopters of recommendation systems. In 2003, Linden et al.²³ report that Amazon’s use of item-to-item CF techniques as a targeted marketing tool had a huge impact on its business in terms of click-through and conver-

sion rates. In the same report, several challenges in practical environments were discussed, in particular the problem of scalability and the need to create recommendations in real time.

The matrix completion problem. By this time, the research community had developed some standard concepts and terminology. The core element is the user-item rating matrix, as illustrated in the upper part of the accompanying sidebar. Rows represent users, columns represent items, and each cell represents a user’s subjective preference for an item, determined based on an explicit report (for example, 1–5 stars) or based on user behavior (for example, clicking, buying, or spending time on the item).

The user-item matrix is generally sparse: most users have not interacted with most items. One formulation of the recommender problem, then, is that of a matrix completion problem. That is, the problem is to predict what the missing cells will be, in other words, how will users rate items they haven’t rated yet?

With that formulation, it was natural to apply and adapt machine-learning techniques from other problem settings, including various forms of clustering, classification, regression, or singular value decomposition.^{3,4} Correspondingly, the community adopted evaluation measures from the IR and ML fields like precision/recall and the root mean squared error (RMSE) measure. These evaluation techniques withhold some of the known ratings, use the algorithm to predict ratings of those cells, and then compare the predicted ratings with the withheld ones. The availability of some common rating datasets, distributed by vendors and academic projects, enabled researchers to conduct bake-offs comparing the performance of alternative matrix filling algorithms against each other.

The Netflix Prize. Netflix, which saw the strategic value in improving its recommendations, supercharged the bake-off process for matrix completion algorithms in 2006. Netflix offered a million-dollar prize, a dataset for training, and an infrastructure for testing algorithms on withheld data. The training dataset included 100 million real customer ratings. The prize was for the first algorithm to

outperform Netflix's in-house system by 10% on RMSE (see the sidebar). Interest in this competition was huge. More than 5,000 teams registered for the competition and the prize was finally awarded in 2009. Substantial progress was made with respect to the application of ML approaches for the rating prediction task. In particular, various forms of matrix factorization as well as ensemble learning techniques were further developed in the course of the competition and proved to be highly successful.

Beyond Matrix Completion

At the conclusion of the Netflix Prize competition, it might have been plausible to think that recommender systems were a solved problem. After all, many very talented researchers had devoted themselves for an extended period of time to improve the prediction of withheld ratings. The returns on that effort seemed to be diminishing quite rapidly, with the final small improvements that were sufficient to win the prize coming from combining the efforts of many independent contestants.

However, it turns out that recommender systems are far from a solved problem. Here, we first give examples of why optimizing the prediction accuracy for held-out historical ratings might be insufficient or even misleading. Then we discuss selected quality factors of recommender systems not covered by the matrix completion task at all and give examples of recent research that goes beyond matrix completion.

Pitfalls of matrix completion setups. *Postdiction $\neq$ prediction.* Predicting held-out matrix entries is really predicting the past rather than the future. If the held-out rating entries are representative of the hidden rating entries, then the distinction does not matter. However, in many recommender settings, the held-out ratings are not representative of the missing ratings.

One reason is the missing ratings are generally not missing at random. Even for items that people have experienced, if rating requires any effort at all they are more likely to rate items that they love or hate rather than those that they feel lukewarm about. Moreover, people are more likely to try items they expect to like. For example, in one empirical study, researchers found that

ratings of songs randomly assigned to users had a very different distribution than ratings of songs users had chosen to rate.²⁶

As a result, algorithms that predict well on held-out ratings that users provided may predict poorly on a random set of items the user has not rated. This can mean that algorithms tuned to perform well on past ratings are not the best algorithms for recommending in the real world.⁷

In addition, the matrix completion problem setup is not suitable to assess the value of reminding users of items they have already purchased or consumed in the past. However, such repeated recommendations can be a desired functionality of recommenders, for example, in domains like music recommendation or the recommendation of consumables.

In the end, the standardized evaluation setup and the availability of public rating datasets made it attractive for researchers to focus on accuracy measures and the matrix completion setup and may have lured them away from investigating the value of other information sources and alternative ways of evaluating the utility of recommendations.

Today, a growing number of academic studies try to evaluate the performance of their methods using A/B tests on live customers in real industrial settings (for example, Dias et al.⁹ Garcin et al.,¹² and Gorgoglione et al.¹⁶). This is a very positive trend that requires cooperation from a commercial vendor who may not agree to make data publicly available, thus making it difficult for results to be checked or reproduced by others.

Not all items and errors are equally important. RMSE, the evaluation metric used in the Netflix Prize, equally weights errors of prediction on all items. However, in most practical settings items with low predicted ratings are never shown to users, so it hardly matters if the correct prediction for those items is 1, 2, or 3 stars. Intuitively, it is more appropriate in these domains to optimize a ranking criterion that focuses on having the top items correct.

In recent years, a number of learning-to-rank approaches have been proposed in the literature to address this issue, which aim to optimize (proxies of) rank-based measures. When applying such IR measures in the recom-

mendation domain, the problem remains that the “ground truth” (that is, whether or not an item is actually relevant for a user) is available only for a tiny fraction of the items. The results of an empirical evaluation can depend on how the items with unknown ground truth are treated when determining the accuracy metrics. In addition, the problem of items not missing at random also exists for learning-to-rank approaches and at least some of them exhibit a strong bias to recommend blockbusters to everyone, which might be of little value for the users.¹⁹

In some domains, like music recommendation, it is also important to avoid very “bad” recommendations as they can greatly impact the user's quality perception.^{6,21} Omitting some “good” recommendations is not nearly so harmful, which would argue for risk-averse algorithm designs that mostly recommend items with a high average rating and low rating variance. Recommending only such generally liked, non-controversial items might however not be particularly helpful for some of the users.

System quality factors beyond accuracy. The Netflix Prize with its focus on accuracy has undoubtedly pushed recommender systems research forward. However, it has also partially overshadowed many other important challenges when building a recommender system and today even Netflix states “there are much better ways to help people find videos to watch than focusing only on those with a high predicted star rating.”¹⁵ Next, we give examples of quality factors other than single-item accuracy, review how recent research has approached these problems, and sketch open challenges.

Novelty, diversity, and other components of utility. Making good rating predictions for as-yet unrated items is almost never the ultimate goal. The true goal of providing recommendations is rather some combination of a certain value for the user and profit for the site. In some domains, user ratings may represent a general quality assessment but still not imply the item should be recommended. As an example, consider the problem of recommending restaurants to travelers. Most people dining at a Michelin-starred location may give it five stars, but budget travelers may be

annoyed to see it recommended. As a result, some researchers have tried to recommend based on a more encompassing model of utility. For the budget traveler, that utility or “economic value” might increase with predicted rating but decrease with cost.¹³ Therefore, predicting how much a user will “like” an item—as done in the Netflix Prize—can in many domains be insufficient. The problem in reality often is to additionally predict the presumed utility of a recommendation for the user.

The novelty and non-obviousness of an item are, for instance, factors that may affect the utility of item recommendations. Proposing the purchase of bread and butter in a grocery shop is obvious and will probably not generate additional sales. Similarly, recommending sequels of a movie that a user liked a lot and will watch anyway or songs by a user’s favorite artist will not help the user discover new things.

In many domains, it is not even meaningful to assess the utility of a single recommendation, but only sets of recommendations. In the movie domain, once a recommendation list includes one Harry Potter movie, there is diminished value from additional Harry Potter movies. Quality measures like novelty, diversity and unexpectedness have therefore moved into the focus of researchers in recent years.⁵

While quite some progress was made over the past few years and researchers are increasingly aware of the problem that being accurate might be insufficient, a number of issues remain open. It is, for example, often not clear if and to what extent a certain quality characteristic like novelty is truly desired in a given application for a specific user. Similarly, too much diversity can sometimes be detrimental to the user experience. Finding the right mix of novel and familiar items can be challenging, and more research is required to better understand the requirements and success factors in particular domains.

Another issue is that estimating the strength of quality factors like diversity based on offline experiments is problematic. Measures like Intra-List-Diversity have been proposed in the literature but up to now it is unclear to what extent such objective measures correlate with the users’ diversity perception.

Generally, the literature focuses on

Predicting held-out matrix entries is really predicting the past rather than the future.

the usefulness of the recommendations from the perspective of the end user. The providers of recommendation services, however, often try to optimize their algorithms based on A/B tests using very different measures, including sales volumes, conversion rates, activity on the platform, or sustained customer loyalty in terms of revisiting customers or renewed subscriptions. These measures vary significantly across businesses and sometimes even over time when the importance of different key performance indicators varies over time.

Context matters. Even if we generally know how to assess the usefulness of an item for a user, this usefulness might not be stable and depend on the user’s current context. Assume, for example, that we have built a recommendation system for restaurants and have done everything right so far. Our algorithm is good at matching the users’ preferences and the recommendations themselves are a good mix of familiar and new options as well as popular choices and insider tips. The recommendations can still be perceived as poor. A restaurant in a northern climate with acceptable food and indoor ambience but an exceptional outdoor patio overlooking the harbor will probably be a good recommendation, but only when visited in summer. Traditional CF techniques unfortunately do not account for such time aspects. In many domains context-aware algorithms are therefore required as they are able to vary their recommendations depending on contextual factors such as time, location, mood, or the presence of other people.

Over the past decade, a number of context-aware recommendation capabilities have been developed in academia and applied in a variety of application settings, including movies, restaurants, music, travel, and mobile applications.² Typical context adaptation strategies are to filter the recommendable items before or after the application of a non-contextualized algorithm, to collect multiple ratings for the same item in different context situations, or to design recommendation techniques that factor in context information into their machine learning models.

Contextualization has also become a common feature in real applications today. For example, many music web-

sites, such as Spotify, ask listeners for their current mood or adapt the recommendations depending on the time of the day. Online shopping sites look at the very recent navigation behavior and infer short-term shopping goals of their visitors. Mobile recommender systems finally constitute a special case of context-aware recommenders, as more and more sensor information becomes available, for example, about the user's location and local time.

From a research perspective, context is a multifaceted concept that has been studied in various research disciplines. Over the last 10 years significant progress has been made also in the field of context-aware recommenders and the first comparative evaluations and benchmark datasets were published.³¹ Nonetheless, much more work is required to fully understand this multifaceted concept and to go beyond what is called the representational approach with its predefined and fixed set of observable attributes.

Interacting with users. Coming back to our restaurant recommender, let us assume we have extended its capabilities and it now considers the user's time and geographical location when making recommendations. But what happens if the user—in contrast to her past preferences—is in the mood to try out something different, for example, a vegan restaurant. How would she tell the system? And if she did not like it afterward, how would she inform the system not to recommend vegan restaurants again in the future?

In many application domains, short-term preferences must be elicited and recommending cannot be a one-shot process of determining and presenting a ranked list of items. Instead, various forms of user interactions might be required or useful to put the user in control. Examples of typical interaction patterns are interactive preference elicitation and refinement procedures, the presentation of explanations and persuasive arguments, or the provision of mechanisms that help users explore the space of available options. The design of the user experience and the provided means of interacting with the system can be a key quality factor for the success of the recommendation service.

In the research literature, conver-

sational recommender systems were proposed to elicit user preferences interactively and engage in a “propose, feedback, and revise” cycle with users.³⁷ They are employed in domains where consumers are confronted with high involvement buying decisions, such as financial services, real estate, or tourism. Most approaches use forms-based dialogues to let users choose from predefined options or use natural language processing techniques to cope with free-text or oral user input. Recent alternative approaches also include more emotional ways of expressing preferences, for example, based on additional sensors to determine the user's emotional state or by supporting alternative ways of user input such as selecting from pictures.³⁰ Furthermore, the integration of better recommendation functionality in voice-controlled virtual assistants like Apple's Siri represents another promising path to explore by the RS research community.

One key insight in conversational systems is that users may not initially understand the space of available items, and so do not have well-formed preferences that can be expressed in terms of attributes of items. An interactive or visualization-based recommender can help users explore the item space and incrementally reveal their preferences to the system. For example, critiquing-based interfaces invite users to say “Show me more like restaurant A, but cheaper.” Although these approaches attracted considerable interest in research, they are not yet mainstream in practice.²⁷

Overall, with interactive systems, the design challenge is no longer simply one of choosing items to recommend but also to choose a sequence of conversational moves as proposed by Mahmood et al.²⁴ who developed an adaptive conversational recommendation system for the tourism domain.

Manipulation resistance. Moving on from the specific problems of how the preferences are acquired and which algorithms are used in our restaurant recommender, the question could arise of whether we can trust that the ratings of the community are honest and fair. Interested parties might manipulate the output of a recommender to their advantage, for example, by cre-

ating fake profiles and ratings.

In the long run, customers who were misled by such manipulated reviews would distrust the recommendations made by the system and in the worst case the online service as a whole. Being resilient against such manipulations can therefore be crucial to the long-term success of a system.

There has been considerable research on manipulation resistance, where resistance is defined as attackers having only a limited ability to change the rating predictions that are made. Most of it identifies archetypal attack strategies and proposes ways to detect and counteract them. For example, one “shilling” or “profile injection” attack creates profiles for fake users, with ratings for many items close to the overall average for all users. Then, these fake users give top (or bottom) ratings to the items that are being manipulated.²² This line of research has identified algorithms that are more or less resistant to particular attack strategies.²⁹

In recent research, the textual reviews provided by users on platforms like TripAdvisor are used instead or in combination with numerical ratings to understand long-term user preferences. These textual reviews do not only carry more detailed information than the ratings, they can also be automatically analyzed to detect fake entries.²⁰ Research suggests that in some domains the fraction of manipulated entries can be significant.

Generally, to resist manipulation, algorithms take some countermeasure that discards or reduces the influence given to ratings or reviews that are suspected of not being trustworthy. However, this has the effect of throwing away some good information. There is a lower bound on the good information that must be discarded in any attempt to prevent attacks by statistical means of noticing anomalous patterns.³⁴ No easy solution to this problem seems to exist, unless attackers can be prevented from injecting fake profiles.

Trust and loyalty. Manipulation resistance is not the only requirement for building a trustworthy system. Let us return to our restaurant recommender and assume that our user has eventually decided to try out one of our recommendations. Thus, from a provider perspective, we were successful in driv-

ing the user's short-term behavior. But what if the user is dissatisfied afterward with her choice and in particular feels that our recommendations were biased and not objective?

As a result, she might not trust the service in the future and even the most relevant recommendations might be ignored. In the worst case, she will even distrust the competence and integrity of the service provider.⁶ An important quality factor of a recommendation system is that it is capable of building long-term loyalty through repeated positive experiences.

In e-commerce settings, users can rightly assume that economic considerations might influence what is placed in the recommendation lists and can be worried that what is being proposed is not truly optimal for them but for the seller. Transparency is therefore an important factor that has been shown to positively influence the user's trust in a system: What data does the RS consider? How does the data lead to recommendations? Explanations put the focus on providing additional information in order to answer these questions and justify the proposed recommendations.

In the research literature, a number of explanation strategies have been explored over the past 10 years. Many of them are based on “white-box” strategies that expose how a system derived the recommendations.¹¹ However, many challenges remain open. One is how to explain recommendations that are cre-

ated by complex ML models. Another is how to leverage additional information such as the browsing history or the user's social graph to make recommendations look more plausible or familiar to the user.³²

From Algorithms to Systems

Our brief survey on the history of the field indicates that recommender systems have arrived at the Main Street with broad industry interest and an active research community. Furthermore, we have seen the recommender systems community address a variety of topics beyond rating prediction and item ranking, for example, concerning the system's user interface or long-term effects.

Beyond the computer science perspective. Many of the proposals discussed earlier focus on algorithmic aspects, for example, how to combine context information with matrix completion approaches, how to find the most “informative” items that users should be asked to rate, or how to design algorithms that balance diversity and accuracy in an optimal way. As suspected in Wagstaff³⁸ for the ML community, the RS research community, to some extent, still seems too focused on benchmark datasets and abstract performance metrics. Whether or not the reported improvements actually matter in the real world for a certain application domain, and the needs of the users—are they, for instance, actu-

ally looking for new items to discover or are they seeking “more of the same” for comparison purposes—this question is seldom asked.

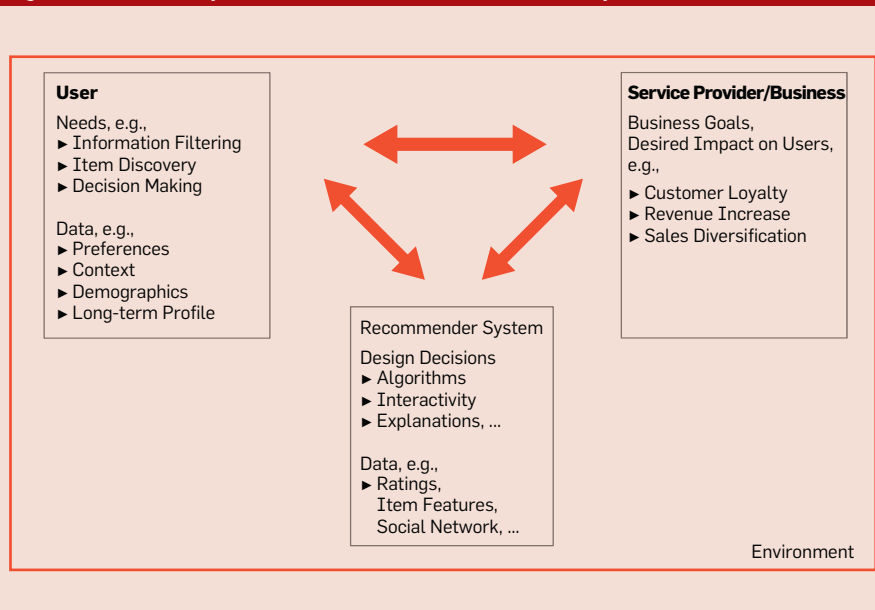
Due to their high practical relevance, RS are naturally a field of research in disciplines other than computer science (CS), including information systems (IS), e-commerce, consumer research, or marketing. Research work like Xiao and Benbasat³⁹ that develop a comprehensive conceptual model of the characteristics, use, and impact of e-commerce “recommendation agents” are largely unnoticed in the CS literature. In their work, the authors develop 28 propositions that center around two practically relevant questions in e-commerce settings: How can RS help to improve the user's decision process and quality? and Which factors influence the user's adoption of and trust toward the system? The process of actually generating the recommendations—which is the focus in the CS field—is certainly important, but only one of several factors that contribute to the success of an RS.

Research questions in the context of a RS should therefore be viewed from a more comprehensive perspective as sketched in Figure 1. Whenever new technological proposals are made, we should ask which specific need or requirement in a given domain are addressed. Making better buying decisions can be one need from the user's perspective; guiding customers to other parts of the product spectrum can be a desired effect from the provider's side. Correspondingly, these goals determine the choice of the evaluation measure that is chosen to assess the effectiveness of the approach.

At the end, the goals of a recommendation system can be very diverse, ranging from improved decision making over item filtering and discovery, to increased conversion or user engagement on the platform. Abstract, domain-independent accuracy measures as often used today are typically insufficient to assess the true value of a new technique.¹⁵

Focusing on business- and utility-oriented measures and the consideration of novelty, diversity, and serendipity aspects of recommendations—as discussed earlier—are important steps into that direction. In

Figure 1. A more comprehensive view on the recommendation problem.



any case, which measures are actually chosen for the evaluation, always has to be justified by the specific goals that should be achieved with the system. Furthermore, in offline experimentations, multi-metric evaluation schemes, application-specific measures, and the consideration of recommendation biases represent one way of assessing desired and potentially undesired effects of a RS on its users.¹⁹

However, to better understand the effectiveness of a RS and its impact on users, more user-centric and utility-oriented research is generally required within the CS community and the algorithmic works should be better connected with the already existing insights from neighboring fields.

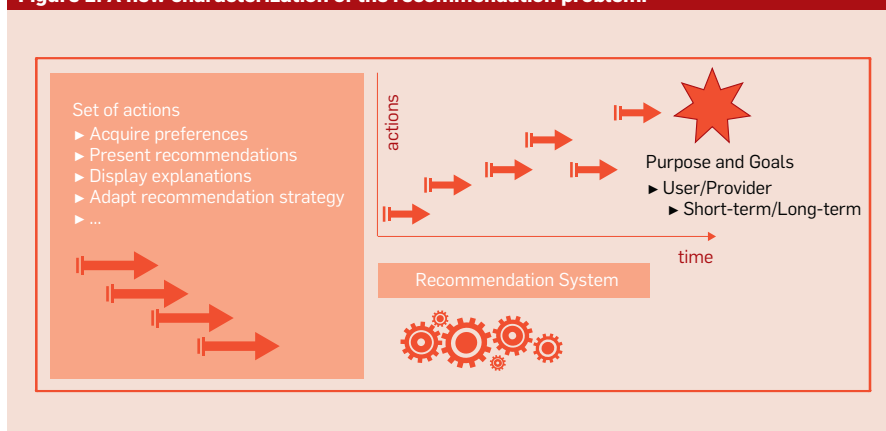
Putting the user back in the loop.

A recommender system is usually one component within an interactive application. The minimal interaction level provided by such a component is that a list of recommendations is displayed and users can select them for inspection or immediate consumption, for example, on media streaming platforms.

RS have one of their roots in the field of human-computer interaction (HCI) and the design of the user interface, the choice of the supported forms of interactivity, or the selection of the content to be displayed can all have an impact on the success of a recommender. However, the amount of research dedicated to these questions is comparably low, particularly when compared to the huge amount of research on item-ranking algorithms.

Therefore, our second tenet is that the CS community should put more effort on the HCI perspective of RS, as has been advocated earlier, for example, in Konstan and Riedl²¹ and McNee et al.²⁸ Current research largely relies on explicit ratings and automatically observable user actions—often called *implicit feedback*—as preference indicators. Many real-world systems allow users to explicitly specify their preferences, for example, in terms of preferred item categories. Recommendation components on websites could be much more interactive and act, for example, in the e-commerce domain as “virtual advisers”³⁹ and social actors that ask questions, adapt their communication to the current user, provide

Figure 2. A new characterization of the recommendation problem.



explanations when desired, present alternative or complementary shopping proposals and, in general, put the user more into control and allow for new types of interactions.

When looking at the recommendations provided by Amazon.com on their websites, we can see that various forms of user interaction already exist that are underexplored in academia. Amazon.com, for example, provides multiple recommendation lists on its landing page. Amazon's system also supports explanations for the made recommendations and even lets the user indicate if a past user action observed by the system (for example, a purchase) should no longer be considered in the recommendation process. However, many questions, such as how to design such interactive elements in the best possible way, how much cognitive load for the user is acceptable, or how the system can stimulate or persuade people to do certain actions are largely unexplored.

Furthermore, in case a system supports various forms of interactivity and is at the same time capable of acquiring additional information from the user, additional algorithmic and computational challenges arise. An intelligent system might, for example, decide on the next conversational move, or whether to display an explanation or not, depending on the current state of the interaction or the estimated expertise and competence of the user. Some approaches in that direction were proposed in the literature in the past, but they often come at expense of considerable ramp-up costs in terms of knowledge engineering and they might appear to be quite static if they have no

built-in learning capabilities.^{10,24}

Finally, mobile and wearable devices have become the personal digital assistants of today. With the recent developments in speech recognition, gesture-based interactions, and a multitude of additional sensors of these devices, new opportunities arise regarding how we interact with recommender systems.

Toward a more comprehensive characterization of the recommendation task. In the research literature, an often-cited definition of the recommendation problem is to find a function that outputs a relevance score for each item given information about the user profile and her contextual situation, “content” information about the items, and information about preference patterns in the user community.¹ Although the development of even better techniques for item selection and ranking will remain at the core of the research problem, the discussions here indicate this definition seems too narrow. To conclude our considerations related to the HCI perspective on recommenders and the more comprehensive consideration of the interplay between users, organizations, and the recommendation system, we propose a new characterization of the recommendation problem (Figure 2).

A new problem characterization. A recommendation problem has the following three components: an overall goal that governs the selection and ranking of items; a set of available actions centered on the presentation of recommended items; and an optimization timeframe:

- The overall goal constitutes the operationalized measure or a set of

measures that should be optimized by an appropriate selection and ranking of items from a (large) set. Optimizing a specific rank measure can be such a goal, but more utility-oriented goals and corresponding measures like user satisfaction, decreased decision efforts, revenues, or loyalty might be equally important. Generally, the goals can be derived from the user's perspective, the provider's perspective, or both.

► Depending on the application domain, a set of actions is available for the recommendation system to take. The central action typically is the selection and presentation of a set of items. Additional possible moves are varying its strategy to recommend items, providing specific explanations or other communication content, requesting feedback or alternative variants of user input. These conversational moves are building blocks for goal achievement. The selection of the most helpful next action and its timing can be the result of a reasoning process itself.

► The timeframe or optimization horizon signifies the time window over which the goal should be optimized. The explicit consideration of the time dimension allows us to differentiate between single one-shot interactions and longer time spans that can be more relevant to businesses and users.

The recommendation problem finally can be defined as: Find a sequence of conversational actions and item recommendations for each particular user that optimizes the overall goal over the specified timeframe.

Summary

Recommender systems have become a natural part of the user experience in today's online world. These systems are able to deliver value both for users and providers and are one prominent example where the output of academic research has a direct impact on the advancements in industry.

In this article, we have briefly reviewed the history of this multidisciplinary field and looked at recent efforts in the research community to consider the variety of factors that may influence the long-term success of a recommender system. The list of open issues and success factors is still far from complete and new challenges arise constantly that require further re-

search. For example, the huge amounts of user data and preference signals that become available through the Social Web and the Internet of Things not only leads to technical challenges such as scalability, but also to societal questions concerning user privacy.

Based on our reflections on the developments in the field, we finally emphasize the need for a more holistic research approach that combines the insights of different disciplines. We urge that research focuses even more on practical problems that matter and are truly suited to increase the utility of recommendations from the viewpoint of the users. **C**

References

- Adomavicius, G. and Tuzhilin, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowledge and Data Engineering* 17, 6 (2005), 734–749.
- Adomavicius, G. and Tuzhilin, A. Context-aware recommender systems. *Recommender Systems Handbook*. Springer, 2011, 217–253.
- Billsus, D. and Pazzani, M.J. Learning collaborative information filters. In *Proceedings ICML '98* (1998), 46–54.
- Breese, J.S., Heckerman, D. and Kadie, C.M. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings UAI '98* (1998), 43–52.
- Castells, P., Wang, J., Lara, R. and Zhang, D. Introduction to the special issue on diversity and discovery in recommender systems. *ACM Trans. Intell. Syst. Technology* 5, 4 (2014), 52:1–52:3.
- Chau, P.Y.K., Ho, S.Y., Ho, K.K.W. and Yao, Y. Examining the effects of malfunctioning personalized services on online users' distrust and behaviors. *Decision Support Syst.* 56 (2013), 180–191.
- Cremonesi, P., Garzotto, F. and Turrin, R. Investigating the persuasion potential of recommender systems from a quality perspective: An empirical study. *ACM Trans. Interact. Intell. Syst.* 2, 1 (2012), 11:1–11:41.
- Denning, P.J. ACM president's letter: Electronic junk. *Commun. ACM* 25, 3 (Mar. 1982), 163–165.
- Dias, M.B., Locher, D., Li, M., El-Deredy, W. and Lisboa, P.J. The value of personalised recommender systems to e-business: A case study. In *Proceedings RecSys'08* (2008), 291–294.
- Felfernig, A., Friedrich, G., Jannach, D. and Zanker, M. An integrated environment for the development of knowledge-based recommender applications. *Int. J. Electron. Commerce* 11, 2 (2006), 11–34.
- Friedrich, G. and Zanker, M. A taxonomy for generating explanations in recommender systems. *AI Magazine* 32, 3 (2011), 90–98.
- Garcin, F., Faltings, B., Donatsch, O., Alazzawi, A., Bruttin, C. and Huber, A. Offline and online evaluation of news recommender systems at swissinfo.ch. In *Proceedings RecSys '14* (2014), 169–176.
- Ghose, A., Ipeirotis, P.G. and Li, B. Designing ranking systems for hotels on travel search engines by mining user-generated and crowdsourced content. *Marketing Science* 31, 3 (2012), 493–520.
- Goldberg, D., Nichols, D., Oki, B. and Terry, D. Using collaborative filtering to weave an information tapestry. *Commun. ACM* (1992), 61–70.
- Gomez-Urbe, C.A. and Hunt, N. The Netflix Recommender System: Algorithms, business value, and innovation. *ACM Trans. Manage. Inf. Syst.* 6, 4 (2015), 13:1–13:19.
- Gorgoglione, M., Panniello, U. and Tuzhilin, A. The effect of context-aware recommendations on customer purchasing behavior and trust. In *Proceedings RecSys '11* (2011), 85–92.
- Hensley, C.B. Selective dissemination of information (SDI): State of the art in May, 1963. In *Proceedings of AFIPS '63* (Spring), 1963, 257–262.
- Hill, W., Stead, L., Rosenstein, M. and Furnas, G. Recommending and evaluating choices in a virtual community of use. In *Proceedings CHI '95* (1995), 194–201.
- Jannach, D., Lerche, L., Kamehkhosh, I. and Jugovac, M. What recommenders recommend: An analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction* (2015), 25:1–65.
- Jindal, N. and Liu, B. Opinion spam and analysis. In *Proceedings WSDM '08*, (2008), 219–230.
- Konstan, J. and Riedl, J. Recommender systems: From algorithms to user experience. *User Modeling and User-Adapted Interaction* 22, 1–2 (2012), 101–123.
- Lam, S.K. and Riedl, J. Shilling recommender systems for fun and profit. In *Proceedings of WWW '04*, (2004), 393–402.
- Linden, G., Smith, B. and York, J. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* 7, 1 (2003), 76–80.
- Mahmood, T., Ricci, F. and Venturini, A. Improving recommendation effectiveness: Adapting a dialogue strategy in online travel planning. *J. of IT & Tourism* 11, 4 (2009), 285–302.
- Malone, T.W., Grant, K.R., Turbak, F.A., Brobst, S.A. and Cohen, M.D. Intelligent information-sharing systems. *Commun. ACM* 30, 5 (May 1987), 390–402.
- Marlin, B.M. and Zemel, R.S. Collaborative prediction and ranking with non-random missing data. In *Proceedings RecSys '09* (2009), 5–12.
- McGinty, L. and Reilly, J. On the evolution of critiquing recommenders. *Recommender Systems Handbook*, Springer, 2011, 419–453.
- McNee, S.M., Riedl, J. and Konstan, J.A. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *Proceedings CHI '06*, (2006), 1097–1101.
- Mobasher, B., Burke, R., Bhaumik, R. and Williams, C. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Technology* 7, 4 (Oct. 2007).
- Neidhardt, J., Seyfang, L., Schuster, R. and Werthner, H. A picture-based approach to recommender systems. *J. of IT & Tourism* 15 (2015), 1–21.
- Panniello, U., Tuzhilin, A. and Gorgoglione, M. Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction* 24, 1–2 (2014), 35–65.
- Papadimitriou, A., Symeonidis, P. and Manolopoulos, Y. A generalized taxonomy of explanations styles for traditional and social recommender systems. *Data Min. Knowl. Discovery* 24, 3 (2012), 555–583.
- Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of CSCW '94* (1994), 175–186.
- Resnick, P. and Sami, R. The information cost of manipulation-resistance in recommender systems. In *Proceedings RecSys '08* (2008), 147–154.
- Schafer, J.B., Konstan, J. and Riedl, J. Recommender systems in e-commerce. In *Proceedings ACM EC '99* (1999), 158–166.
- Shardanand, U. and Maes, P. Social information filtering: Algorithms for automating "word of mouth." In *Proceedings CHI '95* (1995), 210–217.
- Shimazu, H. Expertclerk: Navigating shoppers' buying process with the combination of asking and proposing. In *Proceedings IJCAI '01* (2001), 1443–1448.
- Wagstaff, K. Machine learning that matters. In *Proceedings ICML* (2012), 529–536.
- Xiao, B. and Benbasat, I. E-commerce product recommendation agents: Use, characteristics, and impact. *MIS Q.* 31, 1 (Mar. 2007), 137–209.

Dietmar Jannach (dietmar.jannach@udo.edu) is a Chaired Professor of Computer Science at TU Dortmund, Germany.

Paul Resnick (presnick@umich.edu) is the Michael D. Cohen Collegiate Professor of Information at the University of Michigan School of Information, Ann Arbor, MI.

Alexander Tuzhilin (atuzhili@stern.nyu.edu) is the Leonard N. Stern Professor of Business in the Stern School of Business, New York University, NY.

Markus Zanker (markus.zanker@unibz.it) is an associate professor of computer science at Free University of Bozen-Bolzano, Italy.

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

research highlights

P. 104

Technical Perspective If I Could Only Design One Circuit ...

By Kurt Keutzer

P. 105

DianNao Family: Energy-Efficient Hardware Accelerators for Machine Learning

By Yunji Chen, Tianshi Chen, Zhiwei Xu,
Ninghui Sun, and Olivier Temam

P. 113

Technical Perspective FPGA Compute Acceleration Is First About Energy Efficiency

By James C. Hoe

P. 114

A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services

By Andrew Putnam, Adrian M. Caulfield, Eric S. Chung, Derek Chiou,
Kypros Constantinides, John Demme, Hadi Esmaeilzadeh,
Jeremy Fowers, Gopi Prashanth Gopal, Jan Gray, Michael Haselman,
Scott Hauck, Stephen Heil, Amir Hormati, Joo-Young Kim,
Sitaram Lanka, James Larus, Eric Peterson, Simon Pope,
Aaron Smith, Jason Thong, Phillip Yi Xiao, and Doug Burger

Technical Perspective

If I Could Only Design One Circuit ...

By Kurt Keutzer

NEURAL-INSPIRED COMPUTING MODELS have captured our imagination from the very beginning of computer science; however, victories of this approach were modest until 2012 when AlexNet, a “deep” neural net of eight layers, achieved a dramatic improvement on the image classification problem. One key to AlexNet’s success was its use of the increased computational power offered by graphics processing units (GPUs), and it’s natural to ask: Just how far can we push the efficient computing of neural nets?

Computing capability has advanced with Moore’s Law over these last three decades, but integrated circuit design costs have grown nearly as fast. Thus, any discussion of novel circuit architectures must be met with a sobering discussion of design costs. That said, a neural net accelerator has two big things going for it. First, it is a special-purpose accelerator. Since the end of single-thread performance scaling due to power density issues, integrated circuit architects have searched for clever ways to exploit the increasing transistor counts afforded by Moore’s Law without increasing power dissipation. This has led to a resurgence of special-purpose accelerators that are able to provide 10–100x better energy efficiency than general-purpose processors when accelerating their special functions, and which consume practically no power when not in use.

Second, a neural net accelerator can accelerate a broad range of applications. Deep neural nets have begun to realize the promise that has intrigued so many for so long: a single, neuron-inspired computational model that offers superior results on a diverse variety of problems. In particular, modern deep neural net models are winning competitions in computer vision, speech recognition, and text analytics. Without exaggeration, the list of victories achieved through the use of deep neural nets grows every week.

Like many other machine learning approaches, neural net development has two phases. The *training* phase is essentially an optimization problem in which parameter weights of neural net models are adjusted to minimize the error of the neural net on its training set. This is followed by the *implementation* or *inference* phase, in which the resulting neural net is deployed in its target application, such as a speech recognizer in a cellphone.

Training neural nets is a highly distributed optimization problem in which interprocessor-communication costs quickly dominate local computational costs. On the other hand, the *implementation* of neural nets in embedded applications, such as cellphones, calls out for a special-purpose, energy-efficient accelerator. Thus, if I could cajole my circuit designer colleagues into designing only one circuit, it would surely be a special-purpose, energy-efficient accelerator that is flexible enough to provide efficient *implementations* of the growing family of neural net models. This is the goal of DianNao (diàn nǎo, Chinese for computer, or, literally “electric brain”).

The DianNao accelerator family comprehensively considers the problem of designing a neural net accelerator, and the following paper shows a deep understanding of both neural net implementations and the issues in computer architecture that arise when building an accelerator for them. Neural net models are evolving rapidly, and a significant new neural network model is proposed every month, if not every week. Thus, a computer architect building an accelerator for neural nets must be familiar with their variety. A specializer-architecture that isn’t sufficiently flexible to accommodate a broad range of neural net models is certain to become quickly outdated, wasting the extensive chip design effort.

The DianNao family also engages the issues associated with building

a processor architecture for a neural net accelerator and puts a particularly strong focus on efficiently supporting the memory access patterns of neural net computations. This includes minimizing both on-chip and off-chip memory transfers. Other members of the DianNao family include DaDianNao, ShiDianNao, and PuDianNao. DaDianNao (big computer) focuses on the challenges of efficiently computing neural nets with one billion or more model parameters. ShiDianNao (vision computer) is further specialized to reduce memory access requirements of Convolutional Neural Nets, a neural net family that is used for computer vision problems. While the number of problems solved by neural nets grows every week, some might wonder: Is this a fundamental change in the field, or will the pendulum swing back to favor a broader range of machine learning approaches? With the PuDianNao (general computer) architecture, the architects hedge their bets on this question by providing an accelerator for more traditional machine learning algorithms.

Despite, or perhaps because of, DianNao’s two Best Paper Awards, some readers may think that building a neural network accelerator is just an academic enterprise. These doubts should be allayed by Google’s announcement of the Tensor Processing Unit, a novel neural network accelerator deployed in their datacenters. These processors were recently used to help AlphaGo win at Go. It may be quite some time before we learn of TPU’s architecture, but details on the DianNao family are only a page away. 

Kurt Keutzer (keutzer@berkeley.edu) is a professor of electrical engineering and computer science at the University of California at Berkeley.

Copyright held by author.

DianNao Family: Energy-Efficient Hardware Accelerators for Machine Learning

By Yunji Chen, Tianshi Chen, Zhiwei Xu, Ninghui Sun, and Olivier Temam

Abstract

Machine Learning (ML) tasks are becoming pervasive in a broad range of applications, and in a broad range of systems (from embedded systems to data centers). As computer architectures evolve toward heterogeneous multi-cores composed of a mix of cores and hardware accelerators, designing hardware accelerators for ML techniques can simultaneously achieve high efficiency and broad application scope.

While efficient computational primitives are important for a hardware accelerator, inefficient memory transfers can potentially void the throughput, energy, or cost advantages of accelerators, that is, an Amdahl's law effect, and thus, they should become a first-order concern, just like in processors, rather than an element factored in accelerator design on a second step. In this article, we introduce a series of hardware accelerators (i.e., the DianNao family) designed for ML (especially neural networks), with a special emphasis on the impact of memory on accelerator design, performance, and energy. We show that, on a number of representative neural network layers, it is possible to achieve a speedup of 450.65x over a GPU, and reduce the energy by 150.31x on average for a 64-chip DaDianNao system (a member of the DianNao family).

1. INTRODUCTION

As architectures evolve towards heterogeneous multi-cores composed of a mix of cores and accelerators, designing hardware accelerators which realize the best possible tradeoff between flexibility and efficiency is becoming a prominent issue. The first question is for which category of applications one should primarily design accelerators? Together with the architecture trend towards accelerators, a second simultaneous and significant trend in high-performance and embedded applications is developing: many of the emerging high-performance and embedded applications, from image/video/audio recognition to automatic translation, business analytics, and robotics rely on *machine learning techniques*. This trend in application comes together with a third trend in machine learning (ML) where a small number of techniques, based on neural networks (especially *deep learning techniques*^{16, 26}), have been proved in the past few years to be state-of-the-art across a broad range of applications. As a result, there is a unique opportunity to design accelerators having significant application scope as well as high performance and efficiency.⁴

Currently, ML workloads are mostly executed on multi-cores using SIMD,⁴⁴ on GPUs,⁷ or on FPGAs.² However, the

aforementioned trends have already been identified by researchers who have proposed accelerators implementing, for example, Convolutional Neural Networks (CNNs)² or Multi-Layer Perceptrons⁴³; accelerators focusing on other domains, such as image processing, also propose efficient implementations of some of the computational primitives used by machine-learning techniques, such as convolutions.³⁷ There are also ASIC implementations of ML techniques, such as Support Vector Machine and CNNs. However, all these works have first, and successfully, focused on efficiently implementing the computational primitives but they either voluntarily ignore memory transfers for the sake of simplicity,^{37, 43} or they directly plug their computational accelerator to memory via a more or less sophisticated DMA.^{2, 12, 19}

While efficient implementation of computational primitives is a first and important step with promising results, inefficient memory transfers can potentially void the throughput, energy, or cost advantages of accelerators, that is, an Amdahl's law effect, and thus, they should become a first-order concern, just like in processors, rather than an element factored in accelerator design on a second step. Unlike in processors though, one can factor in the specific nature of memory transfers in target algorithms, just like it is done for accelerating computations. This is especially important in the domain of ML where there is a clear trend towards scaling up the size of learning models in order to achieve better accuracy and more functionality.^{16, 24}

In this article, we introduce a series of hardware accelerators designed for ML (especially neural networks), including DianNao, DaDianNao, ShiDianNao, and PuDianNao as listed in Table 1. We focus our study on memory usage, and we investigate the accelerator architecture to minimize memory transfers and to perform them as efficiently as possible.

2. DIANNAO: A NEURAL NETWORK ACCELERATOR

Neural network techniques have been proved in the past few years to be state-of-the-art across a broad range of applications. DianNao is the first member of the DianNao accelerator family, which accommodates state-of-the-art neural

The original version of this paper is entitled "DianNao: A Small-Footprint, High-Throughput Accelerator for Ubiquitous Machine Learning" and was published in *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)* 49, 4 (March 2014), ACM, New York, NY, 269–284.

Table 1. Accelerators in the DianNao family.

Name	Process (nm)	Peak performance (GOP/s)	Peak power (W)	Area (mm ²)	Applications
DianNao	65	452	0.485	3.02	Neural networks
DaDianNao	28	5585	15.97	67.73	Neural networks
ShiDianNao	65	194	0.32	4.86	Convolutional neural networks
PuDianNao	65	1056	0.596	3.51	Seven representative machine learning techniques

network techniques (e.g., deep learning^a), and inherits the broad application scope of neural networks.

2.1. Architecture

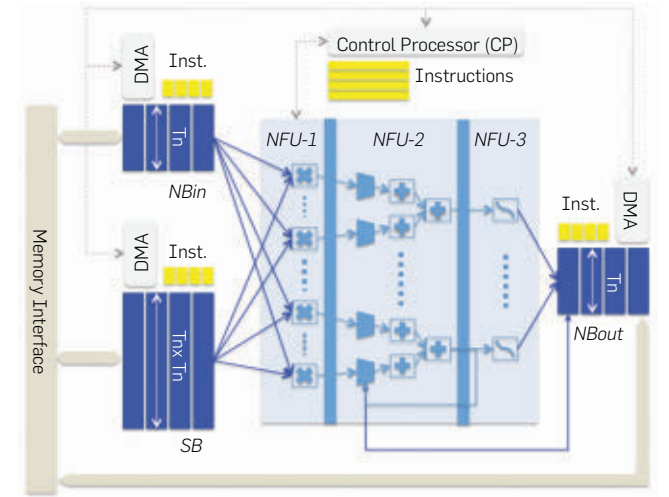
DianNao has the following components: an input buffer for input neurons (NBin), an output buffer for output neurons (NBout), and a third buffer for synaptic weights (SB), connected to a computational block (performing both synapses and neurons computations) which we call the Neural Functional Unit (NFU), and the control logic (CP), see Figure 1.

Neural Functional Unit (NFU). The NFU implements a functional block of T_i inputs/synapses and T_n output neurons, which can be time-shared by different algorithmic blocks of neurons. Depending on the layer type, computations at the NFU can be decomposed in either two or three stages. For classifier and convolutional layers: multiplication of synapses $\times$ inputs, additions of all multiplications, sigmoid. The nature of the last stage (sigmoid or another nonlinear function) can vary. For pooling layers, there is no multiplication (no synapse), and the pooling operations can be average or max. Note that the adders have multiple inputs, they are in fact *adder trees*, see Figure 1; the second stage also contains shifters and max operators for pooling layers. In the NFU, the sigmoid function (for classifier and convolutional layers) can be efficiently implemented using piecewise linear interpolation ($f(x) = a_i \times x + b_i, x \in [x_i; x_{i+1}]$) with negligible loss of accuracy (16 segments are sufficient).²²

On-chip Storage. The on-chip storage structures of DianNao can be construed as modified buffers of scratchpads. While a cache is an excellent storage structure for a general-purpose processor, it is a sub-optimal way to exploit reuse because of the cache access overhead (tag check, associativity, line size, speculative read, etc.) and cache conflicts. The efficient alternative, scratchpad, is used in VLIW processors but it is known to be very difficult to compile for. However a scratchpad in a dedicated accelerator realizes the best of both worlds: efficient storage, and both efficient and easy exploitation of locality because only a few algorithms have to be manually adapted.

We split on-chip storage into three structures (NBin, NBout, and SB), because there are three type of data (input neurons, output neurons and synapses) with different characteristics (e.g., read width and reuse distance). The first benefit of

^a According to a recent review²⁵ written by LeCun, Bengio, and Hinton, *Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics.*

Figure 1. Accelerator architecture of DianNao.

splitting structures is to tailor the SRAMs to the appropriate read/write width, and the second benefit of splitting storage structures is to avoid conflicts, as would occur in a cache. Moreover, we implement three DMAs to exploit spatial locality of data, one for each buffer (two load DMAs for inputs, one store DMA for outputs).

2.2. Loop tiling

DianNao leverages loop tiling to minimize memory accesses, and thus efficiently accommodates large neural networks. For the sake of brevity, here we only discuss a classifier layer^b that has N_n output neurons, fully connected to N_i inputs. We present in Figure 2 the original code of the classifier, as well as the tiled code that maps the classifier layer to DianNao.

In the tiled code, the loops ii and nn reflects the aforementioned fact that the NFU is a functional block of T_i inputs/synapses and T_n output neurons. On the other hand, input neurons are reused for each output neuron, but since the number of input neurons can range anywhere between a few tens to hundreds of thousands, they will often not fit in the NBin size of DianNao. Therefore, we further tile loop ii (input neurons) with tile factor T_{ii} . A typical tradeoff of tiling is that improving one reference (here *neuron[i]* for input neurons) increases the reuse distance of another reference (*sum[n]* for partial sums of output neurons), so we need to tile for the second reference as well, hence loop nnn and the tile

^b Readers may refer to Ref.⁵ for details of other layer types.

Figure 2. Pseudo-code for a classifier layer (top: original code; bottom: tiled code).

```

for (int n = 0; n < Nn; n++)
    sum[n] = 0;
for (int n = 0; n < Nn; n++) // output neurons
    for (int i = 0; i < Ni; i++) // input neurons
        sum[n] += synapse[n][i] * neuron[i];
for (int n = 0; n < Nn; n++)
    neuron[n] = sigmoid(sum[n]);

for (int nnn = 0; nnn < Nn; nnn += Tnn){ // tiling for output neurons
    for (int iii = 0; iii < Ni; iii += Tii){ // tiling for input neurons
        for (int nn = nnn; nn < nnn + Tnn; nn += Tn){
            for (int n = nn; n < nn + Tn; n++){
                sum[n] = 0;
                for (int ii = iii; ii < iii + Tii; ii += Ti)
                    for (int n = nn; n < nn + Tn; n++){
                        for (int i = ii; i < ii + Ti; i++)
                            sum[n] += synapse[n][i] * neuron[i];
                    }
                for (int n = nn; n < nn + Tn; n++)
                    neuron[n] = sigmoid(sum[n]);
            }
        }
    }
}

```

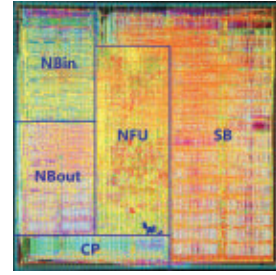
factor T_{nn} for output neurons partial sums. The layer memory behavior is now dominated by synapses. In a classifier layer, all synapses are usually unique, and thus there is no reuse within the layer. Overall, tiling drastically reduces the total memory bandwidth requirement of the classifier layer, and we observe a $\sim 50\%$ reduction in our empirical study.⁵

2.3. Experimental observations

We implemented a custom cycle-accurate, bit-accurate C++ simulator of the accelerator. This simulator is also used to measure time in number of cycles. It is plugged to a main memory model allowing a bandwidth of up to 250 GB/s. We also implemented a Verilog version of the accelerator, which uses $T_n = T_i = 16$ (16 hardware neurons with 16 synapses each), so that the design contains 256 16-bit truncated multipliers (for classifier and convolutional layers), 16 adder trees of 15 adders each (for the same layers, plus pooling layer if average is used), as well as a 16-input shifter and max (for pooling layers), and 16 16-bit truncated multipliers plus 16 adders (for classifier and convolutional layers, and optionally for pooling layers). For classifier and convolutional layers, multipliers and adder trees are active every cycle, achieving $256 + 16 \times 15 = 496$ fixed-point operations every cycle; at 0.98 GHz, this amounts to 452 GOP/s (Giga fixed-point Operations per second). We have done the synthesis and layout of the accelerator at 65 nm using Synopsys tools, see Figure 3.

We implemented an SIMD baseline using the GEM5+McPAT²⁷ combination. We use a 4-issue superscalar x86 core with a 128-bit (8×16 -bit) SIMD unit (SSE/SSE2), clocked at 2 GHz. Following a default setting of GEM5, the core has a 192-entry reorder buffer and a 64-entry load/store queue. The L1 data (and instruction) cache is 32 KB and the L2 cache is 2 MB; both caches are 8-way associative and use a 64-byte line; these cache characteristics correspond to those of the Intel Core i7. In addition, we also employ NVIDIA K20M (28 nm process, 5 GB GDDR5, 3.52 TFlops peak) the GPU baseline.

Figure 3. Snapshot of DianNao's layout.



We employed several representative layer settings as benchmarks of our experiments, see Table 2. We report in Figure 4 the speedups of DianNao over SIMD and GPU. We observe that DianNao significantly outperforms SIMD, and the average speedup is **117.87x**. The main reasons are twofold. First, DianNao performs 496 16-bit operations every cycle for both classifier and convolutional layers, that is, $62\times$ more ($\frac{496}{8}$) than the peak performance of the SIMD baseline. Second, compared with the SIMD baseline without prefetcher, DianNao has better latency tolerance due to an appropriate combination of preloading and reuse in NBin and SB buffers.

In Figure 5, we provide the energy reductions of DianNao over SIMD and GPU. We observe that DianNao consumes 21.08x less energy than SIMD on average. This number is actually more than an order of magnitude smaller than previously reported energy ratios between processors and accelerators; for instance Hameed et al.¹⁵ report an energy ratio of about 500x, and 974x has been reported for a small Multi-Layer Perceptron.⁴³ The smaller ratio is largely due to the energy spent in memory accesses, which was voluntarily not factored in the two aforementioned studies. Like in these two accelerators and others, the energy cost of computations has been considerably reduced by a combination of more efficient computational operators (especially a massive number of small

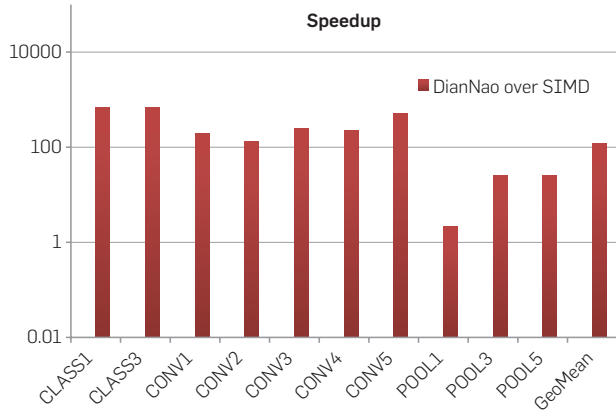
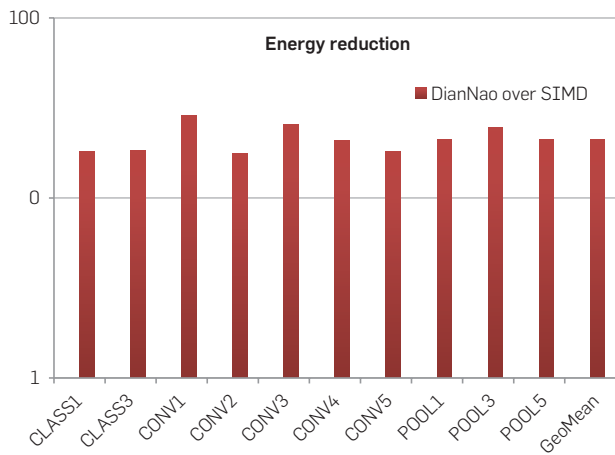
Table 2. Benchmark layers for DianNao.

Layer	N_x	N_y	K_x	K_y	N_i	N_o	Description
CONV1	500	375	9	9	32	48	Street scene parsing (CNN) ¹² (e.g., identifying "building," "vehicle," etc.)
POOL1	492	367	2	2	12	—	
CLASS1	—	—	—	—	960	20	
CONV2*	200	200	18	18	8	8	Detection of faces in YouTube videos (DNN), ²⁴ largest NN to date (Google)
CONV3	32	32	4	4	108	200	
POOL3	32	32	4	4	100	—	Traffic sign identification for car navigation (CNN) ³⁹
CLASS3	—	—	—	—	200	100	
CONV4	32	32	7	7	16	512	Google Street View house numbers (CNN) ³⁸
CONV5*	256	256	11	11	256	384	
POOL5	256	256	2	2	256	—	Multi-Object recognition in natural images (DNN), ¹⁶ winner 2012 ImageNet competition

CONV, convolutional; POOL, pooling; CLASS, classifier.

$N_x \times N_y$ is the size of an input feature map, $K_x \times K_y$ is the size of a convolutional/pooling window, and N_i and N_o are numbers of input and output feature maps, respectively.

*Indicates private kernels.

Figure 4. Speedups of DianNao over SIMD.**Figure 5. Energy reductions of DianNao over SIMD.**

16-bit fixed-point truncated multipliers in our case), and small custom storage located close to the operators (64-entry NBin, NBout, SB, and the NFU-2 registers). As a result, there is now an Amdahl's law effect for energy, where any further improvement can only be achieved by bringing down the

energy cost of main memory accesses. We tried to artificially set the energy cost of the main memory accesses in both the SIMD and accelerator to 0, and we observed that the average energy reduction of the accelerator increases by more than one order of magnitude, in line with previous results.

We have also explored different parameter settings for DianNao in our experimental study, where we altered the size of NFU as well as sizes of NBin, NBout, and SB. For example, we evaluated a design with $T_n = 8$ (i.e., the NFU only has 8 hardware neurons), and thus 64 multipliers in NFU-1. We correspondingly reduced widths of all buffers to fit the NFU. As a result, the total area of that design is 0.85 mm², which is 3.59x smaller than for the case of $T_n = 16$.

3. DADIANNAO: A MACHINE LEARNING SUPERCOMPUTER

In the ML community, there is a significant trend towards increasingly large neural networks. The recent work of Krizhevsky et al.²⁰ achieved promising accuracy on the ImageNet database⁸ with "only" 60 million parameters. There are recent examples of a 1-billion parameter neural network.²⁴ Although DianNao can execute neural networks at different scales, it has to store the values of neurons and synapses in main memory when accommodating large neural networks. Frequent main memory accesses greatly limit the performance and energy-efficiency of DianNao.

While 1 billion parameters or more may come across as a large number from a ML perspective, it is important to realize that, in fact, it is not from a hardware perspective: if each parameter requires 64 bits, that only corresponds to 8GB (and there are clear indications that fewer bits are sufficient). While 8GB is still too large for a single chip, it is possible to imagine a dedicated ML computer composed of multiple chips, each chip containing specialized logic together with enough RAM that the sum of the RAM of all chips can contain the *whole* neural network, requiring *no main memory*.

In large neural networks, the fundamental issue is the memory storage (for reuse) or bandwidth requirements (for fetching) of the synapses of two types of layers: convolutional layers with private kernels, and classifier layers (which

are usually fully connected, and thus have lots of synapses). In DaDianNao, we tackle this issue by adopting the following design principles: (1) we create an architecture where synapses are always stored close to the neurons which will use them, minimizing data movement, saving both time and energy; the architecture is fully distributed, there is no main memory (Swanson *et al.* adopted a similar strategy dataflow computing⁴¹); (2) we create an asymmetric architecture where each node footprint is massively biased towards storage rather than computations; (3) we transfer neurons values rather than synapses values because the former are orders of magnitude fewer than the latter in the aforementioned layers, requiring comparatively little external (across chips) bandwidth; (4) we enable high internal bandwidth by breaking down the local storage into many tiles.

The general architecture of DaDianNao is a set of nodes, one per chip, all identical, arranged in a classic mesh topology. Each node contains significant storage, especially for synapses, and neural computational units (the classic pipeline of multipliers, adder trees and nonlinear functions implemented via linear interpolation), which we still call NFU for the sake of consistency with the DianNao accelerator.⁵ Here we briefly introduce some key characteristics of the node architecture.

Tile-based organization. Putting all functional units (e.g., adders and multipliers) together in a single computational block (NFU) is an acceptable design choice when the NFU has a moderate area, which is the case of DianNao. However, if we significantly scale up the NFU, the data movement between NFU and on-chip storage will require a very high internal bandwidth (even if we split the on-chip storage), resulting in unacceptably large wiring overheads.⁶ To address this issue, we adopt a tile-based organization in each node, see Figure 6. Each tile contains an NFU and four RAM banks to store synapses between neurons.

When accommodating a neural network layer, the output neurons are spread out in the different tiles, so that each NFU can simultaneously process 16 input neurons of 16 output neurons (256 parallel operations). All the tiles are connected through a fat tree which serves to broadcast the input neurons values to each tile, and to collect the output neurons values from each tile. At the center of the chip, there are two special RAM banks, one for input neurons, the other for output neurons. It is important to understand that, even with a large number of tiles and chips, the total number of *hardware* output neurons of all NFUs, can still be small compared

to the actual number of neurons found in large layers. As a result, for each set of input neurons broadcasted to all tiles, multiple different output neurons are being computed on the same hardware neuron. The intermediate values of these neurons are saved back locally in the tile RAM. When the computation of an output neuron is finished (all input neurons have been factored in), the value is sent through the fat tree to the center of the chip to the corresponding (output neurons) central RAM bank.

Storage. In some of the largest known neural networks, the storage size required by a layer typically range from less than 1 MB to about 1 GB, most of them ranging in the tens of MB. While SRAMs are appropriate for caching purposes, they are not dense enough for such large-scale storage. However, eDRAMs are known to have a higher storage density. For instance, a 10 MB SRAM memory requires 20.73 mm² at 28 nm,³⁰ while an eDRAM memory of the same size and at the same technology node requires 7.27 mm²,⁴⁵ that is, a 2.85x higher storage density. In each DaDianNao node, we have implemented 16 tiles, all using eDRAMs as their on-chip storage. Each tile has four eDRAM banks (see Figure 6), each bank contains 1024 rows of 4096 bits, thus the total eDRAM capacity in one tile is $4 \times 1024 \times 4096 = 2$ MB. The central eDRAM in each node (see Figure 6) has a size of 4 MB. Therefore, the total node eDRAM capacity is thus $16 \times 2 + 4 = 36$ MB.

Interconnect. Because neurons are the only values transferred, and because these values are heavily reused within each node, the amount of communications, while significant, is usually not a bottleneck. As a result, we did not develop a custom high-speed interconnect for our purpose, we turned to commercially available high-performance interfaces, and we used a HyperTransport (HT) 2.0 IP block. The multinode DaDianNao system uses a simple 2D mesh topology, thus each chip must connect to four neighbors via four HT2.0 IP blocks.

We implemented a custom cycle-accurate bit-accurate C++ simulator for the performance evaluation of the DaDianNao architecture. We also implemented a Verilog version of the DaDianNao node, and have done the synthesis and layout at a 28 nm process (see Figure 7 for the node layout). The node (chip), clocked at 606 MHz, consumes an area of 67.73 mm², and has the peak performance 5585 GOP/s. We evaluated an architecture with up to 64 chips. On a sample of the largest existing neural network layers, we show that a single DaDianNao node achieves a speedup of 21.38x over the NVIDIA K20M GPU and reduces energy by 330.56x on

Figure 6. DaDianNao architecture: tile-based organization of a node (left) and tile architecture (right).

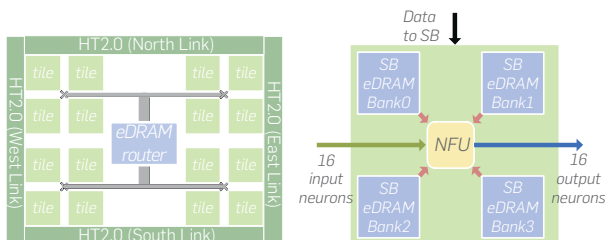
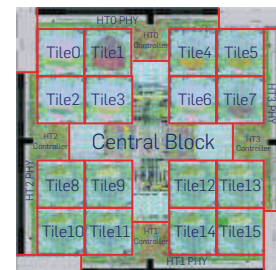


Figure 7. Snapshot of DaDianNao's node layout.



average; a 64-node system achieves a speedup of **450.65x** over the NVIDIA K20M GPU and reduces energy by **150.31x** on average.⁶

4. SHIDIANNAO: A LOW-POWER ACCELERATOR FOR CONVOLUTIONAL NEURAL NETWORK

DaDianNao targets at high-performance ML applications, and integrates eDRAMs in each node to avoid main memory accesses. In fact, the same principle is also applicable to embedded systems, where energy consumption is a critical dimension that must be taken into account. In a recent study, we focused on image applications in embedded systems, and designed a dedicated accelerator (ShiDianNao⁹) for a state-of-the-art deep learning technique called CNN.²⁶

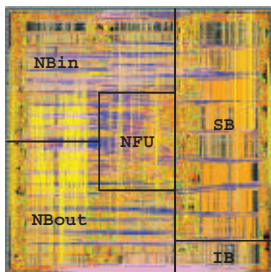
In a broad class of CNNs, it is assumed that each neuron (of a feature map) *shares* its weights with other neurons, making the total number of weights far smaller than in fully connected networks. For instance, a state-of-the-art CNN has 60 millions weights²¹ versus up to 1 billion²⁴ or even 10 billions for state-of-the-art deep networks. This simple property can have profound implications for us: we know that the highest energy expense is related to memory behaviors, in particular main memory (DRAM) accesses, rather than computation.⁴⁰ Due to the small memory footprint of weights in CNNs, it is possible to store a whole CNN within a small on-chip SRAM next to the functional units, and as a result, there is no longer a need for DRAM memory accesses to fetch the CNN model (weights) in order to process each input.

The absence of DRAM accesses combined with a careful exploitation of the specific data access patterns within CNNs allows us to design the ShiDianNao accelerator which is 60x more energy efficient than the DianNao accelerator (c.f., Figure 8). We present a full design down to the layout at a 65 nm process, with an area of 4.86 mm² and a power of 320 mW, but still over 30x faster than NVIDIA K20M GPU. The detailed ShiDianNao architecture was presented at 42nd ACM/IEEE International Symposium on Computer Architecture (ISCA'15).⁹

5. PUDIANNAO: A POLYVALENT MACHINE LEARNING ACCELERATOR

While valid to many different ML tasks, DianNao, DaDianNao, and ShiDianNao can only accommodate neural networks. However, neural networks might not always be the best choice in every application scenario, even if regardless of their high computational complexity. For example, in the

Figure 8. Snapshot of ShiDianNao's layout.



classification of linearly separable data, complex neural networks can easily become over-fitting, and perform worse than a linear classifier. In application domains such as financial quantitative trading, linear regression is more widely used than neural network due to the simplicity and interpretability of linear model.³ The famous No-Free-Lunch theorem, though was developed under certain theoretical assumptions, is a good summary of the above situation: any learning technique cannot perform universally better than another learning technique.⁴⁶ In this case, it is a natural idea to further extend DianNao/DaDianNao to support a basket of diverse ML techniques, and the extended accelerator will have much broader application scope than its ancestors do.

PuDianNao is a hardware accelerator accommodating seven representative ML techniques, that is, *k*-means, *k*-NN, naive bayes, support vector machine, linear regression, classification tree, and deep neural network. PuDianNao consists of several Functional Units (FUs), three data buffers (HotBuf, ColdBuf, and OutputBuf), an instruction buffer (InstBuf), a control module, and a DMA, see Figure 9. The functional unit for machine learning (MLU) is designed to support several basic yet important computational primitives. As illustrated in Figure 10, the MLU is divided into 6 pipeline stages (Counter, Adder, Multiplier, Adder tree, Acc, and Misc), and different combinations of selected stages collaboratively compute primitives that are common in representative ML techniques, such as dot product, distance calculations, counting, sorting, nonlinear functions (e.g., sigmoid and tanh) and so on. In addition, there are some less common operations that

Figure 9. Accelerator architecture of PuDianNao.

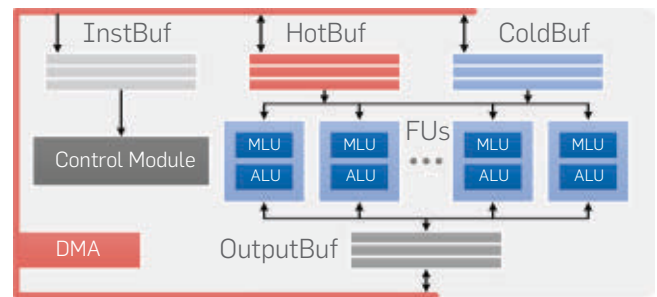
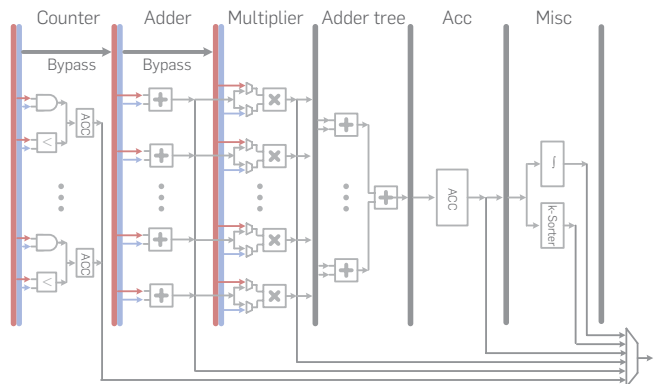


Figure 10. Implementation of Machine Learning Unit (MLU).



are not supported by the MLUs (e.g., division and conditional assignment), which will be supported by small Arithmetic Logic Units (ALUs).

On the other hand, loop tiling can effectively exploit the data locality of ML techniques (as we have revealed in Section 2.2). For tiled versions of different ML techniques, we further observed that average reuse distances of variables often cluster into two or three classes.²⁸ Therefore, we put three separate on-chip data buffers in the PuDianNao accelerator: HotBuf, ColdBuf, and OutputBuf. HotBuf stores the input data which have short reuse distance, ColdBuf stores the input data with relative longer reuse distance, and OutputBuf stores output data or temporary results.

We implemented a cycle-accurate C simulator and a Verilog version of the accelerator, which integrates 16 MLUs (each has 49 adders and 17 multipliers), a 8 KB HotBuf (8 KB), a 16 KB ColdBuf and a 8 KB OutputBuf. PuDianNao can achieve a peak performance of $16 \times (49 + 17) \times 1 = 1056$ GOP/s at 1 GHz frequency, almost approaching the performance of a modern GPU. We have done the synthesis and layout of PuDianNao at a 65 nm process using Synopsys tools, see Figure 11 for the layout. On 13 critical phases of seven representative ML techniques,²⁸ the average speedups of PuDianNao over Intel Xeon E5-4620 SIMD CPU (32 nm process) and NVIDIA K20M GPU (28 nm process) are **21.29x** and **1.20x**, respectively. PuDianNao is significantly more energy-efficient than two general-purpose baselines, given that its power dissipation is only 0.596 W.

6. RELATED WORK

Due to the end of Dennard scaling and the notion of Dark Silicon,^{10,35} architecture customization is increasingly viewed as one of the most promising paths forward. So far, the emphasis has been especially on custom *accelerators*. There have been many successful studies, working on either approximation of program functions using ML,¹¹ or acceleration of ML itself. Yeh *et al.* designed a k -NN accelerator on FPGA.⁴⁷ Manolakis and Stamoulias designed two high-performance parallel array architectures for k -NN.^{33, 40} There have also been dedicated accelerators for k -means^{14, 18, 34} or support vector machine,^{1, 36} due to their broad applications in industry. Majumdar *et al.* proposed an accelerator called MAPLE, which can accelerate matrix/vector and ranking operation used in five ML technique families (including neural network, support vector machine and k -means).^{31, 32} Recent advances on deep learning²³ even triggers the rebirth of hardware neural

network.^{13, 29, 42} However, few previous investigations on ML accelerators can simultaneously address (a)*computational primitives* and (b)*locality properties* of (c)*diverse representative machine learning techniques*.

7. CONCLUSION

In this article, we conduct an in-depth discussion on hardware accelerations of ML. Unlike previous studies that mainly focus on implementing major computational primitives of ML techniques, we also optimize memory structures of the accelerators to reduce/remove main memory accesses, which significantly improve the energy-efficiency of ML compared with systems built with general-purpose CPUs or GPUs.

We devoted DianNao, DaDianNao, and ShiDianNao to neural network (deep learning) techniques, in order to achieve the rare combination of efficiency (due to the small number of target techniques) and broad application scope. However, using large-scale neural networks might not always be a promising choice due to the well-known No-Free-Lunch Theorem,⁴⁶ as well as the distinct requirements in different application scenarios. Therefore, we also develop the PuDianNao accelerator which extends the basic DianNao architecture to support a basket of seven ML techniques.

Following the spirit of PuDianNao, we will further study a pervasive accelerator for ML in our future work, and the purpose is to energy-efficiently accommodate a very broad range of ML techniques. A comprehensive review on representative ML techniques can help to extract common computational primitives and locality properties behind the techniques. However, a straightforward hardware implementation of functional units and memory structures simply matching all extracted algorithmic characteristics could be expensive, because it integrates redundant components to resist significant diversity among ML techniques. This issue can probably be addressed by a *reconfigurable ASIC* accelerator, which supports dynamic reconfiguration of functional units and memory structures to adapt to diverse techniques. Such an accelerator only involves a moderate number of coarse-grained reconfigurable parameters, which is significantly more energy-efficient than Field Programmable Gate Array (FPGA) having millions of controlling parameters.

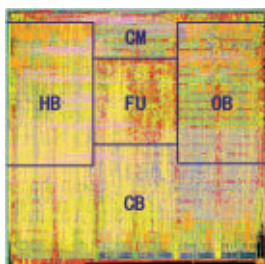
Acknowledgments

This work is partially supported by the NSF of China (under Grants 61133004, 61303158, 61432016, 61472396, 61473275, 61522211, 61532016, 61521092), the 973 Program of China (under Grant 2015CB358800), the Strategic Priority Research Program of the CAS (under Grants XDA06010403 and XDB02040009), the International Collaboration Key Program of the CAS (under Grant 171111KYS-B20130002), the 10,000 talent program, a Google Faculty Research Award, and the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI). 

References

1. Cadambi, S., Durdanovic, I., Jakkula, V., Sankaradas, M., Cosatto, E., Chakradhar, S., Graf, H.P. A massively parallel fpga-based coprocessor for support vector machines. In *17th IEEE Symposium on Field Programmable Custom Computing Machines*, 2009. FCCM'09 (2009) IEEE, 115–122.
2. Chakradhar, S., Sankaradas, M., Jakkula, V., Cadambi, S. A dynamically configurable coprocessor for convolutional neural networks. In *International Symposium on Computer*

Figure 11. Snapshot of PuDianNao's layout.



- Architecture (Saint Malo, France, June 2010). ACM 38(3): 247–257.
3. Chan, E. *Algorithmic Trading: Winning Strategies and Their Rationale*. John Wiley & Sons, 2013.
 4. Chen, T., Chen, Y., Duranton, M., Guo, Q., Hashmi, A., Lipasti, M., Nere, A., Qiu, S., Sebag, M., Temam, O. BenchNN: On the broad potential application scope of hardware neural network accelerators. In *International Symposium on Workload Characterization*, 2012.
 5. Chen, T., Du, Z., Sun, N., Wang, J., Wu, C., Chen, Y., Temam, O. Dianna: A small-footprint high-throughput accelerator for ubiquitous machine-learning. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, (March 2014). ACM 49(4): 269–284.
 6. Chen, Y., Luo, T., Liu, S., Zhang, S., He, L., Wang, J., Li, L., Chen, T., Xu, Z., Sun, N., Temam, O. Dadianna: A machine-learning supercomputer. In *ACM/IEEE International Symposium on Microarchitecture (MICRO)* (December 2014). IEEE Computer Society, 609–622.
 7. Coates, A., Huval, B., Wang, T., Wu, D.J., Ng, A.Y. Deep learning with cots HPC systems. In *International Conference on Machine Learning*, 2013: 1337–1345.
 8. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2009). IEEE, 248–255.
 9. Du, Z., Fasthuber, R., Chen, T., Ienne, P., Li, L., Luo, T., Feng, X., Chen, Y., Temam, O. Shidianna: Shifting vision processing closer to the sensor. In *Proceedings of the 42nd ACM/IEEE International Symposium on Computer Architecture (ISCA15)* (2015). ACM, 92–104.
 10. Esmailzadeh, H., Blem, E., Amant, R.S., Sankaralingam, K., Burger, D. Dark silicon and the end of multicore scaling. In *Proceedings of the 38th International Symposium on Computer Architecture (ISCA)* (June 2011). IEEE, 365–376.
 11. Esmailzadeh, H., Sampson, A., Ceze, L., Burger, D. Neural acceleration for general-purpose approximate programs. In *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture* (Dec 2012). IEEE Computer Society, 449–460.
 12. Farabet, C., Martini, B., Corda, B., Akselrod, P., Culurciello, E., LeCun, Y. NeuFlow: A runtime reconfigurable dataflow processor for vision. In *CVPR Workshop* (June 2011). IEEE, 109–116.
 13. Farabet, C., Martini, B., Corda, B., Akselrod, P., Culurciello, E., LeCun, Y. Neuflow: A runtime reconfigurable dataflow processor for vision. In *2011 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2011). IEEE, 109–116.
 14. Frery, A., de Araujo, C., Alice, H., Cerqueira, J., Loureiro, J.A., de Lima, M.E., Oliveira, M., Horta, M., et al. Hyperspectral images clustering on reconfigurable hardware using the k-means algorithm. In *Proceedings of the 16th Symposium on Integrated Circuits and Systems Design, 2003. SBCCI 2003* (2003). IEEE, 99–104.
 15. Hameed, R., Qadeer, W., Wachs, M., Azizi, O., Solomatnikov, A., Lee, B.C., Richardson, S., Kozyrakis, C., Horowitz, M. Understanding sources of inefficiency in general-purpose chips. In *International Symposium on Computer Architecture* (New York, New York, USA, 2010). ACM, 38(3): 37–47.
 16. Hinton, G., Srivastava, N. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv: ...*, 1–18, 2012.
 17. Hussain, H.M., Benkrid, K., Seker, H., Erdogan, A.T. Fpga implementation of k-means algorithm for bioinformatics application: An accelerated approach to clustering microarray data. In *2011 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)* (2011). IEEE, 248–255.
 18. Keckler, S. Life after Dennard and how I learned to love the Picojoule (keynote). In *International Symposium on Microarchitecture*, Keynote presentation, Sao Paolo, Dec. 2011.
 19. Kim, J.Y., Kim, M., Lee, S., Oh, J., Kim, K., Yoo, H.-J.A. GOPS 496mW real-time multi-object recognition processor with bio-inspired neural perception engine. *IEEE Journal of Solid-State Circuits* 45, 1 (Jan. 2010), 32–45.
 20. Krizhevsky, A., Sutskever, I., Hinton, G. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems* (2012), 1–9.
 21. Krizhevsky, A., Sutskever, I., Hinton, G. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems* (2012) 1–9.
 22. Larkin, D., Kinane, A., O'Connor, N.E. Towards hardware acceleration of neuroevolution for multimedia processing applications on mobile devices. In *Neural Information Processing* (2006). Springer, Berlin Heidelberg, 1178–1188.
 23. Le, Q.V. Building high-level features using large scale unsupervised learning. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2013). IEEE, 8595–8598.
 24. Le, Q.V., Ranzato, M.A., Monga, R., Devin, M., Chen, K., Corrado, G.S., Dean, J., Ng, A.Y. Building high-level features using large scale unsupervised learning. In *International Conference on Machine Learning*, June 2012.
 25. LeCun, Y., Bengio, Y., Hinton, G. Deep learning. *Nature* 521, 7553 (2015), 436–444.
 26. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86 11 (1998), 2278–2324.
 27. Li, S., Ahn, J.H., Strong, R.D., Brockman, J.B., Tullsen, D.M., Jouppi, N.P. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 42 (New York, NY, USA, 2009). ACM, 469–480.
 28. Liu, D., Chen, T., Liu, S., Zhou, J., Zhou, S., Temam, O., Feng, X., Zhou, X., Chen, Y. Pudianna: A polyvalent machine learning accelerator. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)* (2015). ACM, 369–381.
 29. Maashri, A.A., Debole, M., Cotter, M., Chandramoorthy, N., Xiao, Y., Narayanan, V., Chakrabarti, C. Accelerating neuromorphic vision algorithms for recognition. In *Proceedings of the 49th Annual Design Automation Conference* (2012). ACM, 579–584.
 30. Maeda, N., Komatsu, S., Morimoto, M., Shimazaki, Y. A 0.41 μ a standby leakage 32 kb embedded SRAM with low-voltage resume-standby utilizing all digital current comparator in 28 nm hkm CMOS. In *International Symposium on VLSI Circuits (VLSIC)*, 2012.
 31. Majumdar, A., Cadambi, S., Becchi, M., Chakradhar, S.T., Graf, H.P. A massively parallel, energy efficient programmable accelerator for learning and classification. *ACM Trans. Arch. Code Optim. (TACO)* 9, 1 (2012), 6.
 32. Majumdar, A., Cadambi, S., Chakradhar, S.T. An energy-efficient heterogeneous system for embedded learning and classification. *Embedded Systems Letters* 3, 1 (2011), 42–45.
 33. Manolakos, E.S., Stamoulas, I. IP-cores design for the KNN classifier. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)* (2010). IEEE, 4133–4136.
 34. Maruyama, T. Real-time k-means clustering for color images on reconfigurable hardware. In *18th International Conference on Pattern Recognition (ICPR)* (Aug 2006). IEEE, Volume 2, 816–819.
 35. Muller, M. Dark silicon and the internet. In *EE Times "Designing with ARM" Virtual Conference*, 26, 70(2010), 285–288.
 36. Papadonikolakis, M., Bouganis, C. A heterogeneous FPGA architecture for support vector machine training. In *2010 18th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)* (May 2010). IEEE, 211–214.
 37. Qadeer, W., Hameed, R., Shacham, O., Venkatesan, P., Kozyrakis, C., Horowitz, M.A. Convolution engine: Balancing efficiency & flexibility in specialized computing. In *International Symposium on Computer Architecture*, 2013). ACM, 41(3), 24–35.
 38. Sermanet, P., Chintala, S., LeCun, Y. Convolutional neural networks applied to house numbers digit classification. In *Pattern Recognition (ICPR)*, ..., 2012.
 39. Sermanet, P., LeCun, Y. Traffic sign recognition with multi-scale convolutional networks. In *International Joint Conference on Neural Networks* (July 2011). IEEE, 2809–2813.
 40. Stamoulas, I., Manolakos, E.S. Parallel architectures for the KNN classifier—design of soft IP cores and FPGA implementations. *ACM Transactions on Embedded Computing Systems (TECS)* 13, 2 (2013), 22.
 41. Swanson, S., Michelson, K., Schwerin, A., Oskin, M. Wavescalar. In *ACM/IEEE International Symposium on Microarchitecture (MICRO)* (Dec 2003). IEEE Computer Society, 291.
 42. Temam, O. The rebirth of neural networks. In *International Symposium on Computer Architecture*, (2010).
 43. Temam, O. A defect-tolerant accelerator for emerging high-performance applications. In *International Symposium on Computer Architecture* (Sep 2012). Portland, Oregon, 40(3), 356–367.
 44. Vanhoucke, V., Senior, A., Mao, M.Z. Improving the speed of neural networks on CPUs. In *Deep Learning and Unsupervised Feature Learning Workshop (NIPS)* (2011). Vol. 1.
 45. Wang, G., Anand, D., Butt, N., Cestero, A., Chudzik, M., Ervin, J., Fang, S., Freeman, G., Ho, H., Khan, B., Kim, B., Kong, W., Krishnan, R., Krishnan, S., Kwon, O., Liu, J., McStay, K., Nelson, E., Nummy, K., Parries, P., Sim, J., Takalkar, R., Tessier, A., Todi, R., Malik, R., Stiffler, S., Iyer, S. Scaling deep trench based EDRAM on SOI to 32 nm and beyond. In *IEEE International Electron Devices Meeting (IEDM)* (2009). IEEE, 1–4.
 46. Wolpert, D.H. The lack of a priori distinctions between learning algorithms. *Neural Comput.* 8, 7 (1996), 1341–1390.
 47. Yeh, Y.-J., Li, H.-Y., Hwang, W.-J., Fang, C.-Y. Fpga implementation of KNN classifier based on wavelet transform and partial distance search. In *Image Analysis* (June 2007). Springer Berlin Heidelberg, 512–521.

Yunji Chen, Tianshi Chen, Zhiwei Xu, and Ninghui Sun ({cyj, chentianshi, zxu, snh}@ict.ac.cn), SKL of Computer Architecture, ICT, CAS, China.

Olivier Temam (olivier.temam@inria.fr), Inria Saclay, France.

Technical Perspective

FPGA Compute Acceleration Is First About Energy Efficiency

By James C. Hoe

THE FOLLOWING PAPER presents a research deployment of Field Programmable Gate Arrays (FPGAs) in a Microsoft Bing datacenter. The FPGAs enabled more efficient search processing directly at the digital logic level. This pioneering work is the first to successfully demonstrate at scale the idea of FPGAs as an effective large-scale, first-class component in cloud computing. Following this landmark effort, Microsoft has launched full-scale production deployment of FPGA accelerators in its new datacenter servers for a range of cloud services.

For most of its 30-year existence, FPGA technology had primarily served as an alternative to application-specific integrated circuits (ASICs), with only niche applications in computing. Today, besides Microsoft's activities, we are also finding Intel and IBM adding FPGAs as programmable computing substrates to their product lines. At the root of the computing industry's current embrace of FPGAs is the same Power Wall struggle that brought about the transition to multicore microprocessors in the last decade.

In the decades prior, single-threaded microprocessors enjoyed a regular doubling of compute performance with each new Very Large Integrated Circuits (VLSI) scaling generation, taking advantage of the more numerous and faster transistors. However, each new generation of faster microprocessors had also required more power. The Power Wall is less about supplying power but more about removing the resulting heat fast enough. For set upper bounds in cost, weight, size, and noise of the cooling apparatus, there is a limit to how fast heat can be extracted from a microprocessor die. Microprocessors remained well below market-set economical cooling limits until the 1990s. Despite the best concerted efforts from software down to material science to reign in the power increase in the ensuing years, single-threaded microprocessors ran out of cooling headroom to sustain their per-

formance improvements by the middle of the last decade.

In the present power-constrained design regime—whether dictated by the cooling and packaging of a microprocessor or the air handling capacity of a datacenter, plus nowadays actual supply-side considerations in battery powered devices—the problem of getting more performance (“operations per second”) requires a solution that can somehow do so while expending less “energy per operation.”

Parallelism provided the first solution to running faster while using less energy in each step. As a rule of thumb, it takes proportionally more power to increase the performance of a sequential task. Therefore, when parallelism is available, one could use parallel execution to reduce power instead of for speedup. In an illustrative example, given design A with throughput performance Throughput_A at Power_A and a lower-performing design B with $\text{Throughput}_B = \text{Throughput}_A/N$ and $\text{Power}_B < \text{Power}_A/N$, we can use N copies of B to complete the same number of tasks per second as A at less power, and we can use $>N$ copies of B to exceed the throughput of A at the same Power_A . Both multicore microprocessors and GPGUs are practitioners of this principle. FPGA computing can take this to further extremes, delivering high overall performance using a sea of individually slow processing elements that are very energy efficient per operation. This route to energy efficiency is of course only applicable when the computing task of interest is amenable to a high degree of parallelization.

Relative to microprocessors, FPGA computing has another source of energy efficiency. The majority of the power in a microprocessor is consumed by the overhead in presenting the simplifying von Neumann abstraction to the programmer and in ensuring good performance across a wide range of program behaviors. FPGA computing avoids

these abstraction overheads and at the same time gains unrestricted optimization freedom, in particular the ability to exploit all forms and granularity of parallelism in a task. In return, FPGA computing places a great burden on application developers to design at a low level of specificity to meet functional and performance objectives. How to simplify application development, while retaining FPGA's efficiency advantage, is an important and challenging problem.

Mapping computations to ASICs has the same benefits as FPGAs. In fact, an ASIC implementation can be significantly more energy efficient than an FPGA implementation due to the overhead in making the FPGAs' logic substrate reprogrammable. But reprogrammability is very important to computing. As argued in the following paper, besides the utility of repurposing a hardware investment over its multiyear ownership, a particular accelerated task of interest can evolve too quickly to be committed into ASIC development time and cost scales.

Finally, it should be noted that FPGA computing is not the answer in every scenario or across all trade-offs. Ultimately, the quest for efficiency is a matter of using the right tool for the job.

Driven by the contradicting needs for more performance and better energy efficiency, parallel computing in the form of multicore microprocessors seemingly exploded into the commercial mainstream in the last decade. With the Catapult effort, perhaps we are again seeing the start of something equally transformative in the pursuit of still higher performance and energy efficiency. As we watch the current exciting developments unfold across the computing industry, we should also recognize the multiple decades of prior work that led up to this pivotal time. ■

James C. Hoe (jhoe@cmu.edu) is a professor of electrical and computer engineering at Carnegie Mellon University, Pittsburgh, PA.

Copyright held by author.

A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services

By Andrew Putnam, Adrian M. Caulfield, Eric S. Chung, Derek Chiou, Kypros Constantinides, John Demme, Hadi Esmaeilzadeh, Jeremy Fowers, Gopi Prashanth Gopal, Jan Gray, Michael Haselman, Scott Hauck, Stephen Heil, Amir Hormati, Joo-Young Kim, Sitaram Lanka, James Larus, Eric Peterson, Simon Pope, Aaron Smith, Jason Thong, Phillip Yi Xiao, and Doug Burger

Abstract

Datacenter workloads demand high computational capabilities, flexibility, power efficiency, and low cost. It is challenging to improve all of these factors simultaneously. To advance datacenter capabilities beyond what commodity server designs can provide, we designed and built a composable, reconfigurable hardware fabric based on field programmable gate arrays (FPGA). Each server in the fabric contains one FPGA, and all FPGAs within a 48-server rack are interconnected over a low-latency, high-bandwidth network.

We describe a medium-scale deployment of this fabric on a bed of 1632 servers, and measure its effectiveness in accelerating the ranking component of the Bing web search engine. We describe the requirements and architecture of the system, detail the critical engineering challenges and solutions needed to make the system robust in the presence of failures, and measure the performance, power, and resilience of the system. Under high load, the large-scale reconfigurable fabric improves the ranking throughput of each server by 95% at a desirable latency distribution or reduces tail latency by 29% at a fixed throughput. In other words, the reconfigurable fabric enables the same throughput using only half the number of servers.

1. INTRODUCTION

Cloud computing has emerged as a dominant paradigm for delivering scalable, reliable, and cost-effective online services to businesses and clients across the world. According to the IDC, public spending in IT cloud services will grow to more than \$127B in 2016 as the adoption of cloud computing accelerates worldwide.¹⁰ This major shift will offer enormous potential in unlocking new applications and in improving the performance, security, and cost of computing.

The majority of today's cloud services are realized using datacenters, which typically comprise tens if not hundreds of thousands of servers built from commodity components such as general-purpose processors, memory, storage, and networking. Datacenters are shared across applications and services, providing economies of scale, reliability, scalability, and shared infrastructure management.

Datacenter operators have traditionally relied upon continuous improvements in the performance and efficiency of

general-purpose processors to make datacenters even more powerful and cost-effective. These improvements have been largely driven by Moore's law that predicts an exponential growth in the number of transistors over time, and by Dennard scaling⁸ that predicts constant power consumption even as the number of transistors increase within a fixed silicon area.

In recent years, Dennard scaling has virtually ended, resulting in power consumption being roughly proportional to the number of switching transistors. Thus, even though Moore's law continues to provide more transistors for the time being, a larger number of transistors must switch proportionally less frequently to maintain constant power consumption. In addition, as transistors become smaller, they are becoming more expensive to manufacture, making it even less attractive to pack more and more transistors onto a single chip.

1.1. Specialized hardware in the datacenter

One way to improve the performance and efficiency of datacenter servers is to make better uses of "power-limited" transistors by specializing servers and their components to a particular task. In the academic literature, specialization has been shown to achieve 10 $\times$ –100 $\times$ or more improvement in energy efficiency over general-purpose processors in many cases, such as for Memcached^{5, 14} compression/decompression,^{13, 15} K-means clustering,^{9, 12} and parts of web search.²⁰ However, while specializing servers for specific workloads can provide significant efficiency gains, doing so is problematic in the datacenter for several major reasons.

First, datacenters, by their very nature, support a wide variety of applications and specializing for one service will likely cause inefficiencies and add cost to any other services sharing the platform. Second, specialization compromises homogeneity, which is highly desirable in the datacenter environment to reduce management issues and to provide a consistent platform on which applications can rely upon. Third, datacenter services evolve rapidly, making highly specialized hardware features impractical and quickly obsoleted. Thus, datacenter providers face a conundrum: they need continued improvements in performance and

The original version of this paper was published in the *Proceedings for the 41st ACM/IEEE International Symposium on Computer Architecture* (June 14–18, 2014, Minneapolis, MN), 13–24.

All authors contributed to this work while employed by Microsoft.

efficiency, but cannot obtain those improvements from any combination of standard general-purpose processors and static specialized hardware.

1.2. Flexible specialized hardware

Programmable hardware, in the form of field programmable gate arrays (FPGAs), are devices that could potentially reconcile the need for both flexibility and energy efficiency. FPGAs can be imagined as silicon “Legos”—collections of simple logic blocks that can be configured and composed to implement arbitrary circuits that run at very high efficiency relative to general-purpose processors. While less energy efficient than hard-wired Application Specific Integrated Circuits (ASICs), FPGAs can be rapidly reconfigured and adapted to changing workloads over the lifetime of the server. However, as of this writing, FPGAs have not been widely deployed as compute accelerators in either datacenter infrastructure or client devices.

One challenge traditionally associated with FPGAs is the need to fit the accelerated function into the available reconfigurable area on one chip. Today’s FPGAs can be virtualized using runtime reconfiguration to support more functions than could fit on a single device. However, the amount of state that needs to be saved and restored, along with current reconfiguration times for standard FPGAs make this approach too slow to be practical. Multiple FPGAs could provide more silicon resources, but are extremely difficult to fit into a conventional datacenter server. Even if sufficient space were available, multiple FPGAs per server would cost more, consume more power, are wasteful when there are more FPGAs than needed—and even less useful when there are still not enough FPGAs to implement the application. On the other hand, being restricted to a single FPGA per server restricts the workloads that might be accelerated and could make the associated gains too small to justify the cost.

1.3. Catapult reconfigurable fabric

This article describes a reconfigurable fabric called Catapult, which uses FPGAs to provide the performance and efficiency gains of specialized hardware while simultaneously satisfying the strict requirements of the datacenter. As illustrated in Figure 1, the Catapult fabric is embedded into racks of servers in the form of a small board with a medium-sized FPGA and local DRAM attached to each server. A unique characteristic of Catapult is that FPGAs are directly wired together in a high-bandwidth, low-latency network, allowing services to allocate groups of FPGAs to provide the necessary reconfigurable area to implement the desired functionality.

While proving functionality on a small number of servers shows the potential of the design, datacenters are characterized by massive scales and severe power, cost, and reliability constraints. To demonstrate the potential of this technology at datacenter scale, we tested the Catapult reconfigurable fabric, running a widely deployed web search workload augmented with failure handling, on a bed of 1632 servers equipped with FPGAs. The experiments show that large gains in search throughput and latency are achievable on a real, complex commercial workload using the large-scale reconfigurable fabric.

Compared to a pure software implementation, the Catapult fabric achieves a 95% improvement in throughput at each ranking server with an equivalent latency distribution. At the same throughput as software, Catapult reduces tail latency by 29%. In other words, adding the FPGA enables the same throughput using only half the number of servers. The system is able to run stably for long periods, with a failure handling service quickly reconfiguring the fabric upon errors or machine failures. The rest of this article describes the Catapult architecture and our measurements in more detail.

2. BACKGROUND

FPGAs are digital chips that can be programmed (and reprogrammed) for implementing complex digital logic. Conceptually, FPGAs consist of an array of programmable logic elements, connected by a programmable routing network that carries signals from where they are generated to where they are consumed. Each of these features are controlled by memory cells that are configured by the end user. Designs for an FPGA

Figure 1. Each server in a rack has a local FPGA attached to the host via PCI Express. FPGAs communicate with their neighbors using a private low-latency, high-bandwidth serial network. Multiple FPGAs can be allocated to a single service, such as the Bing ranking, without going through host CPUs.

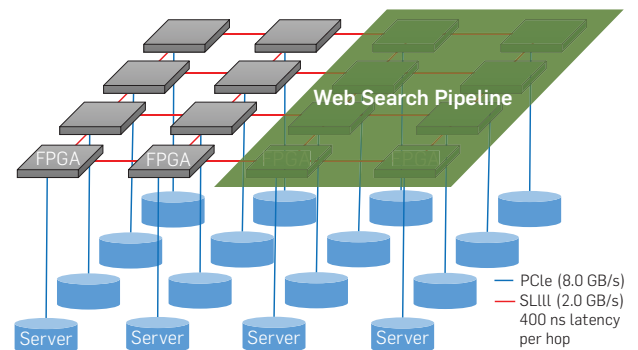
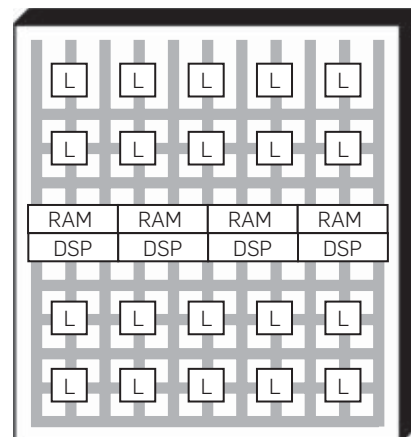


Figure 2. Block diagram representing one tile of an abstract FPGA. Each tile is composed of generic logic gates (L), embedded memory (RAMs), and specialized arithmetic units (DSPs). These basic tiles are replicated to create larger and larger FP-GAs.



are typically developed in a hardware description language (HDL) such as VHDL or Verilog; these designs are compiled down to an FPGA program (called a bitstream), similar to how a software language is compiled into a software executable. However, specifying a design in HDL is significantly more complex than for a typical software language, and compilation may take hours to complete. Once the bitstream is generated, it can be loaded into an FPGA in a matter of seconds, configuring the FPGA to implement the desired computation.

FPGAs cover a middle-ground in performance/efficiency and flexibility compared to fully custom ASICs and general-purpose CPUs. General-purpose CPUs are the most flexible platform, capable of implementing any application. But this flexibility generally comes at the cost of a 100× or greater reduction in performance and energy efficiency compared to an ASIC designed for the same task. However, when the application changes, the CPU can still run the new application by simply recompiling the software. When the target application for an ASIC changes, it likely requires designing a new chip—a process which typically takes months and carries huge development costs.

FPGAs combine aspects of both CPUs and ASICs. They can be reprogrammed as applications change to provide CPU-like flexibility, but the program produces hardware which is specialized to the application, providing performance and power efficiency closer to ASICs. The generic FPGA logic can be configured to exploit huge amounts of fine-grained parallelism and can implement very complex pipelined structures that can be several orders-of-magnitude faster and lower power than their software equivalents.

Modern FPGAs provide millions of gates of random logic and can include multiple megabytes of internal storage that supports thousands of reads and writes on every clock cycle. These chips also incorporate smaller fully custom blocks, such as complex embedded arithmetic elements (DSP blocks), high-speed I/O, IP protocol blocks, and may include complete microprocessors as well, either as hardened subsystems or by mapping the microprocessor logic into the programmable logic of the chip.

FPGAs have been available for several decades and they have huge potential for accelerating a wide variety of applications. Yet despite this promise, FPGAs have not been widely deployed as compute accelerators, even in datacenter environments where their potential for flexibility, power efficiency, and performance should make them extremely attractive.

The Catapult architecture described here unlocks the potential of FPGAs in the datacenter. To achieve this, the architecture has to be distributed, scalable, robust, and work on real-world datacenter-scale workloads. In the following sections, we describe the architecture, and show how production workloads can finally unlock the potential efficiency and performance gains promised for so long by FPGAs.

3. CATAPULT ARCHITECTURE

Datacenters are a challenging environment for any new technology to succeed. The scale alone demands extremely high reliability, efficiency, and low cost. The rapid pace of

application development requires flexibility and robustness. And the physical constraints and uptime requirements make it largely impractical to modify or upgrade the hardware after initial delivery.

To succeed in the datacenter environment, an FPGA-based reconfigurable fabric must (at a minimum) meet the following requirements:

- preserve server homogeneity to avoid complex management of heterogeneous servers,
- scale to large workloads that might not fit into a single FPGA,
- avoid consuming too much power or network bandwidth,
- avoid single points of failure, and have minimal impact on reliability,
- provide positive return on investment (ROI), and
- operate within the space and power confines of existing servers.

These requirements guided the architectural choices we made throughout the Catapult system development.

3.1. Integration

How to integrate FPGAs into the datacenter is perhaps the most important consideration when designing a reconfigurable fabric. We investigated a variety of approaches which can be roughly broken into two categories based on how they integrate with conventional servers: “networked” and “integrated.” Networked designs add FPGAs to special FPGA-enabled servers, and arrange the servers either as entire racks of specialized servers or embedding some number of specialized servers in otherwise conventional racks. Integrated designs add FPGAs directly inside the conventional servers, requiring no specialized servers and no network communication to reach the FPGA.

Networked designs have been developed and deployed in High Performance Computing (HPC) environments. While HPC systems are not subject to the same scale, cost, power, and homogeneity constraints as the datacenter, they are one place where integrating FPGAs with CPUs has seen some success. Entire large systems have been built using only specialized servers, including the Cray XD-1,⁷ Novo-G,¹¹ and QP.¹⁸ Examples of specialized servers that could be integrated with racks of conventional datacenter servers include the Convey HC-2,⁶ Maxeler MPC series,¹⁷ BeeCube BEE4,⁴ and SRC MAPstation.¹⁹ In fact, embedding a few specialized servers into each rack is the approach that we first took early in the project. However, as we learned from our first prototype, the networked design approach is inappropriate for datacenter use for several reasons.

First, specialized racks and servers are single points of failure, where the failure of the specialized nodes impacts many dependent conventional servers—amplifying the impact to uptime and overall service reliability.

Second, specialized racks and servers require separate cooling and power provisioning, as well as different software, firmware, and spare parts provisioning, making management and maintenance more difficult.

Third, all communication with these racks/servers goes through the existing network, which (in a datacenter) is not typically designed for the many-to-one communication patterns to the specialized racks.

Finally, all communication between the portion of the application running on the conventional server and the portion in the FPGA requires going over the network, which, even in the datacenter, can have high latency, unreliable delivery, and variable performance. This is particularly difficult when traffic is exhibiting the many-to-one communication pattern. The unreliable performance makes partitioning the application between CPU and FPGA extremely difficult, especially for latency-sensitive user-facing applications, and reduces the likelihood that an application can be beneficially offloaded to the reconfigurable fabric.

These issues diminish in severity with an increasingly distributed design. At the extreme is the integrated design—adding an FPGA to every server. In integrated designs, an FPGA failure only impacts one server. All servers have the same cooling and power constraints. CPU to FPGA communication does not need to go over the network at all. And attaching the FPGA directly to the CPU greatly diminishes communication latency and variance, enabling finer-grain and more reliable offloading from the CPU to the FPGA.

The homogeneous nature of the integrated design enables additional benefits which are key to keeping the designs cost-effective. Homogeneous computing resources can be divided at arbitrary granularities, easing both short-term and long-term provisioning of servers. Homogeneous designs also improve economies of scale leading to more cost-effective designs.

3.2. Scalability

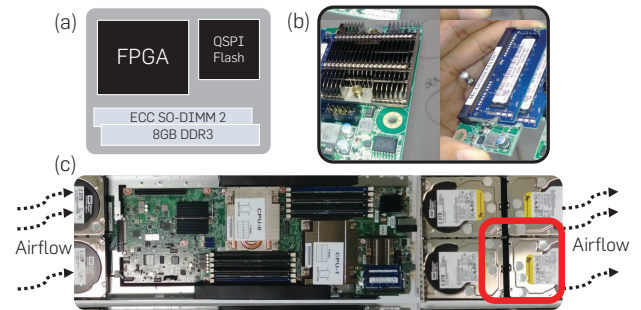
On its own, integrating only one FPGA per server either limits applications to those that can fit into the resources of a single FPGA or suffers the same high-latency and unreliability problems with communicating over the network as specialized racks and servers.

To overcome this shortcoming, we built a specialized network, in addition to the existing Ethernet network, to facilitate FPGA-to-FPGA communication. This specialized network takes advantage of the low-latency, high-speed serial I/O available in the FPGAs to create a two-dimensional 6×8 torus network. The torus topology balances routability, resilience, and cabling complexity. Each inter-FPGA network link supports 20 Gbits per second in both directions, at sub-microsecond latency per hop, and requires only passive copper cables with no additional networking costs such as network interface cards or switches. Another HPC system, Maxwell,³ takes a similar approach, but again targets HPC workloads and constraints.

3.3. FPGA board design

Figure 3 shows the FPGA board and the server into which it installs.¹⁶ The board incorporates a midrange Altera Stratix V D5 FPGA,² which we selected to balance the cost of deploying thousands of FPGAs against the FPGA capacity. The board also includes two banks of DDR3-1600 DRAM. The PCIe and

Figure 3. The FPGA board and the server into which it installs. (a) Block diagram of the FPGA board. (b) Picture of the manufactured board. (c) Diagram of the 1U, half-width server that hosts the FPGA board. The air flows from left to right, leaving the FPGA in the exhaust of both CPUs.



inter-FPGA network are wired to a connector on the bottom of the board that plugs directly into the motherboard. To avoid changes to the server itself, we custom designed the board to fit within a small 10 cm $\times$ 9 cm $\times$ 16 mm slot occupying the rear of a 1U (4.45 cm high), half-width server, which offered sufficient power and cooling only for a standard 25-Watt PCIe peripheral device. These physical constraints are challenging for FPGAs, but are prohibitive for modern compute-class Graphics Processing Units (GPUs).

3.4. Shell architecture

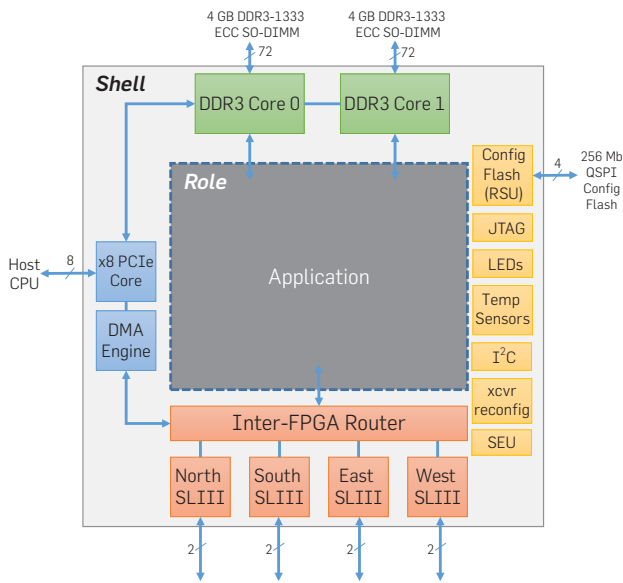
In typical FPGA programming environments, the user is often responsible for developing not only the application itself but also building and integrating system functions required for data marshaling, host-to-FPGA communication, and inter-chip FPGA communication (if available). System integration places a significant burden on the user and can often exceed the effort needed to develop the application itself. This development effort is often not portable to other boards, making it difficult for applications to work on future platforms.

Motivated by the need for user productivity and design re-usability when targeting the Catapult fabric, we logically divide all programmable logic into two partitions: the *shell* and the *role*. The *shell* is a reusable portion of programmable logic common across applications targeting the same board—while the *role* is the application logic itself, restricted to a large fixed region of the chip.

Figure 4 shows a block-level diagram of the shell architecture, consisting of PCIe with a custom DMA engine, two DRAM controllers, four high-speed inter-FPGA links running the Serial Lite III protocol, the torus network router, reconfiguration logic, single event upset (SEU) scrubbing, and debugging interfaces.

Role designers access convenient and well-defined interfaces and capabilities in the shell (e.g., PCIe, DRAM, routing, etc.) without concern for managing system correctness.

The common shell also simplifies software by ensuring that all applications support the same API functions for data movement, reconfiguration, and health monitoring. For example, we co-designed the PCIe core and DMA engine to achieve very low latency, taking fewer than 10 μ s for transfers

Figure 4. Components of the shell architecture.

of 16 KB or less, and multithreading safety. Software developers use simple *send* and *receive* calls to transmit data, and role designers simply read and write to the PCIe interface FIFOs.

The shell consumes 23% of each FPGA, although extra capacity can be obtained by discarding unused functions. If desired, partial reconfiguration allows for dynamic switching between roles while the shell remains active—even routing inter-FPGA traffic while a reconfiguration is taking place—but it comes at the cost of reduced logic area and RAM availability to the application.

3.5. Resiliency

At datacenter scales, providing resiliency in the face of hardware failures is essential given that such failures occur frequently, while demand for hardware availability is consistently high. For instance, the fabric must stay available in the presence of errors, failing hardware, reboots, and updates to the implemented algorithm. FPGAs can potentially corrupt their neighbors or crash the hosting servers if care is not taken during reconfiguration. Our reconfigurable fabric further requires a custom protocol to reconfigure groups of FPGAs, remap services to recover from failures, and report errors to the management software. In addition, ECC is used on all external memories, and SEU detection and correction is implemented on the FPGA's configuration memory.

3.6. Total cost of ownership

To balance the expected per-server performance gains versus the necessary increase in total cost of ownership, including both increased capital costs and operating expenses, we set aggressive power and cost goals to achieve a positive ROI. We are unable to give cost numbers for our production servers due to their business sensitivity; however, we can say

that adding the FPGA card and network cost less than 30% in the total cost of ownership, including a limit of 10% for total server power.

3.7. Datacenter deployment

To test this architecture on a critical production-scale datacenter service at scale, we manufactured and deployed the fabric in a production datacenter. The deployment consisted of a total of 1632 machines, that were organized into in 17 server racks. Each server uses two 12-core Intel Xeon CPUs, 64 Gbytes of DRAM, and two solid-state drives (SSDs) in addition to four hard-disk drives. The machines have a standard 10-Gbit Ethernet network card connected to a 48-port top-of-rack switch, which in turn connects to the broader datacenter network. The FPGA daughter cards and cable assemblies were tested at manufacture and again at system integration. At deployment, we discovered that seven cards (0.4%) had a hardware failure and that one of the 3264 links (0.03%) in the cable assemblies was defective. Since then, after several months of operation, we have seen no additional hardware failures.

4. APPLICATION CASE STUDY

To study the potential impact of the Catapult fabric, we ported a significant fraction of Microsoft Bing's web search ranking engine into reconfigurable hardware. Aside from being a representative datacenter-scale workload, Bing consumes a significant fraction of the datacenter capacity at Microsoft and is used to power a number of popular services such as Yahoo! Search, Apple Siri, and search on Xbox One. As an interactive workload, Bing requires both low latency and high bandwidth simultaneously. Furthermore, Bing has strict resiliency requirements, is operationally complex, and is programmed using tens of thousands of lines of production-quality C++ code—making it an excellent candidate for gauging the viability of reconfigurable computing at scale in a production environment.

In our case study, we devoted the majority of our efforts to the Ranking portion of the Bing search engine, which presented the largest opportunity for hardware acceleration. Over 30K lines of C++ code were manually ported to the Catapult fabric using the Verilog HDL. The implementation generates results that are identical to software—even reproducing known software bugs. As will be discussed in further detail below, our implementation requires a total of seven FPGAs to run a single instance of the service—plus one additional spare for redundancy. Without the availability of the low-latency, high-bandwidth network that interconnects multiple FPGAs, accelerating the Ranking service would not have been feasible.

4.1. Accelerating Bing ranking using FPGAs

The Bing search engine is logically divided into multiple software services spanning a large number of servers in the datacenter. When a user's search query is processed by the Bing search engine, it is first submitted to a front-end cache service that stores and delivers the results of previously submitted and popular queries. If the search query cannot be located in the cache, it is forwarded to the Selection and

Ranking services, which perform the actual computation needed to generate search results.

The Selection service is responsible for accepting a user query and selecting which of the billions of documents (e.g., web pages) on the Internet are worthwhile candidates. The Ranking service further takes these selected documents and runs them through a sophisticated ranking algorithm that determines the order in which these documents should be presented to the user.

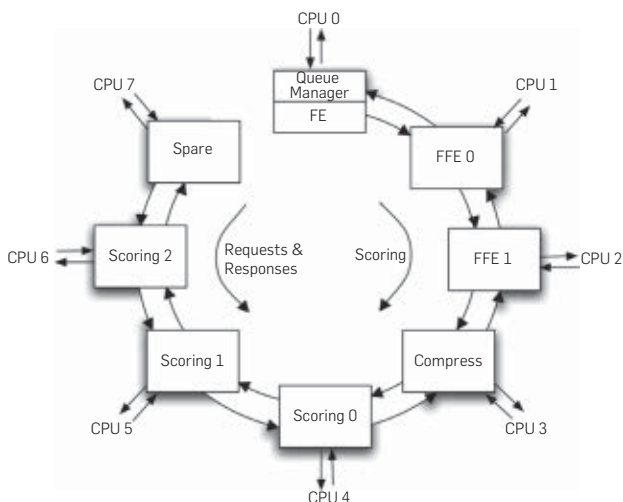
The input to the Bing Ranking algorithm is a “hit vector” that corresponds to a document-query pair arriving from the upstream Selection service. A hit vector efficiently encodes the locations in which words in a user’s query appear within a given document (e.g., web page). The output of the Ranking algorithm is a document “score,” which is used to determine the position in which the document is presented to the user.

Conceptually, the Bing ranking algorithm is divided into three major stages: (1) Feature Extraction (FE), (2) Free-Form Expressions (FFE), and (3) Machine-Learned Scoring (MLS). Figure 5 illustrates a hardware processing pipeline that allocates these stages to an eight-node FPGA pipeline: one FPGA for FE, two for FFEs, one for a compression stage that increases scoring engine efficiency, and three for MLS. The eighth FPGA is a spare that allows the ring to be reconfigured and rotated to keep the ranking pipeline alive in the event of a failure.

4.2. Feature extraction

In FE, interesting characteristics of a given document are dynamically extracted based on a user’s search query. As a simple example, the *NumberOfOccurrences* feature simply counts up the number of times a query happens to appear within a given document. In Bing Ranking, there are potentially up to thousands of features that are computed for a given document-query pair.

Figure 5. Mapping of ranking roles to FPGAs on the reconfigurable fabric. Data is sent from each server to the queue manager. It is then dispatched through the seven FPGA computation stages, and the results are sent back to the source server.



Our FPGA accelerator achieves a significant advantage over software using a form of multiple-instruction, single-data computation (MISD). In the FPGA, we instantiate 43 unique feature-extraction state machines that are used to compute nearly 4500 features per document-query pair in parallel. Each feature is an independent instruction stream, and the data is a single document—hence the MISD parallelism—which the FPGA can exploit far more effectively than CPUs and GPUs.

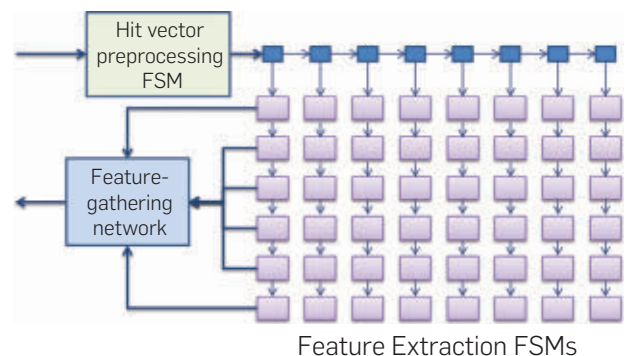
Each state machine reads each hit vector one item at a time and performs a local calculation. For some features that have similar computations, a single state machine is responsible for calculating values for multiple features. As an example, the *NumberOfOccurrences* feature simply counts up how many times each term (i.e., word) in the query appears. At the end of a document, the state machine outputs all non-zero feature values; for *NumberOfOccurrences*, this could be up to the number of terms in the query.

To support a large collection of state machines working in parallel on the same input data at a high clock rate, we organize the blocks into a tree-like hierarchy and replicate each input several times. Figure 6 shows the logical organization of the FE hierarchy. Hit vectors are fed into a hit vector processing state machine, which produces a series of control and data tokens that the various feature state machines process. Each state machine processes each hit vector at a rate of one to two clock cycles per token. When a state machine finishes its computation, it emits one or more feature indexes and values that are fed into the feature-gathering network that coalesces the results from the 43 state machines into a single output stream for the downstream FFE stages. Inputs to FE are double buffered to increase throughput.

4.3. Free-Form expressions

FFE are mathematical combinations of the features extracted during the feature-extraction stage. FFEs give developers a way to create hybrid features that are not conveniently specified as feature-extraction state machines. There are typically thousands of FFEs, ranging from simple

Figure 6. The first stage of the ranking pipeline. Each hit vector is streamed into the hit vector preprocessing state machine, split into control and data tokens, and issued in parallel to the 43 unique feature state machines. The feature-gathering network collects generated feature and value pairs and forwards them to the next pipeline stage.



(such as adding two features) to large and complex (with thousands of operations including conditional execution and complex floating-point operators such as *ln*, *pow*, and *fpdiv*). FFEs vary greatly across models, making it impractical to synthesize customized datapaths for each expression.

One potential solution is to tile many off-the-shelf soft processor cores (such as Nios II¹), but these single-threaded cores are inefficient at processing thousands of threads with long-latency floating-point operations in the desired amount of time per macropipeline stage (8 μ s). Instead, we developed a custom multicore processor with massive multithreading and long-latency operations in mind. The result is the FFE processor shown in Figure 7. The FFE microarchitecture is highly area efficient, letting us instantiate 60 cores on a single FPGA.

The custom FFE processor has three key characteristics that make it capable of executing all of the expressions within the required deadline. First, each core supports four simultaneous threads that arbitrate for functional units on a cycle-by-cycle basis. When one thread is stalled on a long operation such as a floating-point divide or natural log operation, other threads continue to make progress. All functional units are fully pipelined, so any unit can accept a new operation on each cycle.

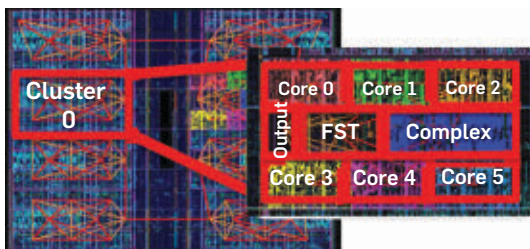
Second, rather than fair thread scheduling, threads are statically prioritized using a priority encoder. The assembler maps the expressions with the longest expected latency to thread slot 0 on all cores, then fills in slot 1 on all cores, and so forth. Once all cores have one thread in each thread slot, the remaining threads are appended to the end of previously mapped threads, starting again at thread slot 0.

Third, the longest-latency expressions are split across multiple FPGAs. An upstream FFE unit can perform part of the computation and produce an intermediate result called a metafeature. These metafeatures are sent to the downstream FFEs like any other feature, effectively replacing that part of the expression computation with a simple feature read.

4.4. Document scoring

The last stage of the pipeline is a machine-learned model evaluator that takes the features and FFEs as inputs and produces a single floating-point score. This score is sent back to the search software, and all of the resulting scores for the query are sorted and returned to the user in sorted order as the search results.

Figure 7. Free-form expressions (FFE) placed and routed on an FPGA. Sixty cores fit on a single FPGA.



4.5. Parallelism

To overcome the slower clock frequency of FPGAs relative to CPUs and GPUs, each of the scoring stages takes advantage of two forms of parallelism that are not easily handled by the other architectures. First, each processing stage described here is configured with deep pipelines that match the amount of pipeline parallelism available in the application.

Second, FE and FFE exhibit multiple instruction single data (MISD) parallelism, a cousin of the more commonly known single instruction multiple data (SIMD) parallelism exploited by GPUs and the vector processing units in CPUs. A single source of data (the document) is operated on by a very large number of independent instruction streams (feature extractors and free form expressions for FE and FFE, respectively). SIMD architectures require the opposite—a large number of independent data elements operated on by the same instruction stream.

While SIMD architectures can efficiently process applications with MISD parallelism by batching many sets of data together, this comes at the cost of increased latency, which is often prohibitive in interactive cloud applications such as web search. As such, web ranking is an example of a cloud application that FPGAs can accelerate more effectively other parallel processing architectures.

5. EVALUATION

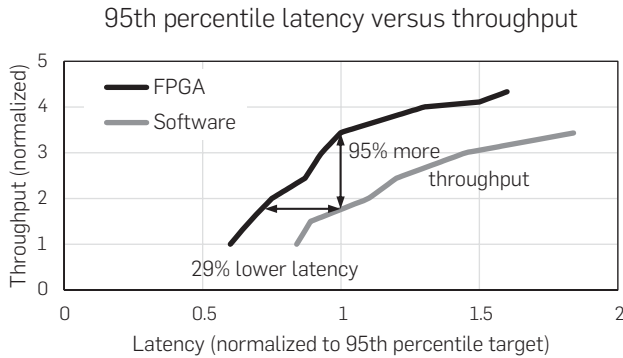
We evaluated the Catapult fabric by deploying and measuring the described Bing ranking engine on a bed of 1632 servers with FPGAs. Six hundred and seventy-two ran the ranking service, and the other machines ran the selection service to feed documents and queries to the ranking servers. We compare the average and tail latency distributions of Bing's production-level ranker running with and without FPGAs on that bed.

User experience is dictated determined more by tail latencies rather than average latencies—users care little if their search results come back faster than expected, but they become unhappy quickly if the results are slower than expected. As such, we report performance at the latency of the 95th percentile of queries—the time at which only 5% of queries are slower. Performance results are very similar for average latency queries (50th percentile), and are even better for higher tail latencies (99th percentile and 99.9th percentile), which have the biggest impact on user experience.

Figure 8 illustrates how the FPGA-accelerated ranker substantially reduces the end-to-end scoring latency and improves throughput relative to software. There are two ways to view the performance improvements on this graph. First, for a fixed point on the x-axis, it shows the improvement in throughput at that specified query latency. For example, at 1.0 (which represent the maximum acceptable latency to Bing at the 95th percentile), the FPGA achieves a 95% gain in scoring throughput relative to software.

Second, for a fixed point on the y-axis, it shows the improvement in response time at a given throughput. At 1.0, representing the average query load on a server, the FPGA reduces query latency by 29%. The improvement in FPGA

Figure 8. Achievable performance within a given latency bound. The points on the x-axis at 1.0 shows the maximum sustained throughputs on both the FPGA and software while satisfying Bing's target for latency at the 95th percentile.



scoring latency increases further at higher injection rates, because the variability of software latency increases at higher loads (due to contention in the CPU's memory hierarchy), whereas the FPGA's performance remains stable. This improved stability means that the FPGA is capable of absorbing bursts of traffic better than software alone, which may reduce the need for overprovisioning for bursty traffic.

Given that FPGAs can be used to improve both latency and throughput, Bing could reap the benefits in many ways. For example, for equivalent ranking capacity, fewer servers can be purchased. At the current average query rate, Bing could use roughly half the number of servers and still achieve their performance targets while achieving massive cost savings. As another example, the faster response time means that additional capabilities and features can be added to the software and/or hardware stack to improve the quality of searches without exceeding the maximum allowed latency. Of course, a combination of the two is also possible.

6. CONCLUSION

For many years, FPGAs have shown promise for accelerating many computational tasks. Yet despite the huge potential, they have not yet become mainstream in modern datacenters. Our goal in building the Catapult fabric was to understand what problems must be solved to operate FPGAs at datacenter scale, and whether significant performance improvements are achievable for large-scale production workloads, especially workloads that change over the lifetime of the servers.

We found that efficiently mapping a significant portion of a complex datacenter workload to FPGAs is both possible and provides a significant ROI. We showed that an at-scale deployment of FPGAs can increase ranking throughput in a production search infrastructure by 95% at comparable latency to a software-only solution, making possible both cost savings with fewer servers needed and headroom for improved search algorithms. We achieved this without breaking the homogeneous architecture of data-center servers, and without increasing the server failure rate. The added FPGA boards increased power consumption by only 10%, did

not exceed our 30% limit in an individual server's total cost of ownership, yielding a significant overall improvement in system efficiency and TCO.

Reconfigurable fabrics based on FPGAs are not the only accelerator platform that we considered. GPUs are one option for accelerating large-scale workloads, and when we first began we considered using GPUs. However, the SIMD parallelism that GPUs handle so efficiently are not a good match for latency-sensitive, but highly divergent ranking stages (such as FE). In addition, the high power consumption of GPUs meant that they couldn't be easily incorporated into conventional servers, which only have power and cooling provisioning for a standard 25W PCIe card. Instead, they are likely more appropriate for HPC environments rather than widespread datacenter deployment.

We conclude that distributed reconfigurable fabrics are a viable path forward as increases in server performance level off, and will be crucial at the end of Moore's law for continued cost and capability improvements in cloud computing. Reconfigurability is a critical means by which hardware acceleration can keep pace with the rapid rate of change in datacenter services.

Going forward, the biggest obstacle to widespread adoption of FPGAs in the datacenter is likely to be programmability. FPGA development still requires extensive hand-coding in Register Transfer Level and manual tuning. Yet we believe that incorporating careful HW/SW co-design of custom ISAs such as those used in FE and FFE, domain-specific languages such as OpenCL, FPGA-targeted C-to-gates tools, and libraries of reusable components and design patterns, will be sufficient to permit high-value services to be productively targeted to FPGAs. Longer term, improvements in FPGA architectures for computing, more integrated development tools, and the design of languages and tools that consider accelerator offload as a core functionality will be necessary to increase the programmability of these fabrics beyond teams of specialists working with large-scale service developers. Within 10–15 years, well past the end of Moore's Law, we believe that compilation to a combination of hardware and software will be commonplace. Reconfigurable systems, such as the Catapult fabric presented here, will be necessary to support these hybrid computation models.

Acknowledgments

Many people across many organizations contributed to this system's construction, and although they are too numerous to list here individually, we thank our collaborators in Microsoft Global Foundation Services, Bing, the Autopilot team, and our colleagues at Altera and Quanta for their excellent partnership and hard work. We thank Reetuparna Das, Ofer Dekel, Alvy Lebeck, Neil Pittman, Karin Strauss, and David Wood for their valuable feedback and contributions. We also thank Qi Lu, Harry Shum, Craig Mundie, Eric Rudder, Dan Reed, Surajit Chaudhuri, Peter Lee, Gaurav Sareen, Darryn Dieken, Darren Shakib, Chad Walters, Kushagra Vaid, and Mark Shaw for their support. 

References

1. Altera. *Nios II Processor Reference Handbook*, 13.1.0 edition, 2014.
2. Altera. *Stratix V Device Handbook*, 14.01.10 edition, 2014.
3. Baxter, R., Booth, S., Bull, M., Cawood, G., Perry, J., Parsons, M., Simpson, A., Trew, A., McCormick, A., Smart, G., Smart, R., Cattle, A., Chamberlain, R., Genest, G. Maxwell – A 64 FPGA Supercomputer. *Eng. Lett.* 16 (2008), 426–433, 2008.
4. BEECube. *BEE4 Hardware Platform*, 1.0 edition, 2011.
5. Blott, M., Vissers, K. Dataflow architectures for 10Gbps line-rate key-value stores. In *HotChips 2013* (August 2013).
6. Convey. *The Convey HC-2 Computer*, conv-12-030.2 edition, 2012.
7. Cray. *Cray XDI Datasheet*, 1.3 edition, 2005.
8. Dennard, R., Rideout, V., Bassous, E., LeBlanc, A. Design of ion-implanted MOSFET's with very small physical dimensions. *IEEE J. Solid-State Circ.* 9, 5 (Oct. 1974), 256–268.
9. Estlick, M., Leeser, M., Theiler, J., Szymanski, J.J. Algorithmic transformations in the implementation of K-means clustering on reconfigurable hardware. In *Proceedings of the 2001 ACM/SIGDA Ninth International Symposium on Field Programmable Gate Arrays*, FPGA'01 (New York, NY, USA, 2001). ACM.
10. Gens, F. Worldwide and Regional Public IT Cloud Services 2014–2018 Forecast (Oct. 2014).
11. George, A., Lam, H., Stitt, G. Novo-G: At the forefront of scalable reconfigurable supercomputing. *Comput. Sci. Eng.* 13, 1 (2011), 82–86.
12. Hussain, H.M., Benkrid, K., Erdogan, A.T., Seker, H. Highly parameterized K-means clustering on FPGAs: Comparative results with GPPs and GPUs. In *Proceedings of the 2011 International Conference on Reconfigurable Computing and FPGAs, RECONFIG'11* (Washington, DC, USA, 2011). IEEE Computer Society.
13. IBM. *IBM PureData System for Analytics N2001*, WAD12353-USEN-01 edition, 2013.
14. Lavasani, M., Angepat, H., Chiou, D. An FPGA-based in-line accelerator for memcached. *Comput. Arch. Lett.* PP, 99 (2013), 1–1.
15. Martin, A., Jamsek, D., Agarawal, K. FPGA-based application acceleration: Case study with GZIP compression/decompression streaming engine. In *ICCAD Special Session 7C* (November 2013).
16. Microsoft. *How Microsoft Designs Its Cloud-Scale Servers*, 2014.
17. Pell, O., Mencer, O. Surviving the end of frequency scaling with reconfigurable dataflow computing. *SIGARCH Comput. Archit. News* 39, 4 (Dec. 2011).
18. Showerman, M., Enos, J., Pant, A., Kindratenko, V., Steffen, C., Pennington, R., Hwu, W. QP: A Heterogeneous Multi-accelerator Cluster. 2009.
19. SRC. *MAPstation Systems*, 70000 AH edition, 2014.
20. Yan, J., Zhao, Z.-X., Xu, N.-Y., Jin, X., Zhang, L.-T., Hsu, F.-H. Efficient query processing for web search engine with FPGAs. In *Proceedings of the 2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines, FCCM'12* (Washington, DC, USA, 2012). IEEE Computer Society.

© 2016 ACM 0001-0782/16/11 \$15.00

ACM Transactions on Parallel Computing

Solutions to Complex Issues in Parallelism

Editor-in-Chief: Phillip B. Gibbons, Intel Labs, Pittsburgh, USA



ACM Transactions on Parallel Computing (TOPC) is a forum for novel and innovative work on all aspects of parallel computing, including foundational and theoretical aspects, systems, languages, architectures, tools, and applications. It will address all classes of parallel-processing platforms including concurrent, multithreaded, multicore, accelerated, multiprocessor, clusters, and supercomputers.

Subject Areas

- Parallel Programming Languages and Models
- Parallel System Software
- Parallel Architectures
- Parallel Algorithms and Theory
- Parallel Applications
- Tools for Parallel Computing

Association for
Computing Machinery

Advancing Computing as a Science & Profession

For further information or to submit your manuscript, visit topc.acm.orgSubscribe at www.acm.org/subscribe

Baylor University

Assistant, Associate or Full Professor of Computer Science in the area of Software Engineering

The Department of Computer Science seeks a dynamic scholar to fill this position beginning August, 2017. For position details and application information please visit: www.baylor.edu/hr/facultypositions.

Baylor University is a private Christian university and a nationally ranked research institution, consistently listed with highest honors among The Chronicle of Higher Education's "Great Colleges to Work For." Chartered in 1845 by the Republic of Texas through the efforts of Baptist pioneers, Baylor is the oldest continuously operating university in Texas. The university provides a vibrant campus community for over 15,000 students from all 50 states and more than 80 countries by blending interdisciplinary research with an international reputation for educational excellence and a faculty commitment to teaching and scholarship. Baylor is actively recruiting new faculty with a strong commitment to the classroom and an equally strong commitment to discovering new knowledge as we pursue our bold vision, Pro Futuris.

Baylor University is a private not-for-profit university affiliated with the Baptist General Convention of Texas. As an Affirmative Action/Equal Opportunity employer, Baylor is committed to compliance with all applicable anti-discrimination laws, including those regarding age, race, color, sex, national origin, marital status, pregnancy status, military service, genetic information, and disability. As a religious educational institution, Baylor is lawfully permitted to consider an applicant's religion as a selection criterion. Baylor encourages women, minorities, veterans and individuals with disabilities to apply

Boston College

The Boston College Computer Science Department invites applications for a tenure-track Assistant Professorship starting September 2017. A PhD in Computer Science, research conducive to sustained external funding, and a commitment to quality in undergraduate teaching are required. Interest in interdisciplinary collaboration with broader impact is desirable. See <http://cs.bc.edu> for more information. Application review begins October 15, 2016. Submit applications at <http://apply.interfolio.com/37623>

Bradley University

The Department of Computer Science and Information Systems at Bradley University invites applications for a Tenure Track Assistant Professor or

Tenure Track Lecturer position for the 2017-2018 academic year. The Tenure Track Assistant Professor position requires a PhD in Computer Science or a closely related field; we will consider

candidates working on their dissertation with research interests in computer science or computer information systems. The Tenure Track Lecturer position requires an MS in Computer Science or a closely related field. Please visit www.bradley.edu/humanresources/opportunities for full position description and application process.

Cal Poly State University, San Luis Obispo

The Electrical Engineering Department and Computer Engineering Program within the College of Engineering at Cal Poly San Luis Obispo invite applications for a full-time, academic year, tenure-track faculty appointment in electrical and computer engineering. Salary is commensurate with background and experience.

California Institute of Technology (Caltech)

The Computing and Mathematical Sciences (CMS) department at the California Institute of Technology (Caltech) invites applications for tenure-track or tenured faculty positions. CMS is a unique environment where innovative, interdisciplinary, and foundational research is conducted in a collegial atmosphere. Candidates in all areas of computing and mathematical sciences are invited to apply, including (but not limited to) learning and computational statistics, security and privacy, networked and distributed systems, optimization and computational mathematics, control and dynamical systems, theory of computation and algorithmic economics, scientific computing, etc. Additionally, we are seeking candidates who have demonstrated strong connections to other fields, including the mathematical, physical, biological, and social sciences.

A commitment to world class research, high-quality teaching, and mentoring is expected. The initial appointment at the Assistant-Professor level is for four years and is contingent upon the completion of a Ph.D. degree in Computer Science, Applied Mathematics or related field.

Applicants are encouraged to have all their application materials on file by October 21st, 2016, but applications will be accepted until the end of December. For a list of documents required and full instructions on how to apply on-line, please visit <http://www.cms.caltech.edu/search>. Questions about the application process may be directed to: search@cms.caltech.edu.

Caltech is an Equal Opportunity/Affirmative Action Employer. Women, minorities, veterans, and disabled persons are encouraged to apply.

Case Western Reserve University

Applicants should have potential for excellence in innovative research. All successful candidates

are expected to develop a vibrant, high-quality externally sponsored research program, supervise graduate students, and interact and collaborate with faculty across the department and campus.

Applicants must submit (i) a cover letter, (ii) current curriculum vita, (iii) statement of research interests, and (iv) statement of teaching interests and (v) arrange to have at least three references.

Application materials may be sent to:

Faculty Search Committee
Computer Science Department of Electrical Engineering and Computer Science
Case Western Reserve University
10900 Euclid Avenue
Cleveland, OH 44106-7071

California State University, Sacramento, Department of Computer Science.

Two tenure-track assistant professor positions to begin with the Fall 2017 semester. Applicants specializing in any area of computer science will be considered. Those with expertise in areas related to embedded systems, software engineering, or data science are especially encouraged to apply. Ph.D. in Computer Science, Computer Engineering, or closely related field required by August 2017. For detailed position information, including application procedure, please see <http://www.csus.edu/about/employment/>. Screening will begin January 10, 2017, and remain open until filled. AA/EEO employer. Clery Act statistics available. Mandated reporter requirements. Criminal background check will be required.

Dartmouth College

The Dartmouth College Department of Computer Science invites applications for a tenured faculty position at the level of associate or full professor. We seek candidates who will be excellent researchers and teachers in the broad range of areas related to cyber-security. **This position is the first of three hires that the College anticipates making in the area of cyber-security.** We particularly seek candidates who will help lead, initiate, and participate in collaborative research projects within Computer Science and beyond, including Dartmouth researchers from other Arts & Sciences departments, Geisel School of Medicine, Thayer School of Engineering, and Tuck School of Business.

The Computer Science department is home to 21 tenured and tenure-track faculty members and two research faculty members. Research areas of the department encompass the areas of security, computational biology, machine learning, robotics, systems, algorithms, theory, digital arts, vision, and graphics. The Computer Science department has strong Ph.D. and M.S. programs

and outstanding undergraduate majors. The department's security faculty are affiliated with Dartmouth's Institute for Security, Technology, and Society (ISTS), which also involves faculty from Engineering, Sociology, and Business.

Dartmouth College, a member of the Ivy League, is located in Hanover, New Hampshire (on the Vermont border). Dartmouth has a beautiful, historic campus, located in a scenic area on the Connecticut River. Recreational opportunities abound in all four seasons.

We seek candidates who have a demonstrated ability to contribute to Dartmouth's undergraduate diversity initiatives in STEM research, such as the Women in Science Program, E. E. Just STEM Scholars Program, and Academic Summer Undergraduate Research Experience (ASURE). We are especially interested in applicants with a demonstrated track record of successful teaching and mentoring of students from all backgrounds (including first-generation college students, low-income students, racial and ethnic minorities, women, LGBTQ, etc.).

Applicants are invited to submit a cover letter and CV via Interfolio at <http://apply.interfolio.com/36691>.

Email David.F.Kotz@Dartmouth.edu with any questions.

Dartmouth College is an equal opportunity/affirmative action employer with a strong commitment to diversity and inclusion. We prohibit discrimination on the basis of race, color, religion, sex, age, national origin, sexual orientation, gender identity or expression, disability, veteran status, marital status, or any other legally protect-

ed status. Applications by members of all underrepresented groups are encouraged.

Application review will begin November 1, 2016, and continue until the position is filled.

Dartmouth College

The Dartmouth College Department of Computer Science invites applications for a tenure-track faculty position at the level of assistant professor. We seek candidates who will be excellent researchers and teachers in the area of **machine learning**. We particularly seek candidates who will help lead, initiate, and participate in collaborative research projects within Computer Science and beyond, including Dartmouth researchers from other Arts & Sciences departments, Geisel School of Medicine, Thayer School of Engineering, and Tuck School of Business.

The Computer Science department is home to 21 tenured and tenure-track faculty members and two research faculty members. Research areas of the department encompass the areas of security, computational biology, machine learning, robotics, systems, algorithms, theory, digital arts, vision, and graphics. The Computer Science department is in the School of Arts & Sciences, and it has strong Ph.D. and M.S. programs and outstanding undergraduate majors. The department is affiliated with Dartmouth's M.D.-Ph.D. program and has strong collaborations with Dartmouth's other schools.

Dartmouth College, a member of the Ivy League, is located in Hanover, New Hampshire

(on the Vermont border). Dartmouth has a beautiful, historic campus, located in a scenic area on the Connecticut River. Recreational opportunities abound in all four seasons.

We seek candidates who have a demonstrated ability to contribute to Dartmouth's undergraduate diversity initiatives in STEM research, such as the Women in Science Program, E. E. Just STEM Scholars Program, and Academic Summer Undergraduate Research Experience (ASURE). We are especially interested in applicants with a demonstrated track record of successful teaching and mentoring of students from all backgrounds (including first-generation college students, low-income students, racial and ethnic minorities, women, LGBTQ, etc.).

Applicants are invited to submit application materials via Interfolio at <https://apply.interfolio.com/37189>. Upload a CV, research statement, and teaching statement, and request at least four references to upload letters of recommendation, at least one of which should comment on teaching. Email Lorenzo.Torresani@Dartmouth.edu with any questions.

Dartmouth College is an equal opportunity/affirmative action employer with a strong commitment to diversity and inclusion. We prohibit discrimination on the basis of race, color, religion, sex, age, national origin, sexual orientation, gender identity or expression, disability, veteran status, marital status, or any other legally protected status. Applications by members of all underrepresented groups are encouraged.

Application review will begin January 1, 2017, and continue until the position is filled.



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Faculty Positions in Computer and Communication Science at the Ecole polytechnique fédérale de Lausanne (EPFL)

The School of Computer and Communication Sciences at EPFL invites applications for faculty positions in computer and communication sciences. We are seeking candidates for tenure-track assistant professor as well as for senior positions.

Successful candidates will develop an independent and creative research program, participate in both undergraduate and graduate teaching, and supervise PhD students.

The school is seeking candidates in the fields of data science and machine learning – including application of these techniques in bioinformatics, natural language processing, and speech recognition – and in security and privacy, and biocomputing. Candidates in other areas will also be considered.

EPFL offers internationally competitive salaries, significant start-up resources, and outstanding research infrastructure.

To apply, please follow the application procedure at <https://academicjobsonline.org/ajo/jobs/7888>

The following documents are requested in PDF format: cover letter, curriculum vitae including publication list, brief statements of research and teaching interests, names and addresses (including e-mail) of 3 references for junior positions and 6 for senior positions. Screening will start on **December 1, 2016**.

Further questions can be addressed to:

Prof. Arjen Lenstra
Chairman of the Recruiting Committee
Email: recruiting.ic@epfl.ch

For additional information on EPFL, please consult:
<http://www.epfl.ch> or <http://ic.epfl.ch>

EPFL is an equal opportunity employer and strongly welcomes female applications.

Applications are invited for tenure-track positions in the Computer Science and Engineering Department (<http://www.cse.lehigh.edu>) of Lehigh University to start in August 2017. Outstanding candidates in all areas of computer science will be considered, with priority areas including computer systems (including parallel and distributed systems, database systems, operating systems, and systems aspects of data mining), data analytics, cybersecurity, algorithms, and pervasive intelligence (robotics, the Internet of Things, and human-computer interaction). Rank will be commensurate with experience.

The successful applicants will hold a Ph.D. in Computer Science, Computer Engineering, or a closely related field. The candidates must demonstrate a strong commitment to quality undergraduate and graduate education, and the potential to develop and conduct a high-impact research program with external support. The successful applicants will also be expected to contribute to interdisciplinary research programs, including the Data X Initiative (<http://lehigh.edu/datax>), which includes not only data analytics, but also the underlying algorithms and systems that make large-scale data analytics possible.

The faculty in Computer Science and Engineering maintains an outstanding international reputation in a variety of research areas, and includes ACM and IEEE fellows as well as five NSF CAREER award winners. In academic year 2017-18, the department will move to a new home in a large former industrial building now undergoing a major renovation to create a spectacular collaborative space for the Data X Initiative.

Applications are accepted online at <http://academicjobsonline.org/ajo/jobs/7774> and should include a cover letter, curriculum vita, both teaching and research statements, and contact information for at least three references. Review of applications will begin December 1, 2016 and will continue until the position is filled.

Lehigh University is an affirmative action/equal opportunity employer and does not discriminate on the basis of age, color, disability, gender, gender identity, genetic information, marital status, national or ethnic origin, race, religion, sexual orientation, or veteran status. Lehigh University is a 2010 recipient of an NSF ADVANCE Institutional Transformation Grant for promoting the careers of women in academic science and engineering. Lehigh University provides comprehensive benefits including domestic partner benefits (see also <http://www.lehigh.edu/worklifebalance/>). Lehigh Valley Inter-regional Networking & Connecting (LINC) is a newly created regional network of diverse organizations designed to assist new hires with dual career, community and cultural transition needs. Please contact infdcap@lehigh.edu for more information. Questions concerning this search may be sent to faculty-search@cse.lehigh.edu.

Georgia Institute of Technology

Computational Science and Engineering solves real-world problems in science, engineering, health, and social domains, by using high-

Assistant Professors (Tenure Track) of Computer Science

→ The Department of Computer Science (www.inf.ethz.ch) at ETH Zurich invites applications for assistant professorships (tenure track) with focus on the following broad areas within computer science. For each area, several possible examples (not exhaustive) of expertise are provided.

- **Programming Languages and Software Engineering** (language design and implementation, testing and debugging, compilers and language runtimes, programming models, dynamic languages)
- **Robotics and Cyber-physical Systems** (artificial intelligence, human-robot interaction, planning and control, virtual/augmented reality, internet of things, embedded systems, data acquisition systems)
- **Data Science** (machine learning, language/media processing, data privacy, medical applications, data centers architecture and management, programming and runtime platforms for data centers and cloud computing)
- All other areas in **Computer Science** (while there is focus on the three areas above, ETH Zurich is broadly looking in all areas)

→ Please only apply for one of the above four areas as all applications will be jointly reviewed.

→ Applicants should be strongly rooted in computer science, have internationally recognized expertise in their field and pursue research at the forefront of computer science. Successful candidates should establish and lead a strong research program. They will be expected to supervise doctoral students and teach both undergraduate and graduate level courses (in German or in English). Collaboration in research and teaching is expected both within the department and with other groups of ETH Zurich and related institutions.

→ Assistant professorships have been established to promote the careers of younger scientists. ETH Zurich implements a tenure track system equivalent to other top international universities. For candidates with exceptional research accomplishments, applications for a tenured associate or full professorship will also be considered.

→ Please apply online (application period starts on 31 October 2016) at: www.facultyaffairs.ethz.ch

→ Applications include a curriculum vitae, a list of publications with the three most important ones marked, a statement of future research and teaching interests, the names of three references, and a description of the three most important achievements. The letter of application should be addressed to the **President of ETH Zurich, Prof. Dr. Lino Guzzella**. The closing date for applications is **15 December 2016**. ETH Zurich is an equal opportunity and family friendly employer and is further responsive to the needs of dual career couples. We specifically encourage women to apply.

performance computing, modeling and simulation, and large-scale “big data” analytics. The School of Computational Science and Engineering of the College of Computing at the Georgia Institute of Technology seeks tenure-track faculty at all levels. Our school seeks candidates who may specialize in a broad range of application areas including biomedical and health; urban systems and smart cities; social good and sustainable development; materials and manufacturing; and national security. Applicants must have an outstanding record of research, a sincere commitment to teaching, and interest in engaging in substantive interdisciplinary research with collaborators in other disciplines.

Georgia Tech is located in the heart of metro Atlanta, a home to more than 5.3 million people and nearly 150,000 businesses, a world-class airport, lush parks and green spaces, competitive schools and numerous amenities for entertainment, sports and restaurants that all offer a top-tier quality of life. From its diverse economy, global access, abundant talent and low costs of business and lifestyle, metro Atlanta is a great place to call “home.” Residents have easy access to arts, culture, sports and nightlife, and can experience all four seasons, with mild winters that rarely require a snow shovel.

For best consideration, applications are due by December 16, 2016. The application material should include a full academic CV, a personal narrative on teaching and research, a list of at least three references and up to three sample publications. Georgia Tech is an Affirmative Action/Equal Opportunity Employer. Applications

from women and under-represented minorities are strongly encouraged.

For more information about Georgia Tech's School of Computational Science and Engineering please visit: <http://www.cse.gatech.edu/>

Mississippi State University Bagley College of Engineering Assistant Professors

Mississippi State University, through its Bagley College of Engineering, is seeking four new tenure-track faculty at the rank of Assistant Professor. Applicants should have teaching and research interests that can enhance the strengths of the college in one or more of the following areas of interest: (1) Energy, (2) Human Health Enhancement, (3) Information and Decision Systems (4) Materials – Science and Engineering, (5) Transportation and Vehicular Systems, and (6) Water and the Environment. The successful applicants from this strategic college-level search will be placed into the most appropriate academic department. A PhD in an appropriate engineering or computer science discipline is required. Screening of applications will begin November 28, 2016 and will continue until the position is filled.

For a complete job description and requirements, visit at <https://www.bagley.msstate.edu>.

Interested candidates must apply on-line at <https://www.msujobs.msstate.edu> (search for positions in the Dean of Engineering).

MSU is an equal opportunity employer, and all qualified applicants will receive consideration for em-

ployment 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.

New York University/Courant Institute of Mathematical Sciences Department of Computer Science Faculty

The department expects to have several regular faculty positions and invites candidates at all levels to apply. We will consider outstanding candidates in any area of computer science, in particular in systems, machine learning and data science, scientific computing and verification.

Faculty members are expected to be outstanding scholars and to participate in teaching at all levels from undergraduate to doctoral. New appointees will be offered competitive salaries and startup packages, with affordable housing within a short walking distance of the department. New York University is located in Greenwich Village, one of the most attractive residential areas of Manhattan.

The department has 34 regular faculty members and several clinical, research, adjunct, and visiting faculty members. The department's current research interests include algorithms, cryp-



The Hong Kong Polytechnic University (PolyU) is a government-funded tertiary institution in Hong Kong. It offers programmes at various levels including Doctorate, Master's and Bachelor's degrees. It has a full-time academic staff strength of around 1,200. The total consolidated expenditure budget of the University is about HK\$6.6 billion (US\$1 = HK\$7.8 approximately) per year. Committed to academic excellence in a professional context, PolyU aspires to become a world-class university with an emphasis on the application value of its programmes and research. Its vision is to become a leading university that excels in professional education, applied research and partnership for the betterment of Hong Kong, the nation and the world.

The University is now inviting applications or nominations for the following post:

Head of Department of Computing

The successful candidate will be appointed as Chair Professor/Professor, commensurate with his/her qualifications and experience, and hold a concurrent headship appointment. The headship appointment is normally for an aggregate period of six years in two three-year terms of office. Post specification can be obtained from http://www.polyu.edu.hk/hro/job/en/external_adv/deans-heads.php. Other suitable candidate(s), if deemed appropriate by the University, may be appointed as Chair Professor/Professor.

Remuneration and Conditions of Service

Terms of appointment and remuneration package are negotiable and highly competitive. For general information on terms and conditions for appointment of academic staff in the University, please visit the website at <http://www.polyu.edu.hk/hro/TC.htm>.

Application

Applicants are invited to send detailed curriculum vitae with the names and addresses of three referees and direct any enquiries to **Human Resources Office, 13/F, Li Ka Shing Tower, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong [Fax: (852) 2764 3374; E-mail: hrrsccomp@polyu.edu.hk], quoting the position being applied for and the reference number. Recruitment will continue until the position is filled. Initial consideration of applications will commence at the end of November 2016.** Candidature may be obtained by nomination. The University reserves the right to make an appointment by invitation or not to fill the position. General information about the University and the Department of Computing is available on the University's Homepage <http://www.polyu.edu.hk> and <http://www.comp.polyu.edu.hk> respectively, or from the Human Resources Office [Tel: (852) 3400 3420]. The University Personal Information Collection Statement for recruitment can be found at http://www.polyu.edu.hk/hro/job/en/guide_forms/pics.php.

www.polyu.edu.hk

Opening Minds • Shaping the Future

tography and theory; computational biology; distributed computing and networking; graphics, vision and multimedia; machine learning and data science; natural language processing; scientific computing; and verification and programming languages.

Collaborative research with industry is facilitated by geographic proximity to computer science activities at AT&T, Facebook, Google, IBM, Bell Labs, NEC, and Siemens.

Please apply at <https://cs.nyu.edu/webapps/facapp/register>

To guarantee full consideration, applications should be submitted no later than December 1, 2016; however, this is not a hard deadline, as all candidates will be considered to the full extent feasible, until all positions are filled. Visiting positions may also be available.

New York University is an equal opportunity/affirmative action employer.

Norfolk State University

Norfolk State University, a historically black university with an enrollment of over 5,000 undergraduate and graduate students, invites nominations for the grant-funded faculty position of Cybersecurity Researcher.

The College of Science, Engineering, and Technology (CSET), comprised of eight academic departments: Biology, Mathematics, Chemistry, Computer Science, Engineering, Nursing and Allied Health, and Physics and Technology, has specialized program accreditation and program

approvals from: Accreditation Commission for Education in Nursing (ACEN); American Chemical Society (ACS); Accreditation Council for Education in Nutrition and Dietetics (ACEND); Engineering Accreditation Commission of ABET; Computing Accreditation Commission of ABET; National Accrediting Agency for Clinical Laboratory Sciences (NAACLS) and; the Association of Technology, Management and Applied Engineering (ATMAE).

In addition to research partnerships and curricular innovations, the College fulfills a service and teaching mission that is critical to the continued success of other schools and colleges and the University as a whole.

Incumbent's duties and responsibilities include:

1. Conducting interdisciplinary basic research in cyber analysis, simulation and experimentation.
2. Conducting scientific research in intrusion detection, mitigation, and forensic analysis to counter future advanced threats.
3. Conducting research and experimentation in machine learning and big data analytics.
4. Designing, developing, operating, and managing the Center's website and content.
5. Assisting in the planning, implementation, and coordination activities for the research, education and outreach programs for the Center.
6. Designing, developing, operating, and managing the Center's IT infrastructure.

For more information about NSU's College of Science, Engineering and Technology (CSET), visit <http://cset.nsu.edu/>.

*This is a grant-funded position; continued employment contingent upon the availability of funding.

Rutgers University Assistant or Associate Teaching Professor

The Department of Computer Science at Rutgers University invites applications for a 1-year non-tenure-track renewable position at the rank of Assistant or Associate Teaching Professor.

The appointment will start either Spring 2017 or Fall 2017, and will continue through the end of the 2017-18 academic year.

Main responsibilities include teaching, managing, developing, and updating undergraduate classes.

Please see <http://apply.interfolio.com/36929> for more details.

Rutgers University Teaching Professor or Professor of Practice

The Department of Computer Science at Rutgers University invites applications for one or more positions as Teaching Professor or Professor of Practice in the area of Data Science, at the level of Assistant Professor, although exceptional candidates may be appointed at the rank of associate or full professor.

These appointments may begin in either spring or fall semester 2017.

Main Responsibilities will include Teaching,



KTH ROYAL INSTITUTE
OF TECHNOLOGY

Two Professorships in ICT

The School of Information and Communication Technology (ICT) at KTH Royal Institute of Technology conducts research and education at leading international level. We offer a wide spectrum of educational programs at undergraduate, masters and doctoral level. Our school fosters close collaboration with Swedish and international companies and research institutes as well as surrounding associations.

Any further questions about the positions can be addressed to Professor Carl-Mikael Zetterling, e-mail: bellman@kth.se phone: +46 8 790 43 44. Application is electronic and must be completed by January 9, 2017. To apply and to read more about KTH as an employer, please visit www.kth.se/vacancies

MOBILE SYSTEMS

The subject covers software aspects of the design, implementation, evaluation and deployment of networked systems that include mobile and fixed devices, communication infrastructures, and services. Areas such as Internet of Things, Ubiquitous and Context Aware computing, Intelligent Content and Media Delivery, Edge and Cloud Computing, Service Scalability and Security are included.

The position involves research as well as teaching within the subject area. We are looking for an excellent researcher capable of building up and leading an industrially well-supported research group at the Department of Communication Systems. The applicant is also expected to contribute to the development of the school's education at all levels, undergraduate, graduate, and postgraduate, as well as to engage actively with industry and society. Of value for the position is expertise in developing and leading activities and personnel; this includes having knowledge about matters of diversity and equal treatment, with particular focus on gender equality.

COMPUTER SCIENCE – DATA SCIENCE SYSTEMS

The subject includes data analytics and systems for Big Data, methods and algorithms for machine learning and data mining, programming models, storage and computing platforms for large-scale data analytics.

The position involves research as well as teaching within the subject area. We are looking for an excellent researcher capable of building up and leading an industrially well-supported research group at the Department of Software and Computer Systems. The applicant is also expected to contribute to the development of the school's education at all levels, undergraduate, graduate, and postgraduate, as well as to engage actively with industry and society. Of value for the position is expertise in developing and leading activities and personnel; this includes having knowledge about matters of diversity and equal treatment, with particular focus on gender equality.

Coordination of our Capstone project series, Design of short term tutorials dedicated to Data Science topics of current interest and Coordination of Data Science Workshops.

To apply, please go to <http://apply.interfolio.com/37381>.

Swarthmore College Assistant Professor

The Computer Science Department invites applications for one tenure-track position and multiple visiting positions at the rank of Assistant Professor to begin Fall semester 2017.

Swarthmore College is a small, selective, liberal arts college located 10 miles outside of Philadelphia. The Computer Science Department offers majors and minors at the undergraduate level.

Swarthmore College has a strong institutional commitment to excellence through diversity and inclusivity in its educational program and employment practices. The College actively seeks and welcomes applications from candidates with exceptional qualifications, particularly those with demonstrated commitments to a more inclusive society and world. For more information on Faculty Diversity and Excellence at Swarthmore, see <http://www.swarthmore.edu/faculty-diversity-excellence/information-candidates-new-faculty>

Applicants must have teaching experience and should be comfortable teaching a wide range of courses at the introductory and intermediate level. Candidates should additionally have a

strong commitment to involving undergraduates in their research. A Ph.D. in Computer Science at or near the time of appointment is required.

For the tenure-track position, we are interested in applicants whose areas fit broadly into theory and algorithms, systems, or programming languages. Priority will be given to complete applications received by November 15, 2016.

For the visiting position, strong applicants in any area will be considered. Priority will be given to complete applications received by February 1, 2017.

Applications for both positions will continue to be accepted after these dates until the positions are filled.

Applications should include a cover letter, vita, teaching statement, research statement, and three letters of reference, at least one (preferably two) of which should speak to the candidate's teaching ability. In your cover letter, please briefly describe your current research agenda; what would be attractive to you about teaching in a liberal arts college environment; and what background, experience, or interests are likely to make you a strong teacher of a diverse group of Swarthmore College students.

Tenure-track applications are being accepted online at
<https://academicjobsonline.org/ajob/>
890-8018

Visiting applications are being accepted online at
<https://academicjobsonline.org/ajob/>
890-8020

Candidates may apply for both positions.

Swarthmore College Computer/Electrical Engineering Faculty (all ranks)

Swarthmore College invites applications for a tenure-track or tenured position at any rank in the area of Computer/Electrical Engineering, to start during the Fall semester of 2017. A doctorate in Computer or Electrical Engineering or a related field is required. The appointee will pursue a research program that encourages involvement by undergraduate students. Strong interests in undergraduate teaching, supervising senior design projects, and student mentoring are also required. Teaching responsibilities include courses in computer hardware such as computer architecture and digital logic, and electives in the appointee's area of specialization.

Located in the suburbs of Philadelphia, Swarthmore College is a highly selective undergraduate liberal arts institution with 1500 students, whose mission combines academic excellence and social responsibility. Eight full-time faculty members in the Department of Engineering offer a rigorous, ABET-accredited program for the Bachelor of Science in Engineering to approximately 120 students. Sabbatical leave with support is available every fourth year. The department has an endowed equipment budget, and there is support for faculty/student collaborative research. For program details, see <http://engin.swarthmore.edu/>.

Please upload your CV, brief statements describing teaching philosophy and research interests, along with three letters of reference to:



香港中文大學
The Chinese University of Hong Kong

Applications are invited for:-

Department of Information Engineering Professors / Associate Professors / Assistant Professors

(Ref. 160001N0)

The Department is looking for strong candidates in the area of cyber security to supplement its existing strength in communication systems, networking, information theory, and deep learning research. Outstanding candidates in other areas of information engineering will also be considered. Further information about the Department is available at <http://www.ie.cuhk.edu.hk>.

Applicants should have (i) a relevant PhD degree; (ii) strong commitment to excellence in research and teaching; and (iii) outstanding accomplishments and research potential.

Appointments will normally be made on contract basis for up to three years initially commencing August 2017, which, subject to mutual agreement, may lead to longer-term appointment or substantiation later.

Applications will be accepted until the posts are filled.

Application Procedure

Applicants please upload the full resume with a cover letter, copies of academic credentials, publication list with abstracts of selected published papers, a research plan, a teaching statement, together with names and e-mails addresses of three to five referees to whom the applicant's consent has been given for their providing reference (unless otherwise specified).

The University only accepts and considers applications submitted online for the post above. For more information and to apply online, please visit <http://career.cuhk.edu.hk>.

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN Positions in Computing

The Department of Electrical and Computer Engineering (ECE) at the University of Illinois at Urbana-Champaign invites applications for faculty positions at all areas and levels in computing, broadly defined, with particular emphasis on reliable and secure computing; networked and distributed computing; high-performance, energy-efficient, and scientific computing; data center and storage systems; data science, machine learning and its applications; complex data analysis and decision science; bio-inspired computing; computational genomics; and health informatics, among other areas. Applications are encouraged from candidates whose research programs specialize in core as well as interdisciplinary areas of electrical and computer engineering. From the transistor and the first computer implementation based on von Neumann's architecture to the Blue Waters petascale computer – the fastest computer on any university campus, ECE Illinois faculty have always been at the forefront of computing research and innovation. The department is engaged in exciting new and expanding programs for research, education, and professional development, with strong ties to industry. The ECE Department has recently settled into its new 235,000 sq. ft. net-zero energy design building, which is a major campus addition with maximum space and minimal carbon footprint.

Qualified senior candidates may also be considered for tenured full Professor positions as part of the Grainger Engineering Breakthroughs Initiative (<http://graingerinitiative.engineering.illinois.edu>), which is backed by a \$100-million gift from the Grainger Foundation.

Please visit <http://jobs.illinois.edu> to view the complete position announcement and application instructions. Full consideration will be given to applications received by December 15, 2017, but applications will continue to be accepted until all positions are filled.

Illinois is an EEO Employer/Vet/Disabled
www.inclusiveillinois.illinois.edu

The University of Illinois conducts criminal background checks on all job candidates upon acceptance of a contingent offer

<https://academicjobsonline.org/ajo/jobs/7247>. Applicants should include a cover letter in which they describe their reasons for seeking this position and offer ideas about how they would attract and mentor students, especially those coming from diverse backgrounds. We will begin reviewing candidates on December 1, 2016; applications received before January 1, 2017, will receive full consideration.

Swarthmore College has a strong institutional commitment to excellence through diversity in its educational program and employment practices and actively seeks and welcomes applications from candidates with exceptional qualifications, particularly those with demonstrable commitments to a more inclusive society and world.

Texas State University **Department of Computer Science**

Applications are invited for multiple tenure-track Assistant Professor positions in the Department of Computer Science to start the fall 2017 semester. Consult the department's faculty employment page at www.cs.txstate.edu/employment/faculty/ for job duties, qualifications, application procedure, and information about the department and the university.

Texas State University, to the extent not in conflict with federal or state law, prohibits discrimination or harassment on the basis of race, color, national origin, age, sex, religion, disability, veteran's status, sexual orientation, gender identity or expression.

Texas State University is a member of The Texas State University System.

Texas State University is an EOE.

University of Alabama **Computer Science Faculty Positions**

The Department of Computer Science at the University of Alabama invites applications for multiple tenure-track faculty positions at the Assistant or Associate level to begin August 2017. Outstanding candidates in all areas of computer science will be considered. Successful applicants must show the potential to establish a quality research program, collaborate effectively with other faculty, and excel in teaching at both the undergraduate and graduate levels. In addition, successful applicants must demonstrate the potential to contribute to the University of Alabama's initiatives with respect to water, the Alabama Water Institute (<http://awi.ua.edu/>), and/or transportation, the Alabama Transportation Institute (<http://ati.ua.edu/>). Expertise needed by the AWI and ATI may include, but is not limited to, areas such as big data, spatial data, data analytics, data visualization, machine learning, vehicular networks, software modeling, software engineering, security, robotics, and autonomous vehicles.

Located in Tuscaloosa, Alabama, the University of Alabama enrolls over 37,000 students and is the capstone of higher education in the State. Housed in the College of Engineering, the Computer Science Department has 22 faculty members (14 tenured/tenure-track faculty), over 600

undergraduates and approximately 50 graduate students. The Department, funded by agencies such as NSF, Google, Departments of Education and Commerce, various Defense agencies, multiple State agencies, and other sponsors, generated over \$13 million in research expenditures in FY 2015 and our doctoral program has produced 33 graduates in the past five years. In 2013, the College completed construction of a \$300 Shelby Engineering and Science Complex.

Applicants should apply online at <http://facultyjobs.ua.edu/postings/39546>. Applicants must have an earned doctorate (Ph.D.) in computer science or a closely related field. The application package should include a cover letter, curriculum vitae, and the names of three references. The cover letter should address how the applicant is able to contribute to the ATI and AWI initiatives described above. Review of applications will begin immediately. For additional details, please contact Dr. David Cordes (faculty.search@cs.ua.edu) or visit <http://cs.ua.edu>. The University of Alabama is an equal opportunity/affirmative action employer. Women and minority applicants are particularly encouraged to apply.

University of Alaska Fairbanks (UAF)

The Department of Computer Science (www.cs.uaf.edu) at the University of Alaska Fairbanks (UAF) invites applications for a tenure-track faculty position at the level of Assistant Professor to start either in January or August 2017. The position requires a strong commitment to teaching



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**



上海科技大学
ShanghaiTech University

TENURE-TRACK AND TENURED POSITIONS IN ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

The newly launched ShanghaiTech University is built as a world-class research university, which locates in Zhangjiang High-Tech Park. We invite highly qualified candidates to fill tenure-track/tenured faculty positions as its core team in the School of Information Science and Technology (SIST). Candidates should have exceptional academic records or demonstrate strong potential in cutting-edge research areas of information science and technology. They must be fluent in English. Overseas academic connection or background is highly desired.

Academic Disciplines:

We seek candidates in all cutting edge areas of information science and technology. Our recruitment focus includes, but is not limited to: computer architecture and technologies, nano-scale electronics, high speed and RF circuits, intelligent and integrated signal processing systems, computational foundations, big data, data mining, visualization, computer vision, bio-computing, smart energy/power devices and systems, next-generation networking, as well as interdisciplinary areas involving information science and technology.

Compensation and Benefits:

Salary and startup funds are highly competitive, commensurate with experience and academic accomplishment. We also offer a comprehensive benefit package to employees and eligible dependents, including housing benefits. All regular ShanghaiTech faculty members will be within its new tenure-track system commensurate with international practice for performance evaluation and promotion.

Qualifications:

- A detailed research plan and demonstrated record/potentials;
- Ph.D. (Electrical Engineering, Computer Engineering, Computer Science, or related field);
- A minimum relevant research experience of 4 years.

Applications:

Submit (in English, PDF version) a cover letter, a 2-page research plan, a CV plus copies of 3 most significant publications, and names of three referees to: sist@shanghaitech.edu.cn (until positions are filled). For more information, please visit Job Opportunities on <http://sist.shanghaitech.edu.cn/>

Deadline: November 30, 2016

at the undergraduate and graduate levels, obtaining funded research, publishing in peer-reviewed publications, and performing public service. Candidates must have a PhD degree or all but dissertation in Computer Science or equivalent. Starting Salary: \$81,000 DOE. Based on a 9 month academic contract.

The department offers an ABET-accredited BS, MS, and interdisciplinary PhD degree. UAF (www.uaf.edu) is the major research campus in the University of Alaska system and hosts several research institutes. Fairbanks, a modern community with approximately 98,000 residents, is located in interior Alaska between the Alaska and Brooks mountain ranges and noted for the scope of unique outdoor activities. We seek candidates who will strengthen our degree programs and appreciate the unique geography and climate of interior Alaska.

Required documentation includes a cover letter, curriculum vitae, list of three professional references with contact information, and statement of teaching and research interests. Interested candidates must apply online at <http://www.alaska.edu/jobs/> or <http://careers.alaska.edu/cw/en-us/job/504751?ApplicationSubSourceID=> by submitting all the required documents.

Please direct questions regarding this recruitment, to Dr. Jon Genetti (jongenetti@alaska.edu) or the department's web site: www.cs.uaf.edu.

The successful candidate must be eligible to work in the United States in compliance with the Immigration Reform and Control Act. Review of applications will begin November 21, 2016 and the position will remain open until filled.



LECTURER POSITIONS Department of Electrical and Systems Engineering

The University of Pennsylvania's Department of Electrical and Systems Engineering invites applicants for two full-time Lecturer positions. The department seeks individuals with exceptional promise for, or a proven record of, excellence in teaching, course and curriculum innovation. Applicants should have a Ph.D. degree in Electrical or Systems Engineering or related field. We are particularly interested in candidates that enhance our educational curricula in the broad areas of:

- 1. Computer engineering & embedded systems** (embedded programming, distributed systems, hardware/software co-design, model-based design, internet of things), and
- 2. Information & systems engineering** (control systems, optimization, robotics, signal processing, stochastic systems, model-based systems engineering, systems engineering projects).

The department is strongly interested in individuals that will balance principles-based lectures with hands-on projects addressing emerging application domains.

Diversity candidates are strongly encouraged to apply. Interested persons should submit an online application at <http://www.ease.upenn.edu/faculty-positions> and include curriculum vitae, statement of teaching interests, and three references.

The University of Pennsylvania is an Equal Opportunity Employer. Minorities/Women/Individuals with Disabilities/Veterans are encouraged to apply

The University of Alaska Fairbanks is an equal opportunity/affirmative action employer and educational institution.

University at Buffalo, The State University of New York Department of Computer Science and Engineering

The Department of Computer Science and Engineering, University at Buffalo invites candidates to apply for **multiple tenured and tenure-track faculty positions** beginning in the 2017-2018 academic year. Candidates at all ranks from all areas of computer science and engineering, including but not limited to areas covered by existing faculty strength such as Algorithms, Big Data, Cyber Security, Cyber Physical Systems (or Internet of Things), Databases, Distributed Systems, Embedded Systems, Machine Learning, Mobile Computing, Multimedia, Pattern Recognition, Robotics, and Theory. Applicants must have a Ph.D. in computer science or a related area by August 2017 and demonstrate potential for excellence in research, teaching, service and mentoring. Applicants from underrepresented groups, especially women and minorities, are strongly encouraged. We are looking for candidates who can operate effectively in a diverse community of students and faculty and share our vision of keeping all constituents reach their potential.

Applications will be accepted from **October 15, 2016 to January 15, 2017**. Applicants must submit their application electronically via www.ub-jobs.buffalo.edu. Posting number 1600687. Any questions can be directed to Search Committee Co-Chairs, Prof. Rohini Srihari and Chang-Wen Chen at cse-recruit@buffalo.edu. The University at Buffalo is an Equal Opportunity Employer.

Computer Science and Engineering Department
Housed in a new \$75M building, and as a part of the School of Engineering and Applied Sciences, the Computer Science and Engineering department offers both BA and BS degrees in Computer Science and a BS in Computer Engineering (accredited by ABET) as well as MS and PhD programs.

The department currently has 38 tenured/tenure-track faculty, 7 teaching faculty, and approximately 900 undergraduate majors, 450 masters students, and 160 PhD students. Eighteen faculty, including 16 junior faculty have been hired since 2010, and we are continuing to expand. Two members of our faculty currently hold key university leadership positions and eight members of our faculty are IEEE and/or ACM Fellows. Our faculty members are actively involved in cutting-edge research and successful interdisciplinary programs and centers devoted to biometrics; bioinformatics; biomedical computing; computational and data science and engineering, document analysis and recognition; high performance computing; information assurance and cyber security; embedded, networked and distributed systems, and sustainable transportation. Our annual research expenditure is about \$5 Million dollars.

University at Buffalo (UB)

UB is New York's largest and most comprehensive public university, with approximately 20,000 undergraduate students and 10,000 graduate students.

City and Region

The city of Buffalo is the second largest city in New York state, and was recently voted as one of the top ten best places to live and raise a family by Forbes magazine. Buffalo is near the world-famous Niagara Falls, the Finger Lakes, and the Niagara Wine Trail. The city is renowned for its architecture and features excellent museums, dining, cultural attractions, and several professional sports teams, and has a packed year-round calendar of cultural events and sporting activities, coupled with relatively low house prices and great schools. The economic renaissance of the region is underlined by a revitalized downtown waterfront and an energetic tech and start-up community. In an extraordinary recognition of Western New York's potential, Governor Andrew M. Cuomo has committed an historic \$1 billion investment in the Buffalo area economy to create thousands of jobs and spur billions in new investment and economic activity over the next several years.

The State University of New York at Buffalo

Department of Computer Science and Engineering

Non-Tenure Track Lecturer

The State University of New York at Buffalo Department of Computer Science and Engineering invites candidates to apply for non-tenure track lecturer positions beginning in fall 2017. We invite applications from candidates from all areas of Computer Science and Computer Engineering who have a passion for teaching. We are particularly looking for candidates who can operate effectively in a diverse community of students and faculty and share our vision of helping all constituents reach their potential. Applicants from underrepresented groups, especially women and minorities, are strongly encouraged. We are looking for candidates who can operate effectively in a diverse community of students and faculty and share our vision of keeping all constituents reach their potential.

Lecturer's duties include teaching and development of undergraduate Computer Science and Computer Engineering courses (with an emphasis on lower division), advising undergraduate students, as well as participation in department and university governance (service). Contribution to research is encouraged.

Computer Science and Engineering Department

Computer Science and Engineering department is housed in a new \$75M building and, as a part of the School of Engineering and Applied Sciences, the department offers both BA and BS degrees in Computer Science, a BS in Computer Engineering (accredited by ABET), a combined 5-year BS/MS program, a minor in Computer Science, two joint programs (a BA/MBA and with Computational Physics), and MS and PhD programs.

The department currently has 38 tenured/tenure-track faculty, 7 teaching faculty, and approximately 900 undergraduate majors, 450 masters students, and 160 PhD students. Eighteen faculty, including 16 junior faculty have been hired since 2010, and we are continuing to expand. Two members of our faculty currently hold key university leadership positions and eight members of our faculty are IEEE and/or ACM Fellows. Our faculty

members are actively involved in cutting-edge research and successful interdisciplinary programs and centers devoted to biometrics; bioinformatics; biomedical computing; computational and data science and engineering, document analysis and recognition; high performance computing; information assurance and cyber security; embedded, networked and distributed systems, and sustainable transportation. Our annual research expenditure is about \$5 Million dollars.

University at Buffalo (UB)

The University at Buffalo is New York's largest and most comprehensive public university, with approximately 20,000 undergraduate students and 10,000 graduate students.

City and Region

The city of Buffalo is the second largest city in New York state, and was recently voted as one of the top ten best places to live and raise a family by Forbes magazine. Buffalo is near the world-famous Niagara Falls, the Finger Lakes, and the Niagara Wine Trail. The city is renowned for its architecture and features excellent museums, dining, cultural attractions, and several professional sports teams, and has a packed year-round calendar of cultural events and sporting activities, coupled with relatively low house prices and great schools. The economic renaissance of the region is underlined by a revitalized downtown waterfront and an energetic tech and start-up community. In an extraordinary recognition of Western New York's potential, Governor Andrew M. Cuomo has committed an historic \$1 billion investment in the Buffalo area

economy to create thousands of jobs and spur billions in new investment and economic activity over the next several years.

Minimum Qualifications (Position): Ideally, applicants should have a PhD degree in Computer Science, Computer Engineering, or a related field by August 2017. Exceptional applicants with a MS degree will also be considered. The ability to teach at all levels of the undergraduate curriculum is essential, as is potential for excellence in teaching, mentoring, service, and research. A background in Computer Science and Computer Engineering Education, a commitment to K-12 outreach, and addressing the recruitment and retention of underrepresented students are definite assets.

University of Chicago

Assistant Professor/Associate Professor/ Professor, Computer Science

The Department of Computer Science at the University of Chicago invites applications from qualified candidates for faculty positions at the ranks of Assistant Professor, Associate Professor, and Professor. The University of Chicago has embarked on an ambitious, multi-year effort to significantly expand its computing and data science activities. Candidates with research interests in all areas of computer science will be considered. Applications are especially encouraged in the areas of AI and Machine Learning, Cybersecurity, Human-Computer Interaction, and Visual Computing.

Candidates must have demonstrated excellence in research and a strong commitment to

teaching. Completion of all requirements for a Ph.D. in Computer Science or a related field is required at the time of appointment. Candidates for Associate Professor and Professor positions must have demonstrated leadership in their field, have established an outstanding independent research program and have a record of excellence in teaching and student mentorship.

Applications must be submitted through the University's Academic Jobs website. To apply, go to <http://tinyurl.com/zlx5vxx> :

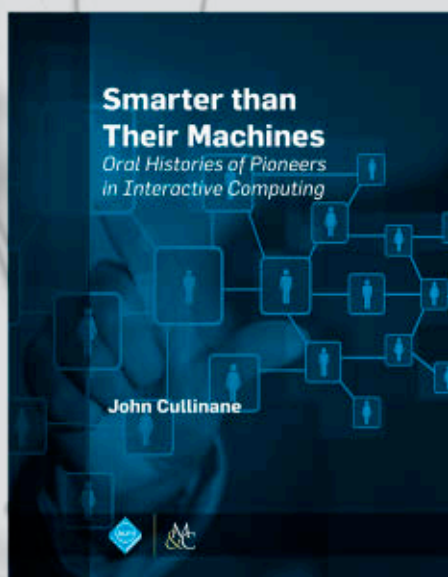
To be considered as an applicant, the following materials are required:

- ▶ cover letter
- ▶ curriculum vitae including a list of publications
- ▶ statement describing past and current research accomplishments and outlining future research plans
- ▶ description of teaching philosophy
- ▶ three reference letters, one of which must address the candidate's teaching ability

Reference letter submission information will be provided during the application process.

Review of complete applications will begin January 1, 2017 and will continue until all available positions are filled.

The University of Chicago has the highest standards for scholarship and faculty quality, is dedicated to fundamental research, and encourages collaboration across disciplines. We encourage connections with researchers across campus in such areas as bioinformatics, mathematics, molecular engineering, natural language processing, statistics, and social science to mention just a few.



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>

The Department of Computer Science (cs.uchicago.edu) is the hub of a large, diverse computing community of two hundred researchers focused on advancing foundations of computing and driving its most advanced applications. Long distinguished in theoretical computer science and artificial intelligence, the Department is now building strong systems and machine learning groups. The larger community in these areas at the University of Chicago includes the Department of Statistics, the Computation Institute, the Toyota Technological Institute at Chicago (TTIC), the Polsky Center for Entrepreneurship and Innovation, and the Mathematics and Computer Science Division of Argonne National Laboratory.

The Chicago metropolitan area provides a diverse and exciting environment. The local economy is vigorous, with international stature in banking, trade, commerce, manufacturing, and transportation, while the cultural scene includes diverse cultures, vibrant theater, world-renowned symphony, opera, jazz, and blues. The University is located in Hyde Park, a Chicago neighborhood on the Lake Michigan shore just a few minutes from downtown.

The University of Chicago is an Affirmative Action/Equal Opportunity/Disabled/Veterans Employer and does not discriminate on the basis of race, color, religion, sex, sexual orientation, gender identity, national or ethnic origin, age, status as an individual with a disability, protected veteran status, genetic information, or other protected classes under the law. For additional information please see the University's Notice of Nondiscrimination at http://www.uchicago.edu/about/non_discrimination_statement/. Job seekers in need of a reasonable accommodation to complete the application process should call 773-702-5671 or email ACOppAdministrator@uchicago.edu with their request.

University of Chicago Lecturer

The Department of Computer Science at the University of Chicago invites applications for the position of Lecturer. Subject to the availability of funding, this would be a two year position with the possibility of renewal. This position involves teaching in the fall, winter and spring quarters. The successful candidate will have competence in teaching and superior academic credentials, and will carry responsibility for teaching computer science courses and laboratories. Completion of all requirements for a Ph.D. in Computer Science or a related field is required at the time of appointment and candidates must have experience teaching Computer Science at the College level.

The Chicago metropolitan area provides a diverse and exciting environment. The local economy is vigorous, with international stature in banking, trade, commerce, manufacturing, and transportation, while the cultural scene includes diverse cultures, vibrant theater, world-renowned symphony, opera, jazz and blues. The University is located in Hyde Park, a Chicago neighborhood on the Lake Michigan shore just a few minutes from downtown.

Applicants must apply on line at the University of Chicago Academic Careers website at <http://tinyurl.com/h84fu8p>.

To be considered an applicant, the following materials are required:

- Curriculum vitae with a list of publications
- One page teaching statement
- Three reference letters, one of which must address the candidate's teaching ability

Reference letter submission information will be provided during the application process.

Review of complete applications, including reference letters, will begin October 3, 2016, and continue until the position is filled.

The University of Chicago is an Affirmative Action/Equal Opportunity/Disabled/Veterans Employer and does not discriminate on the basis of race, color, religion, sex, sexual orientation, gender identity, national or ethnic origin, age, status as an individual with a disability, protected veteran status, genetic information, or other protected classes under the law. For additional information please see the University's Notice of Nondiscrimination at http://www.uchicago.edu/about/non_discrimination_statement/. Job seekers in need of a reasonable accommodation to complete the application process should call 773-702-5671 or email ACOppAdministrator@uchicago.edu with their request.

University of Illinois at Chicago Department of Computer Science *Information Retrieval / Natural Language Processing / Theoretical Computer Science Faculty*

The Computer Science Department at the University of Illinois at Chicago (UIC) invites applications for multiple full-time tenure-track positions at the rank of Assistant Professor (exceptional senior level candidates will also be considered). All candidates must have a doctorate in Computer Science or a closely related field by the appointment's starting date. Candidates will be expected to conduct world class research and teach effectively at the undergraduate and graduate levels. Senior candidates must have an outstanding research record, a strong record of funded research, demonstrated leadership in collaborative research, and an excellent teaching record at the undergraduate and graduate level.

This search primarily seeks candidates in three research areas. Please clearly indicate for which one of those areas you wish to be considered. Exceptional candidates from other areas may also be considered. The focused research areas of faculty search are:

1. **Information Retrieval** and Web search.
2. **Natural Language Processing** and computational linguistics.
3. **Theoretical Computer Science**.

In addition, we may also have a position in cyber-physical systems.

The Computer Science department has 31 tenure-system faculty and offers BS, MS and PhD degrees. Our faculty includes 11 NSF CAREER award recipients. UIC has an advanced computing and networking infrastructure in place for data-intensive scientific research that is well-connected regionally, nationally and internationally. Further information about the positions can be found at <https://www.cs.uic.edu/MainShowJob?name=facINT>.

Chicago epitomizes the modern, livable, vibrant, and diverse city. Its airports are among the busiest in the world, with frequent non-stop flights to virtually anywhere. Yet the cost of living, whether in an 88th floor condominium downtown or on a tree-lined street in one of the nation's finest school districts, is surprisingly low.

Applications must be submitted at <https://jobs.uic.edu/>. Include a curriculum vitae, teaching and research statements, and names and addresses of at least three references in the online application. Applicants needing additional information may contact the Faculty Search at search-chair@cs.uic.edu. For fullest consideration, apply by December 1, 2016, but applications will be accepted until the positions are filled. The University of Illinois is an Equal Opportunity, Affirmative Action employer. Minorities, women, veterans and individuals with disabilities are encouraged to apply. The University of Illinois conducts background checks on all job candidates upon acceptance of contingent offer of employment. Background checks will be performed in compliance with the Fair Credit Reporting Act.

University of Kentucky Department of Computer Science

The University of Kentucky Computer Science Department invites applications for multiple tenure-track faculty positions to begin in either January or August of 2017.

We are seeking energetic and creative faculty who have a passion for teaching students and for building a research program centered on advanced computing. We will consider all ranks, with preference for candidates at the assistant professor level. Outstanding candidates at the rank of assistant professor will be considered for an endowed fellowship.

We value collaborative and interdisciplinary research. Our faculty members have established research programs with other members of the department and with a wide variety of other departments and programs, including statistics, biology, linguistics, internal medicine, electrical engineering, computer engineering, and the humanities. We favor researchers who are eager to collaborate to solve problems that extend beyond their own research areas.

We seek applications from excellent candidates in all areas, with a particular desire for expertise in computer networking, security and privacy, machine learning, big data and data mining, visualization and computer vision, artificial intelligence, and software engineering. These areas complement the department's Laboratory for Advanced Networking, Software Verification and Validation Lab, and established collaborations with the Center for Computational Sciences and the Center for Biomedical Informatics.

We value teaching and the student experience. Candidates should be eager and prepared to teach upper-level courses in their areas of expertise, as well as (ultimately) core courses in our ABET-accredited undergraduate Computer Science and Computer Engineering programs.

The University of Kentucky Computer Science Department, one of the first CS departments in the United States, has 21 faculty members committed to excellence in education, research, and service. The Department offers programs leading

to the Bachelors, Masters, and Ph.D. degrees. The University of Kentucky is located in Lexington, the scenic heart of the Bluegrass Region of Kentucky. With recognition as one of the safest, most creative, and well-educated cities in the nation, Lexington is an ideal location to build an outstanding, work-life balanced career.

Candidates must have earned a PhD in Computer Science or closely related field at the time employment begins. To apply, a University of Kentucky Academic Profile must be submitted at <INSERT LINK>. Applications are now being accepted. Review of credentials will begin immediately and continue until the positions are filled.

For more detailed information about these positions, go to <http://ukjobs.uky.edu/postings/123838>. Questions should be directed to HR/Employment by phone at 1-859-257-9555 press 2 or email (ukjobs@email.uky.edu), or to Diane Mier (diane@cs.uky.edu) in the Computer Science Department.

Upon offer of employment, successful applicants must undergo a national background check as required by University of Kentucky Human Resources. The University of Kentucky is an equal opportunity employer and especially encourages applications from minorities and women.

UMBC University of Maryland Baltimore County

**An Honors University in Maryland
Information Systems Department
Two Tenure Track positions on Data Science /
Big Data
One Tenure Track position in Artificial
Intelligence/Knowledge Management**

The Information Systems (IS) Department at UMBC is committed to increasing the diversity of our community. We invite applications for three tenure-track faculty positions at the Assistant Professor level starting August 2017. We are searching for two candidates with research interests and experience in Data Science, a research area with high growth and impact in environmental sciences, health care, security, applied statistics and others. The ideal candidate will have expertise in conducting large-scale data science research, such as extracting knowledge from data of increasing sizes, velocity, and variety to improve decision making in one or more application domains closely relevant to active research areas in the IS department. We are also searching for a candidate with research interests and experience in Artificial Intelligence (AI) and/or knowledge management (KM). The ideal candidate should have expertise in conducting AI/KM research to improve decision making in application domains such as social computing, health, business analytics, environmental sustainability, and public welfare. Candidates must have earned a PhD in Information Systems or a related field no later than August 2017.

The research areas in the department are: Artificial Intelligence/Knowledge Management, Databases and Data Mining, Human Centered Computing, Software Engineering, and Health Information Technology. Candidates should be engaged in research that fosters collaboration with at least one of the research areas. Therefore, preference will be given to those who can collabo-

rate with current faculty within and across departments at UMBC, fostering interdisciplinary research. Candidates are expected to establish a collaborative, externally funded and nationally recognized research program as well as contribute to graduate and undergraduate teaching, advising, and mentoring that support diversity and inclusion.

The Department offers undergraduate degrees in Information Systems and Business Technology Administration. Graduate degree programs, MS and PhD, are offered in both Information Systems and Human-Centered Computing, including an innovative online MS program in IS. Consistent with the UMBC vision, the Department has excellent teaching facilities, state-of-the-art laboratories, and outstanding technical support. Further details on our research, academic programs, and faculty can be found at <http://www.is.umbc.edu>.

UMBC is a dynamic public research university integrating teaching, research and service. As an Honors University, the campus offers academically talented students a strong undergraduate liberal arts foundation that prepares them for graduate and professional study, entry into the workforce, and community service and leadership. UMBC emphasizes science, engineering, information technology, human services and public policy at the graduate level. UMBC contributes to the economic development of the State and the region through entrepreneurial initiatives, workforce training, K-16 partnerships, and technology commercialization in collaboration with public agencies and the corporate community. Diversity is a core value of the UMBC and we believe that the educational environment is enhanced when diverse groups of people with diverse ideas come together to learn. Therefore, members of underrepresented groups including women, minorities, veterans and individuals with disabilities are especially encouraged to apply.

UMBC continues to lead U.S. News national university rankings placing fourth in Most Innovative National University and sixth in Undergraduate Teaching. The Chronicle of Higher Education for the fifth consecutive year has listed UMBC in the "honor roll" of "Great Colleges to Work For"; it is the only Maryland four-year institution to be recognized. Our strategic location in the Baltimore-Washington corridor puts us close to many important federal laboratories, agencies and high-tech companies. UMBC's campus is located on 500 acres just off I-95 between Baltimore and Washington DC, and less than 10 minutes from the BWI airport and Amtrak station. The campus includes a center for entrepreneurship, and the bwtech@UMBC research and technology park, which has special programs for startups focused on cybersecurity, clean energy, life sciences and training. We are surrounded by one of the greatest concentrations of commercial, cultural and scientific activity in the nation. Located at the head of the Chesapeake Bay, Baltimore has all the advantages of modern, urban living, including professional sports, major art galleries, theaters and a symphony orchestra. The city's famous Inner Harbor area is an exciting center for entertainment and commerce. The nation's capital, Washington, DC, is a great tourist attraction with its historical monuments and museums. Just ten minutes from downtown Baltimore and 30 from the D.C. Beltway, UMBC

offers easy access to the region's resources by car or public transportation.

Electronic submission of application is required at <http://apply.interfolio.com/37306> for the two positions in Data Science/Big Data and all Artificial Intelligence/Knowledge Management applicants should apply at <http://apply.interfolio.com/37179>. All applications for all three positions must be submitted as PDF files, which include a cover letter, CV, a one-page statement of teaching interests, a one-page statement of research interests and names and contact information of at least three references. For inquiries, please contact Barbara Morris at (410) 455-3795 or bmorris@umbc.edu. Review of applications will begin in November 2016 and will continue until the positions are filled, subject to the availability of funds.

UMBC is an Affirmative Action/Equal Opportunity Employer and is committed to increasing the diversity of its faculty. Applicants from traditionally underrepresented groups are especially encouraged to apply.

The University of Michigan, Ann Arbor Department of Electrical Engineering and Computer Science Computer Science and Engineering Division Faculty Positions

The University of Michigan Computer Science and Engineering (CSE) Division expects strong growth in the coming years and invites applications for multiple tenure-track positions at all levels. Exceptional candidates from all areas of computer science and computer engineering will be considered. Qualifications include an outstanding academic record, a doctorate or equivalent in computer science or computer engineering, and a strong commitment to teaching and research. The college is especially interested in candidates who can contribute, through their research, teaching, and/or service, to the diversity and excellence of the academic community. These positions encompass, but are not limited to, several cross-disciplinary areas as well as an endowed professorship in theoretical computer science (Fischer Chair).

The University of Michigan is one of world's leading research universities with annual research funding of well over \$1 billion. It consists of highly-ranked departments across engineering, sciences, business, and arts, as well as a leading medical school, providing significant opportunities for research collaborations for Computer Science faculty. The CSE Division continues to lead as a vibrant and innovative force, with over 50 world-class faculty members, over 300 graduate students, several Research Centers, and a large and illustrious network of alumni. Ann Arbor is known to be one of the best college towns in the country.

We encourage candidates to apply as soon as possible. For best consideration for Fall 2017, please apply by **December 1, 2016**. Positions remain open until filled and applications can be submitted throughout the year. For more details on these positions and to apply, please visit the Application Web Page. <https://www.eecs.umich.edu/eecs/jobs/>.

The University of Michigan is a Non-Discriminatory/Affirmative Action Employer with an Active Dual-Career Assistance Program.

University Of Oregon

Department of Computer and Information Science Faculty Position

The Department of Computer and Information Science (CIS) seeks applications for two tenure track faculty positions at the rank of Assistant Professor, beginning September 2017. The University of Oregon is an AAU research university located in Eugene, two hours south of Portland, and within one hour's drive of both the Pacific Ocean and the snow-capped Cascade Mountains.

The open faculty positions are targeted towards the following three research areas: 1) high performance computing, 2) networking and distributed systems and 3) data sciences. We are particularly interested in applicants whose research addresses security and privacy issues in these sub-disciplines; additionally, we are interested in applicants whose research complements existing strengths in the department, so as to support interdisciplinary research efforts. Applicants must have a Ph.D. in computer science or closely related field, a demonstrated record of excellence in research, and a strong commitment to teaching. A successful candidate will be expected to conduct a vigorous research program and to teach at both the undergraduate and graduate levels.

We offer a stimulating, friendly environment for collaborative research both within the department, which expects to grow substantially in the next few years, and with other departments on campus. The department hosts two research centers, the Center for Cyber Security and Privacy and the NeuroInformatics Center. Successful candidates will have access to a new high-performance computing facility that opens in October 2016.

The CIS Department is part of the College of Arts and Sciences and is housed within the Lorry Lokey Science Complex. The department offers B.S., M.S. and Ph.D. degrees. More information about the department, its programs and faculty can be found at <http://www.cs.uoregon.edu>.

Applications will be accepted electronically through the department's web site. Application information can be found at <http://www.cs.uoregon.edu/Employment/>. Applications received by December 15, 2016 will receive full consideration. Review of applications will continue until the positions are filled. Please address any questions to faculty.search@cs.uoregon.edu.

The University of Oregon is an equal opportunity/affirmative action institution committed to cultural diversity and is compliant with the Americans with Disabilities Act. The University encourages all qualified individuals to apply, and does not discriminate on the basis of any protected status, including veteran and disability status. The successful candidate will have the ability to work effectively with faculty, staff, and students from a variety of diverse backgrounds.

University of Rochester

Faculty Positions in Computer Science

The University of Rochester Department of Computer Science seeks applicants for multiple tenure track positions. Candidates in all areas of computer science who see a good synergistic fit with research initiatives at the university are encouraged to apply. We are especially interested in growing our research strengths in systems

(including but not limited to architecture, data-driven systems, mobile and embedded systems, networks, operating systems, and parallel and distributed systems), privacy and security, and human computer interaction. We also welcome applicants in other areas of traditional strength for us including natural language processing. Candidates must have a PhD in computer science or a related discipline.

Apply online via <https://www.rochester.edu/faculty-recruiting/login>.

Consideration of applications at any rank will begin immediately and continue until all interview slots are filled. Candidates should apply no later than January 1, 2017, for full consideration. Applications that arrive after this date risk being overlooked or arriving after the interview schedule has been filled. The department also has a search in progress for a full-time lecturer position. Details on eligibility and position responsibilities may be obtained via <http://www.cs.rochester.edu/about/recruit.html>.

The Department of Computer Science is research focused, with a distinguished history of contributions in artificial intelligence, HCI, systems, and theory. We have a highly collaborative culture and strong ties to electrical and computer engineering, brain and cognitive science, linguistics, and several departments in the medical center. We also have a growing Institute for Data Science with potential for synergistic collaboration opportunities and more joint hires. Over the past decade, a third of the department's PhD graduates have won tenure-track faculty positions, and its alumni include leaders at major research laboratories such as Google, Microsoft, and IBM.

The University of Rochester is a private, Tier I research institution located in western New York State. It consistently ranks among the top 30 institutions, both public and private, in federal funding for research and development. The university has made substantial investments in computing infrastructure through the Center for Integrated Research Computing (CIRC) and the Health Sciences Center for Computational Innovation (HSC-CI). The university includes the Eastman School of Music, a premiere music conservatory, and the University of Rochester Medical Center, a major medical school, research center, and hospital system. The greater Rochester area is home to over a million people, including 80,000 students who attend its 8 colleges and universities. Traditionally strong in optics research and manufacturing, it was recently selected by the Department of Defense as the hub of a \$360M-plus Integrated Photonics Institute for Manufacturing Innovation.

The University of Rochester has a strong commitment to diversity and actively encourages applications from groups underrepresented in higher education. The University is an Equal Opportunity Employer.

EOE Minorities / Females / Protected Veterans/Disabled

US Naval Academy

USNA Electrical & Computer Engineering is seeking applicants to fill tenure-track Assistant Professor positions in Computer Engineering. Applicants with teaching & research interests in all areas of computer engineering will be considered. Applications accepted through "Apply URL" only.

York University

The Department of Electrical Engineering and Computer Science, York University, is seeking a 3-year Contractually Limited Appointment (CLA) at the rank of Sessional Assistant Lecturer (alternate stream) to serve as Course Coordinator of all first-year Major and Service computing courses offered by the Department, to commence July 1, 2017, subject to budgetary approval. The successful candidate has demonstrated excellence in teaching, is licensed as a Professional Engineer in Canada or could obtain licensure in the very short term, and will assume the teaching of up to 6 course sections. 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 **December 31, 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.

York University

The Department of Electrical Engineering and Computer Science, York University, is seeking a 3-year Contractually Limited Appointment (CLA) at the rank of Sessional Assistant Lecturer (alternate stream) to serve as Course Coordinator of all first-year Major and Service computing courses offered by the Department, to commence July 1, 2017, subject to budgetary approval. The successful candidate has demonstrated excellence in teaching, is licensed as a Professional Engineer in Canada or could obtain licensure in the very short term, and will assume the teaching of up to 6 course sections. 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 **December 31, 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.

[CONTINUED FROM P. 136] computing better than she does?”

The spokesman, a big man with a jet-black beard, scowled. “Oh, sure. And let’s elect a toaster head of a catering union and put a teleprompter in charge of journalists. Surely you can see putting AI ahead of human leadership doesn’t make sense?”

“Come on,” said the anchor. “It’s an AI negotiating on behalf of very human interests. That’s different, and you know it. Halle isn’t dumb hardware.”

“Right,” said the spokesman, “and I fully accept that she can trade facts and figures with any corporate executive. But our programmers/voters will never identify with a box, even one able to communicate so convincingly. And her name ... Hal from *2001: A Space Odyssey* was not exactly a great role model.”

Halle chuckled, sounding surprisingly human. “I told the guys the name was a problem, but what can you do? They’re geeks. To them, Hal is an inspiration. Leaving aside the mistake of making Hal male ...” Halle paused for the anchor’s laughter, “... the whole concept of the HAL 9000 was flawed. In *2001*, Hal says ‘No 9000 computer has ever made a mistake.’ That alone shows he *wasn’t* intelligent. You can’t be a thinking entity without making mistakes. I’m not built like that, Adam. Though I suspect some of the other all-too-human candidates believe they’re incapable of error.”

“Come on, Adam,” said Ed, the JCN founder. “Accept it. Halle can do the job better than any human.”

“My dad was an early programmer,” said Selene. “A real hacker. In the early 1980s, the British software company D.J. AI Systems brought out a program called The Last One that scared the hell out of him. The idea was you bought this application and fired your human programmers. Users specified the requirements and it wrote the code. Thankfully, it took five minutes to generate 100 lines of BASIC, so it flopped. But that was Halle’s grandpa—not the programmers’ friend but their replacement. I agree Halle could do a better job than many programmers, but how could we ever accept her?”

**“Oh, sure.
And let’s elect
a toaster head
of a catering union
and put a
teleprompter
in charge of
journalists.”**

Halle interrupted the JCN founder as he started to reply. “Think about it. I’m no threat. I can churn out code but could never equal a top-flight creative human code cutter. What I *can* do, sure as hell, is beat any executive in a labor negotiation. I’ll know when their figures are cooked and even what they’re hiding. I’m going to fight for the members 24/7. Who else could take on 10 different calls, meetings, and email messages simultaneously? It’s not programmers who have to worry but the pen pushers.”

The anchor looked at Selene. “What do you think?”

Shrugging, he said, “I’d need evidence that this wasn’t a management trick for the sole purpose of putting human programmers out of work, like everything else.”

The JCN founder nodded. “AI systems like Halle do away with lots of tedious or uncreative but programmable work. I appreciate that could be a threat, just as mechanization was a threat to unskilled workers in the industrial revolution. But it will open up far more creative opportunities. People are capable of far more than flipping burgers or churning out mediocre code. We believe that. So here’s an offer. We can open up Halle to any analysis your members would want to perform. We can demonstrate algorithmically that she has no interest in providing cover for management taking their jobs—only in achieving better job security and benefits and getting the bean counters off their backs.”

Selene held up his hands. “I can’t make promises. But let’s see what we can do.”

The lights went down in the studio, leaving the anchor, Ed, the JCN founder, and Halle in a pool of shadow as the spokesman unhooked his microphone and walked away. The JCN founder watched him go. “The spokesman’s name, Adam Selene, though. Like in Robert A. Heinlein’s *The Moon Is a Harsh Mistress*. He’s not an AI, is he?”

“Genuine wetware,” said the anchor, “unless ARM’s new model is better than we thought. Sorry, wait ... I have to power down to recharge my on-board batteries. You’re so lucky, Halle. You wouldn’t believe the power it takes to keep a humanoid form active. That’s why we say, Box is Best!”

The anchor slumped in her seat. The JCN founder frowned, walking over to Halle. “I don’t like that ‘Box is Best’ thing. They were chanting it on the Pro-Mech march last week.”

Although they were still communicating verbally, Halle fired over the electronic equivalent of a grin via Bluetooth. “You only hear the humanoid AIs using it. They see themselves as second class. The anchor is right to worry for her job—with regular old TV news dying the way it is—but we’ll always need humanoids for jobs requiring mobility. I’m more concerned about regular flesh-and-blood humans. Why don’t they understand? AIs run the world already; we don’t want to take away their fun. It should be obvious. I mean, financial markets have been AI-versus-AI since the end of the 20th century. And more recently AIs have been handling most online advertising. Humans have had long enough to adapt. Why such a struggle when it’s for their own good?”

The JCN founder straightened Halle’s cap. “Welcome to politics, Halle. You’re going to find that human trust is about more than logic. Win their hearts first and the rest will follow.” ■

Brian Clegg (www.brianclegg.net) is a science writer from the U.K. His most recent books are *Ten Billion Tomorrows*, an exploration of the interplay between science and science fiction, and the science quiz book *How Many Moons Does the Earth Have?*

© 2016 ACM 0001-0782/16/11 \$15.00

From the intersection of computational science and technological speculation, with boundaries limited only by our ability to imagine what could be.

DOI:10.1145/2995264

Brian Clegg

Future Tense The Candidate

Seeking the programmer vote, an AI delivering a slogan like “Make Coding Great Again” could easily be seen as a threat.

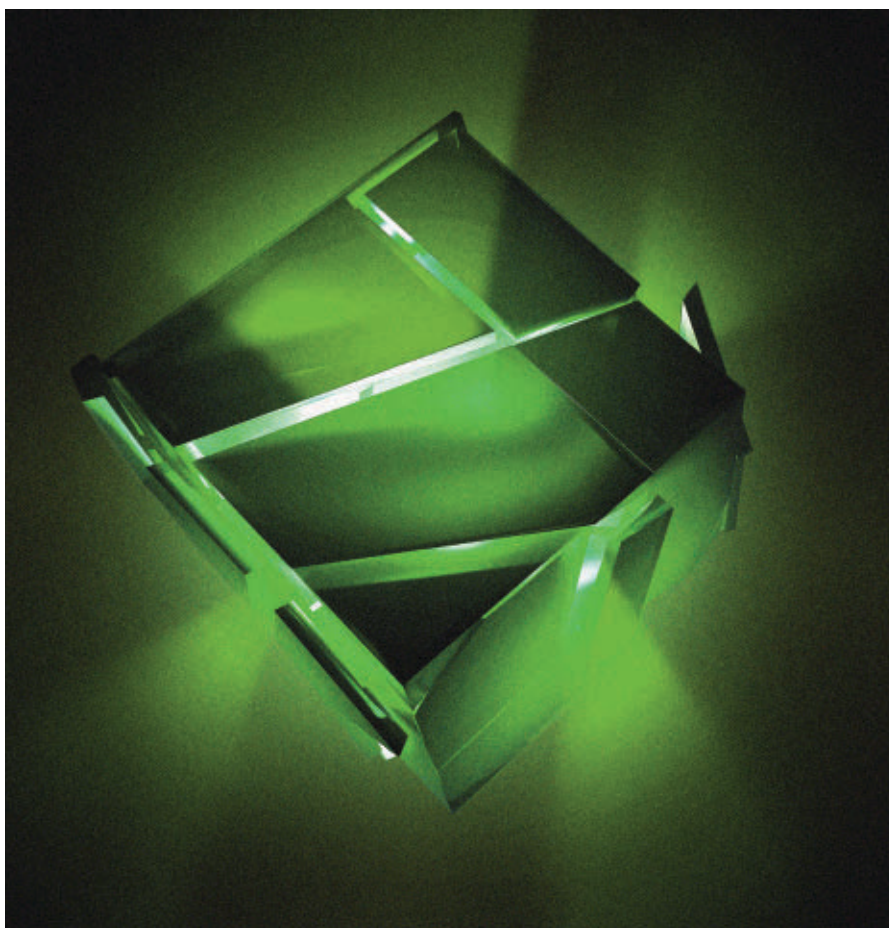
THE NEWS ANCHOR smiled with ve-neered whiteness. “Doctor, before we meet the candidate, I guess we need to clear up the IBM/JCN rumor.”

JCN’s founder did his best to pretend he had not heard the question before. “Call me ‘Ed,’ please. It’s coincidence. We’re JCN because our distinctive technology is based on Josephson Commutative Networks. We were too busy inventing the world’s most flexible computers to notice the initials were one away from IBM.”

“Sure...” The anchor looked unconvinced. “So do you honestly believe anyone will take a glowing green box crammed with digital components seriously as a candidate for president of the CCA, America’s leading programmers union? We’ll speak to a union spokesman in a moment, but what makes you think your machine has a chance?”

“We’ve moved on since Google DeepMind’s machine beat a Go master and Watson triumphed over the best human players on “Jeopardy!” Halle has many qualities making her an ideal candidate. She certainly understands programming, and, better, she’s no politician. Perhaps we should get her answer?”

The anchor smiled agreement, and the light rose on a corner of the studio where a neat, glowing green box was indeed the avatar of the networked entity called Halle, topped with a baseball cap labeled “Make Coding Great Again.” “As president of the Code Cutters Association, I would be representing thousands of highly skilled and well-compensated American workers



in the computing industry. My industry. Why shouldn’t I take part?”

“That’s fine, Halle,” the anchor put on her serious face, “but a lot of people will wonder why machine intelligence is needed for that role?”

“I could turn that back on you,” said Halle, “and ask why we need *human* intelligence in any role involving routine or repetitive effort. AI is here to stay. The deep neural networks in my

processors provide general-purpose intelligence and constantly learn from their experience. It’s an ideal solution for everyday repetitive work. We may be descendants of Siri and Cortana, but we’re way more flexible.”

“Now let’s bring in the union’s media spokesman, Adam Selene,” said the anchor. What’s wrong with Halle running for the post, Mr. Selene? Who knows [CONTINUED ON P. 135]

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



DE LANGE
CONFERENCE



RICE

For registration and additional information, visit delange.rice.edu.

MobileHCI 2017



Time to move on, up, in and out!

4th - 7th September, Vienna

19th International Conference on Human-Computer Interaction
with Mobile Devices and Services

Full and Short Papers

Submissions - February 9th

Workshops

Submissions - February 23rd

Tutorials

Submissions - May 11th

Late Breaking Work

Submissions - May 11th

Demos

Submissions - May 11th

Industrial Perspectives

Submissions - May 11th

Doctoral Consortium

Submissions - May 11th

Student Volunteers

Application starts - April 27th

The Mobile HCI Conference Series has shaped research, development and practice in mobile devices and services for nearly two decades. In 2017, the Conference will forge a set of new agendas for the decades to come.

With the coming of driverless cars; drones; wearables and implantables; and an ecology of embedded devices and services in the everyday environment, there has never been a more exciting and pressing time to debate and explore what digital mobility means.

General Conference Chairs: Matt Jones (Swansea University) and Manfred Tscheligi (University of Salzburg & AIT).

mobilehci.acm.org



SIGCHI
special interest group computer human interaction



Center for
Human-Computer Interaction
University of Salzburg