

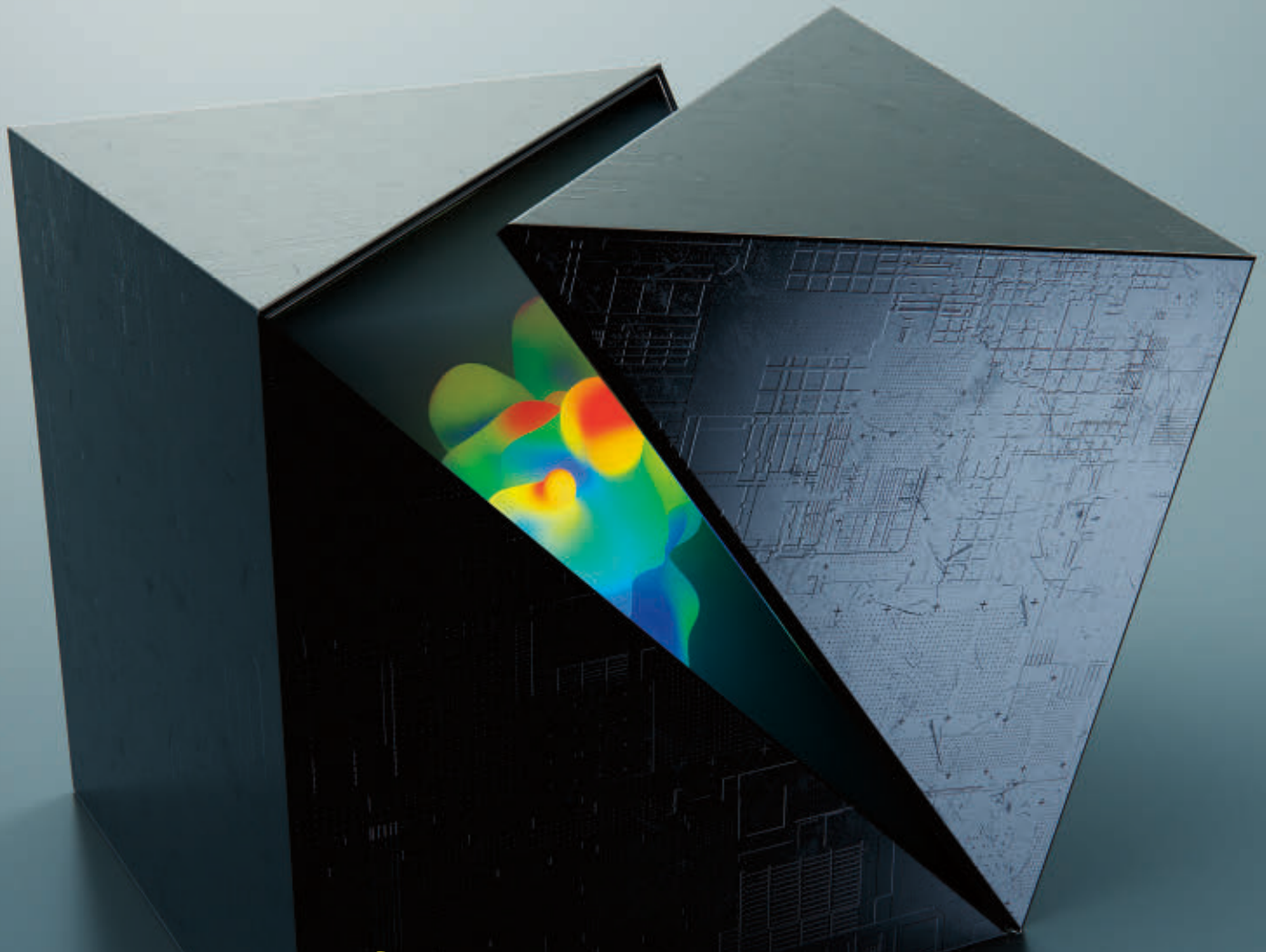
COMMUNICATIONS

CACM.ACM.ORG

OF THE

ACM

01/2020 VOL.63 NO.01



Techniques for Interpretable Machine Learning

Dependable Edge Computing

The Reliability of Enterprise Applications

Will Deepfakes Do Deep Damage?

Cracks in Open Collaboration in Universities



ACM BOOKS Collection II

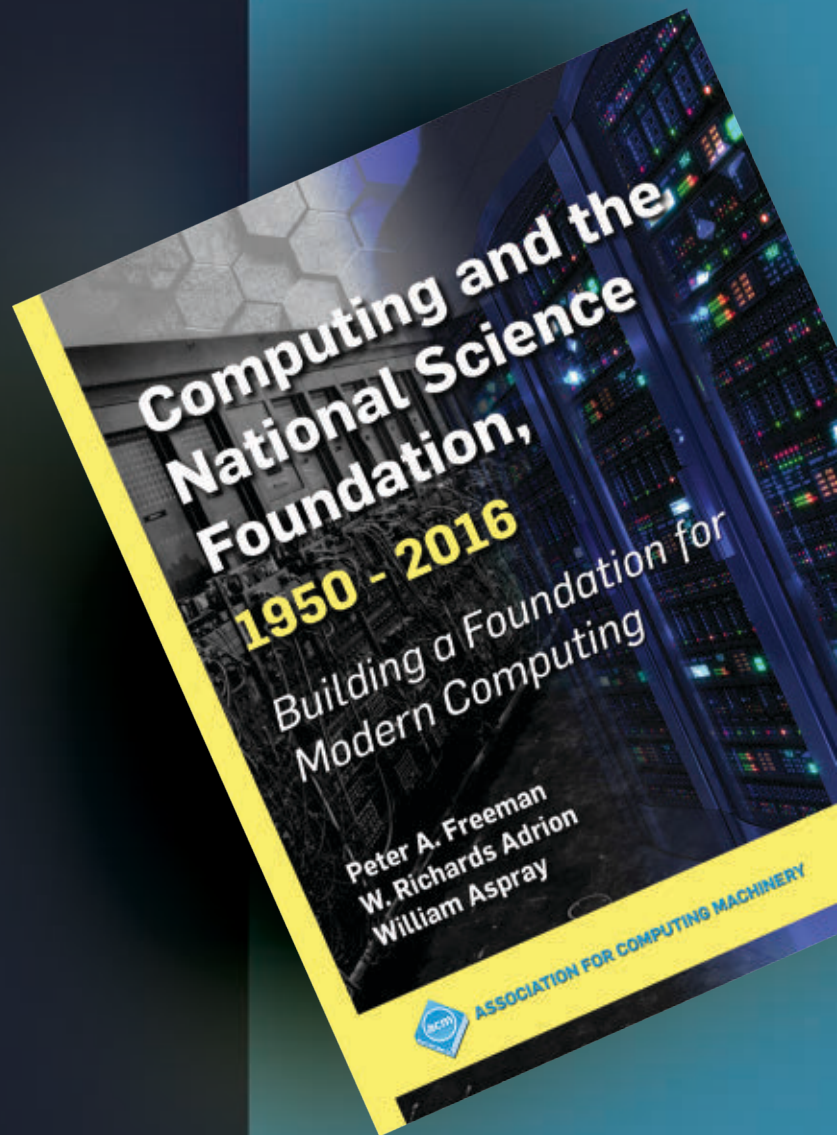
This organizational history relates the role of the National Science Foundation (NSF) in the development of modern computing. Drawing upon new and existing oral histories, extensive use of NSF documents, and the experience of two of the authors as senior managers, this book describes how NSF's programmatic activities originated and evolved to become the primary source of funding for fundamental research in computing and information technologies.

The book traces how NSF's support has provided facilities and education for computing usage by all scientific disciplines, aided in institution and professional community building, supported fundamental research in computer science and allied disciplines, and led the efforts to broaden participation in computing by all segments of society.

Today, the research and infrastructure facilitated by NSF computing programs are significant economic drivers of American society and industry. The NSF has advanced the development of human capital and ideas for future advances in computing and its applications.

This account is the first comprehensive coverage of NSF's role in the extraordinary growth and expansion of modern computing and its use. It will appeal to historians of computing, policy makers and leaders in government and academia, and individuals interested in the history and development of computing and the NSF.

<http://books.acm.org>
<http://store.morganclaypool.com/acm>



Computing and the National Science Foundation 1950-2016 *Building a Foundation for Modern Computing*

**Peter J. Freeman
W. Richards Adrion
William Aspray**

ISBN: 978-1-4503-7271-8
DOI: 10.1145/3335772

Honolulu, Hawai'i, USA
April 25–30 2020



Aloha!

Welcome to CHI 2020, a conference with hundreds of research presentations, dozens of workshops and courses, and over 80,000 ft² (7,500 m²) of interactive exhibits, all on the latest in interactive technology across various domains in Human-Computer Interaction. Join our multicultural community from highly diverse backgrounds at CHI2020 to further investigate and design new and creative ways for people to interact using technology!



chi2020.acm.org

Departments

- 5 **Editorial**
Cracks in Open Collaboration in Universities
By Andrew A. Chien
-
- 7 **Vardi's Insights**
Publish and Perish
By Moshe Y. Vardi
-
- 8 **BLOG@CACM**
In Search of the Shortest Possible Schedule
Bertrand Meyer considers how to speed up software engineering.

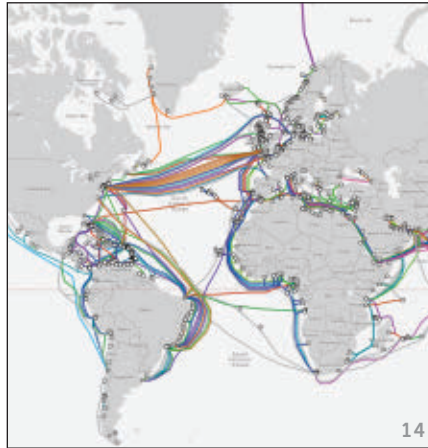
25 **Calendar**

99 **Careers**

Last Byte

- 104 **Upstart Puzzles**
Feedback for Foxes
Searching for the best strategy for shifty maneuvers.
By Dennis Shasha

News



- 11 **Multiplication Hits the Speed Limit**
A problem “around since antiquity” may have been resolved by a new algorithm.
By Erica Klarreich
-
- 14 **How the Internet Spans the Globe**
The modern Internet is made possible by hundreds of thousands of miles of undersea cables.
By Logan Kugler
-
- 17 **Will Deepfakes Do Deep Damage?**
The ability to produce fake videos that appear amazingly real is here. Researchers are now developing ways to detect and prevent them.
By Samuel Greengard

Viewpoints

- 20 **Law and Technology**
Increasing Automation in Policing
Seeking the delicate balance between civil liberties and policing public safety.
By Elizabeth E. Joh
-
- 23 **Technology Strategy and Management**
‘Platformizing’ a Bad Business Does Not Make It a Good Business
Transaction platforms link third-party applications and services providers with users.
By Michael A. Cusumano
-
- 26 **Historical Reflections**
Von Neumann Thought Turing’s Universal Machine was ‘Simple and Neat.’ But That Didn’t Tell Him How to Design a Computer.
New discoveries answer an old question.
By Thomas Haigh and Mark Priestley
-
- 33 **Viewpoint**
Ethics of Technology Needs More Political Philosophy
Incorporating considerations of reasonable pluralism, individual agency, and legitimate authority.
By Johannes Himmelreich
-
- Watch the author discuss this Viewpoint in the exclusive *Communications* video.
<https://cacm.acm.org/videos/more-political-philosophy>
-
- 36 **Viewpoint**
A* Search: What’s in a Name?
A search for algorithmic answers returns unique results.
By James W. Davis and Jeff Hachtel

Practice



38

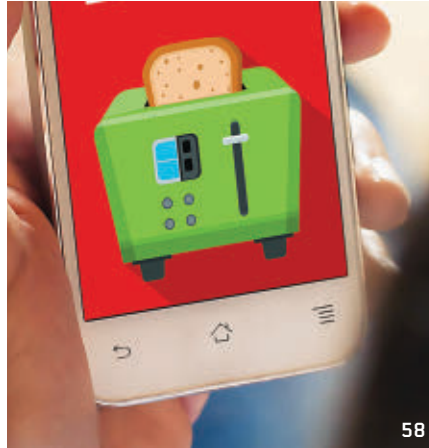
38 **The Reliability of Enterprise Applications**
Understanding enterprise reliability.
By Sanjay Sha

46 **Blockchain Technology: What Is It Good For?**
Industry's dreams and fears for this new technology.
By Scott Ruoti, Ben Kaiser, Arkady Yerukhimovich, Jeremy Clark, and Robert Cunningham

54 **Space Time Discontinuum**
Combining data from many sources may cause painful delays.
By Pat Helland

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

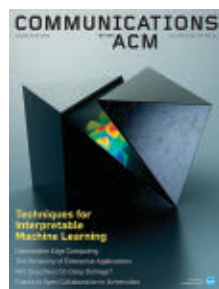
Contributed Articles



58

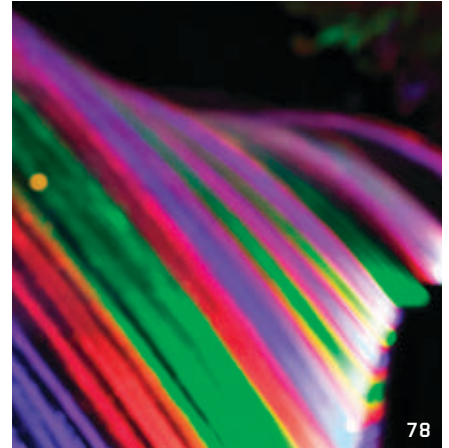
58 **Dependability in Edge Computing**
Edge computing holds great promise, and almost as many challenges in deployment.
By Saurabh Bagchi, Muhammad-Bilal Siddiqui, Paul Wood, and Heng Zhang

About the Cover:



The reigning metaphor for machine learning is the black box; borne of the notion that input and output can be observed, but its inner workings remain a paradox. This month's cover story peers into the black box to interpret complex machine learning models and how those models make decisions. Cover illustration by Peter Crowther Associates.

Review Articles



78

68 **Techniques for Interpretable Machine Learning**
Uncovering the mysterious ways machine learning models make decisions.
By Mengnan Du, Ninghao Liu, and Xia Hu



Watch the authors discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/interpretable-machine-learning>

78 **Mastering Concurrent Computing through Sequential Thinking**
A 50-year history of concurrency.
By Sergio Rajsbbaum and Michel Raynal

Research Highlights

90 **Technical Perspective**
Is There a Geek Gene?
By Mark Guzdial

91 **Evidence That Computer Science Grades Are Not Bimodal**
By Elizabeth Patitsas, Jesse Berlin, Michelle Craig, and Steve Easterbrook



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

Vicki L. Hanson

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

Scott E. Delman

ACM COUNCIL

President

Cherri M. Pancake

Vice-President

Elizabeth Churchill

Secretary/Treasurer

Yannis Ioannidis

Past President

Alexander L. Wolf

Chair, SGB Board

Jeff Jortner

Co-Chairs, Publications Board

Jack Davidson and Joseph Konstan

Members-at-Large

Gabriele Kotsis; Susan Dumais;
Renée McCauley; Claudia Baizer Medeiros;
Elizabeth D. Mynatt; Pamela Samuelson;
Theo Schlossnagle; Eugene H. Spafford
SGB Council Representatives
Sarita Adve and Jeanna Neefe Matthews

BOARD CHAIRS

Education Board

Mehran Sahami and Jane Chu Prey

Practitioners Board

Terry Coatta

REGIONAL COUNCIL CHAIRS

ACM Europe Council

Chris Hankin

ACM India Council

Abhram Ranade

ACM China Council

Wenguang Chen

PUBLICATIONS BOARD

Co-Chairs

Jack Davidson and Joseph Konstan

Board Members

Phoebe Ayers; Nicole Forsgren; Chris Hankin;
Mike Heroux; Nenad Medvidovic;
Tulika Mitra; Michael L. Nelson;
Sharon Oviatt; Eugene H. Spafford;
Stephen N. Spencer; Divesh Srivastava;
Robert Walker; Julie R. Williamson

ACM U.S. Technology Policy Office

Adam Eisgrau

Director of Global Policy and Public Affairs
1701 Pennsylvania Ave NW, Suite 200,
Washington, DC 20006 USA
T (202) 580-6555; acmpo@acm.org

Computer Science Teachers Association

Jake Baskin

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 PUBLICATIONS

Scott E. Delman

cacm-publisher@cacm.acm.org

Executive Editor

Diane Crawford

Managing Editor

Thomas E. Lambert

Senior Editor

Andrew Rosenbloom

Senior Editor/News

Lawrence M. Fisher

Web Editor

David Roman

Editorial Assistant

Danbi Yu

Art Director

Andrij Borys

Associate Art Director

Margaret Gray

Assistant Art Director

Mia Angelica Balaquiot

Production Manager

Bernadette Shade

Intellectual Property Rights Coordinator

Barbara Ryan

Advertising Sales Account Manager

Ilia Rodriguez

Columnists

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

CONTACT POINTS

Copyright permission

permissions@hq.acm.org

Calendar items

calendar@cacm.acm.org

Change of address

acmhhelp@acm.org

Letters to the Editor

letters@cacm.acm.org

WEBSITE

http://cacm.acm.org

WEB BOARD

Chair

James Landay

Board Members

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

AUTHOR GUIDELINES

http://cacm.acm.org/about-communications/author-center

ACM ADVERTISING DEPARTMENT

1601 Broadway, 10th Floor
New York, NY 10019-7434 USA
T (212) 626-0686
F (212) 869-0481

Advertising Sales Account Manager

Ilia Rodriguez
ilia.rodriguez@hq.acm.org

Media Kit acmm mediasales@acm.org

Association for Computing Machinery (ACM)

1601 Broadway, 10th Floor
New York, NY 10019-7434 USA
T (212) 869-7440; F (212) 869-0481

EDITORIAL BOARD

EDITOR-IN-CHIEF

Andrew A. Chien

aic@cacm.acm.org

Deputy to the Editor-in-Chief

Morgan Denlow

cacm.deputy.to.aic@gmail.com

SENIOR EDITOR

Moshe Y. Vardi

NEWS

Co-Chairs

Marc Snir and Alain Chesnais

Board Members

Tom Conte; Monica Divitini; Mei Kobayashi;
Rajeev Rastogi; François Sillion

VIEWPOINTS

Co-Chairs

Tim Finin; Susanne E. Hambrusch;

John Leslie King; Paul Rosenbloom

Board Members

Terry Benzel; Michael L. Best; Judith Bishop;
Lorrie Cranor; Boi Falting; James Grimmelman;
Mark Guzdial; Haym B. Hirsch;
Richard Ladner; Carl Landwehr; Beng Chin Ooi;
Francesca Rossi; Len Shustek; Loren Terveen;
Marshall Van Alstyne; Jeannette Wing;
Susan J. Winter

PRACTICE

Co-Chairs

Stephen Bourne and Theo Schlossnagle

Board Members

Eric Allman; Samy Bahra; Peter Bailis;
Betsy Beyer; Terry Coatta; Stuart Feldman;
Nicole Forsgren; Camille Fournier;
Jessie Frazelle; Benjamin Fried; Tom Killalea;
Tom Limoncelli; Kate Matsudaira;
Marshall Kirk McKusick; Erik Meijer;
George Neville-Neil; Jim Waldo;
Meredith Whittaker

CONTRIBUTED ARTICLES

Co-Chairs

James Larus and Gail Murphy

Board Members

Robert Austin; Kim Bruce; Alan Bundy;
Peter Buneman; Jeff Chase; Yannis Ioannidis;
Gal A. Kaminka; Ben C. Lee; Igor Markov;
Lionel M. Ni; Doina Precup;
Shankar Sastry; m.c. schraefel; Ron Shamir;
Sebastian Uchitel; Hannes Werthner;
Reinhard Wilhelm

RESEARCH HIGHLIGHTS

Co-Chairs

Azer Bestavros, Shriram Krishnamurthi,

and Orna Kupferman

Board Members

Martin Abadi; Amr El Abbadi;
Animashree Anandkumar; Sanjeev Arora;
Michael Backes; Maria-Florina Balcan;
David Brooks; Stuart K. Card; Jon Crowcroft;
Alexei Efros; Bryan Ford; Alon Halevy;
Gernot Heiser; Takeo Igarashi;
Srinivasan Keshav; Sven Koenig;
Ran Libeskind-Hadas; Karen Liu; Greg Morrisett;
Tim Roughgarden; Guy Steele, Jr.;
Robert Williamson; Margaret H. Wright;
Nicolai Zeldovich; Andreas Zeller

SPECIAL SECTIONS

Co-Chairs

Sriram Rajamani, Jakob Rehof,
and Haibo Chen

Board Members

Tao Xie; Kenjiro Taura; David Padua

ACM Copyright Notice

Copyright © 2020 by Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to publish from permissions@hq.acm.org or fax (212) 869-0481.

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

Subscriptions

An annual subscription cost is included in ACM member dues of \$99 (\$40 of which is allocated to a subscription to *Communications*); for students, cost is included in \$42 dues (\$20 of which is allocated to a *Communications* subscription). A nonmember annual subscription is \$269.

ACM Media Advertising Policy

Communications of the ACM and other ACM Media publications accept advertising in both print and electronic formats. All advertising in ACM Media publications is at the discretion of ACM and is intended to provide financial support for the various activities and services for ACM members. Current advertising rates can be found by visiting <http://www.acm-media.org> or by contacting ACM Media Sales at (212) 626-0686.

Single Copies

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

COMMUNICATIONS OF THE ACM

(ISSN 0001-0782) is published monthly by ACM Media, 1601 Broadway, 10th Floor New York, NY 10019-7434 USA. Periodicals postage paid at New York, NY 10001, and other mailing offices.

POSTMASTER

Please send address changes to *Communications of the ACM* 1601 Broadway, 10th Floor New York, NY 10019-7434 USA

Printed in the USA.



Association for Computing Machinery





Andrew A. Chien

DOI:10.1145/3374766

Cracks in Open Collaboration in Universities

UNIVERSITIES AND THE interchange of scholars and students in international collaborations have long played an important role in knitting a fabric of human relationships and shared understanding. This fabric is fraying rapidly.

In recent *Communications'* editorials (January and September 2019), I called out rising political tensions and conflict that challenge international open collaboration in the university. My objective was to inspire the computing community to come together and address these rising challenges.

In the past six months, the situation has worsened with accelerating economic decoupling betwixt China and the U.S., increasing political conflict, and the persistent reality that China's political system is not evolving toward openness or democracy. Instead, China's growing economic and military strength appears to embolden assertion of international power and internal crackdowns.^{a,b,c,d} Analysts increasingly opine that China lacks the internal civil stability essential to be a good global economic partner. The U.S. has moved to restrict technology flow and investment from allies to China. Directly, a growing strategic consensus believes competition between U.S. and

China is shaping the world into two Internets, separated by a new "Berlin wall."^e These two models for the Internet differ in control (government control vs. open), companies (Baidu/Tencent/Alibaba vs. Google/Facebook/Amazon), and increasingly on their technology base.^f

What Are the Implications?

U.S. government actions to restrict technology flow (export control) are reducing the ability of universities to collaborate with some foreign corporations and organizations. Programs instituted to combat intellectual property theft create new reporting requirements. Funded research by corporations on the "entity list" is prohibited, causing Huawei to disband its U.S. research organization.^g Collectively, these actions have a chilling impact on collaboration and create rising anxiety. China's government-rhetoric and action against its own citizens in Xinjiang and Hong Kong reminds the world the government maintains power by force within China and through intimidation campaigns abroad. The resulting tensions produce covert and overt suppression of open communication and learning.^{h,i,j}

There is surprising growth of Internet application bubbles that feed Chinese students abroad a steady stream of censored facts and spin from behind the "Great Firewall"; see, for example, "College Daily."^k

How Can We Protect Collaboration?

In a peacefully integrated and globalized world, nationality need not be apparent. In a world with national military and "no rules" economic competition, nationality takes on critical importance in technologies of direct military and economic importance—such as many dimensions of computing (AI, cybersecurity, HPC). In such a world, we must do three things to protect open collaboration: 1) Community members must be open and transparent about nationality (citizenship) so others can fulfill their responsibilities; 2) all must recognize that nationality (citizenship) confers responsibilities that should be respected by the community; and 3) communities should talk about sensitive areas and design policies/structures that are inclusive where possible, but draw clearly defined and understood boundaries where necessary. If we do not address these problems, solutions will be imposed upon us. Without these steps, the risk is that distrust will grow, undermining the relationships and trust that make open collaboration so fruitful.

k H. Zhang. The 'post-truth' publication where Chinese students in America get their news. *New Yorker*, Aug. 19, 2019.

Andrew A. Chien, EDITOR-IN-CHIEF

From 2005–2010, **Andrew Chien** led Intel's international academic research and education engagements as well as its network of Open Collaborative Research Laboratories. He created two thriving industry-university collaborative research centers at UCSD and the University of Chicago.

- a I. Ali. Pentagon says China missile test in South China Sea 'disturbing.' Reuters, July 2, 2019.
- b C. Buckley and C. Horton. Xi Jinping warns Taiwan that unification is the goal and force is an option. *New York Times*, Jan. 1, 2019.
- c L. Beachum. Uighurs and their supporters decry Chinese 'concentration camps,' 'genocide' after Xinjiang documents leaked. *Washington Post*, Nov. 17, 2019.
- d J. Griffiths, R. Wright, B. Westcott and H. Regan. Protesters try to escape Hong Kong university after violent night. CNN, Nov. 18, 2019.

- e T.J. Friedman. The world-shaking news that you're missing: The U.S.-China divide isn't just about trade. *New York Times*, Nov. 26, 2019.
- f J. Bernstein. The American Internet sucks. The alternative is China. Buzzfeed News, Nov. 17, 2019.
- g Huawei's U.S. research arm slashes more than 70% of workforce. *The Straits Times*, July 23, 2019.
- h I. Kwai. What Chinese students abroad really think about Hong Kong's protests. *New York Times*, Sept. 17, 2019.
- i Hong Kong protests: Sheffield university students clash. BBC News, Oct. 2, 2019.
- j M. Melia. Tensions over Hong Kong unrest flare on US college campuses. AP, Oct. 2, 2019.

Inviting Young Scientists



HEIDELBERG
LAUREATE
FORUM



Association for
Computing Machinery

Meet Great Minds in Computer Science and Mathematics

As one of the founding organizations of the Heidelberg Laureate Forum <http://www.heidelberg-laureate-forum.org/>, ACM invites young computer science and mathematics researchers to meet some of the preeminent scientists in their field. These may be the very pioneering researchers who sparked your passion for research in computer science and/or mathematics.

These laureates include recipients of the ACM A.M. Turing Award, the ACM Prize in Computing, Abel Prize, the Fields Medal, and the Nevanlinna Prize.

The 8th Heidelberg Laureate Forum will take place **September 20–25, 2020** in Heidelberg, Germany.

This week-long event features presentations, workshops, panel discussions, and social events focusing on scientific inspiration and exchange among laureates and young scientists.

Who can participate?

Students of computer science and mathematics (or closely related fields) at the Undergraduate/Pre-Master, Graduate Ph.D. and Postdoc levels can apply for the Heidelberg Laureate Forum. Postdocs include young researchers in classical postdoc positions as well as those who have recently completed (within five years) their Ph.D. and are still strongly interested in scientific matters, even if currently working in a non-scientific environment.

How to apply:

Online: <https://application.heidelberg-laureate-forum.org/>
Materials to complete applications are listed on the site.

What is the schedule?

The application deadline is **February 14, 2020**.

Successful applicants will be notified by **mid-late April 2020**.

More information available on
<https://www.heidelberg-laureate-forum.org/young-researchers/faq.html>





Moshe Y. Vardi

DOI:10.1145/3373386

Publish and Perish

OVER THE PAST decade I have penned several columns that were critical of the current computing-research publication system, with its heavy reliance on conference publishing. These columns were widely read, and the feedback I received was generally quite positive, but they had zero impact on how we go about publishing our research. Conferences still provide the main vehicle for dissemination of curated computing research. What did I miss?

I believe the main advantage that conferences have over journals is that of predictability. Conferences have clear dates for submission, author feedback, and notification. Journals, in general, have none of these. Such predictability is both a powerful motivating factor and a source of comfort. In spite of their flaws, conference publishing dominates because of the predictability it provides.

Yet the dominance of conference publication comes at a cost. As publishing one's paper at a prestigious conference has become the standard way to build professional credentials, expectations with respect to quality have risen. Reviewers expect papers to be of polished archival quality, and often reject papers—even ones that present innovative, interesting research—that fail to meet their standards for such quality. Rejected papers are then revised and submitted to another conference, but will be judged by another set of reviewers, with somewhat differing expectations. Good papers can bounce from conference to conference, imposing a huge cost on the research community in terms of the reviewing effort. Viewed from this perspective, the predictability of conference publishing is rather illusory.

A simple remedy to this problem is for conferences to adopt a standard element of journal publishing, which is the revision. Some conferences have already adopted this practice and allow a submission cycle to include two rounds of reviewing, to enable authors to revise their papers. This practice should be adopted by all computing-research conferences, I would argue.

But the much bigger issue is that a computing-research conference is “a journal that meets in a hotel.” Every paper accepted to a conference requires one or more authors to travel up to halfway around the world in order to present the paper in an oral or poster presentation. The conference publication system is, therefore, a major source of air travel. For example, over the past 20 years I have air traveled more than two million miles. I used to think of that as a professional status symbol.

But as every passenger of a transoceanic flight contributes about 1.8 tons of CO₂ to the atmosphere, it is fair to estimate that a participant in a conference contributes on the average one ton of CO₂ to the atmosphere (also taking into account hotel and conference rooms air-conditioning and the like). The conference-publication system thus adds to the atmosphere annually tens of thousands of tons of CO₂. As the reality of human-caused climate change is getting clearer by the day, the contribution of our profession to the approaching “climate apocalypse” cannot be ignored. My professional “badge of honor” is turning into a badge of shame, I am afraid.

Of course, conferences are more than a paper-publishing system. First and foremost, they are vehicles for information sharing, community building, and networking. But these can be decoupled from research publishing,

and other disciplines are able to achieve them with much less travel, usually with one major conference per year. Can we reduce the carbon footprint of computing-research publishing?

While ACM has instituted the Carbon Offset Program,^a I believe we need to go further. I propose that ACM (and other professional computing associations) establish a sweeping policy change that would apply immediately to all its conferences, requiring that authors of accepted papers that must fly to participate in a conference may opt out from in-person involvement and contribute instead by video. This will not only reduce air travel but will also broaden participation in computing-research conferences by enabling authors with disabilities or with family constraints to partake as well. An author who elects to participate remotely should pay a reduced registration fee to cover conference expenses. Once we allow authors to attend conferences virtually, we should allow the same option to other conference participants. We will then be able to observe the value of in-person conference participation to our community. My suspicion is that it is much less valuable than we would like to believe.

I believe that ACM should take a leadership role, as it did when ACM Council adopted the Policy Against Harassment at ACM Activities, in recognizing the climate emergency we face and in doing its share to reduce its environmental footprint.

Follow me on Facebook and Twitter.  

^a <http://bit.ly/2QRgz3c>

Moshe Y. Vardi (vardi@cs.rice.edu) is the Karen Ostrum George Distinguished Service Professor in Computational Engineering and Director of the Ken Kennedy Institute for Information Technology at Rice University, Houston, TX, USA. He is the former Editor-in-Chief of *Communications*.

Copyright held by author.

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



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

DOI:10.1145/3371385

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

In Search of the Shortest Possible Schedule

Bertrand Meyer considers how to speed up software engineering.



Bertrand Meyer
The Shortest Possible Schedule Theorem: Yes, You Can Throw Money at Software Deadlines

<http://bit.ly/36tYxcX>

October 27, 2019

Some folk wisdom going around in software engineering, often repeated for decades, is just wrong. It can be particularly damaging when it affects key aspects of software development and is contradicted by solid scientific evidence. The present discussion covers a question that meets both of these conditions: whether it makes sense to add staff to a project to shorten its delivery time.

My aim is to popularize a result that is well known in the software engineering literature, going back to the early work of Barry Boehm,¹ and explained with great clarity by Steve McConnell in his 2006 book on software cost estimation *Shortest Possible Schedule*.² While an empirical rather than a logical result, I believe it deserves to be called a theorem (McConnell stays shy of using the term) because it is as close as we have in the area of software engineering management to a universal property, confirmed

by numerous experimental studies.

This article contributes no new concept since McConnell's Chapter 20 says all there is to say about the topic; my aim is simply to make the Shortest Possible Schedule Theorem better known, in particular to practitioners.

The myth about shortening project times begins with an observation clearly correct. Everyone understands if our project has been evaluated through accepted cost-estimation techniques, to require three developers over a year, we cannot hire 36 people to complete it in a month. Productivity does not always scale up.

Yet neither does common sense. Too often the conclusion from the preceding trivial observation takes the form of an old saw, "Brooks' Law": adding people to a late project delays it further. The explanation is that newcomers cost more through communication overhead than they bring through actual contributions. While a few other sayings of Brooks' *Mythical Man-Month* have stood the test of time, this one has always struck me as describing, rather than any actual law, a definition of bad management. Of course if you keep haplessly throwing people at deadlines, you are going to add

communication problems and make things worse. But if you are a competent manager, expanding team size is one of the tools at your disposal to improve the state of a project, and it would be foolish to deprive yourself of it. A definitive refutation of the supposed law, also by McConnell, was published 20 years ago.³

For all the criticism it deserves, Brooks' pronouncement was limited in its scope: it addressed adding staff to a project already late. It is even more wrong to apply it to the more general issue of cost-estimating and staffing software projects at any stage of their progress. Forty-year-old platitudes have even less weight here. As McConnell's book shows, cost estimation is no longer a black art. It is not an exact science either, but techniques exist to produce solid estimates.

The Shortest Possible Schedule theorem is one of the most interesting results. It is much more interesting than Brooks's purported law, because it is backed by empirical studies (rather than asking us to believe a pithy pronouncement), and instead of a general negative view, it provides a positive result complemented by a limitation of that result, and both are expressed quantitatively.

Figure 1 gives the general idea of the SPS theorem. Figure 2 provides a more precise view.

The "nominal project" is the result of a cost and schedule estimation yielding the optimum point. The figure and the theorem give project managers both a reason to rejoice and a reason to despair:

► Rejoice: by putting in more money, that is, more people (in software engi-

neering, project costs are people costs),^a you can bring code to fruition faster.

► Despair: there is a firm limit to the time you can gain: 25%. It seems a universal constant of software engineering.

The “despair” typically gets the most attention at first, since it sets an absolute value on how much money can buy (so to speak) in software: try as hard as you like, you will never get below 75% of the nominal (optimal) value. The “impossible zone” in Figure 1 expresses the fundamental limitation. This negative result is the reasoned, precise modern replacement for the older folk “law.”

The positive part is just as important. A 75%-empty glass is also 25% full. It may be disappointing for a project manager to realize no amount of extra manpower will make it possible to guarantee more than a 25% reduction in time. But it is just as important to know such a reduction, not at all insignificant, is reachable given the right funding, people, tools, and management skills. The last point is critical: money by itself does not suffice, you need management; Brooks’ Law, as noted, is mostly an observation of the effects of bad management.

Figure 1 only carries the essential idea, and is not meant to provide precise numerical values. Figure 2, the original figure from McConnell’s book, plots effort against time rather than the reverse but, more importantly, it shows several curves, each corresponding to a published empirical study or cost model surveyed by the book.

On the left of the nominal point, the curves show how, according to each study, increased cost leads to decreased time. They differ on the details: how much the project needs to spend, and which maximal reduction it can achieve. They all agree on the basic Shortest Possible Schedule result: spending can decrease time, and the maximal reduction will not exceed 25%.

The figure also provides an answer, although a disappointing one, to another question. So far this discussion has assumed time was the critical resource and we were prepared to spend more to get a product out sooner. But sometimes it is the other way around: the critical resource is cost or, concretely, the number of devel-

^a This is the accepted view, even though one might wish the industry paid more attention to investment in tools in addition to people.

Figure 1. General view of the Shortest Possible Schedule theorem.

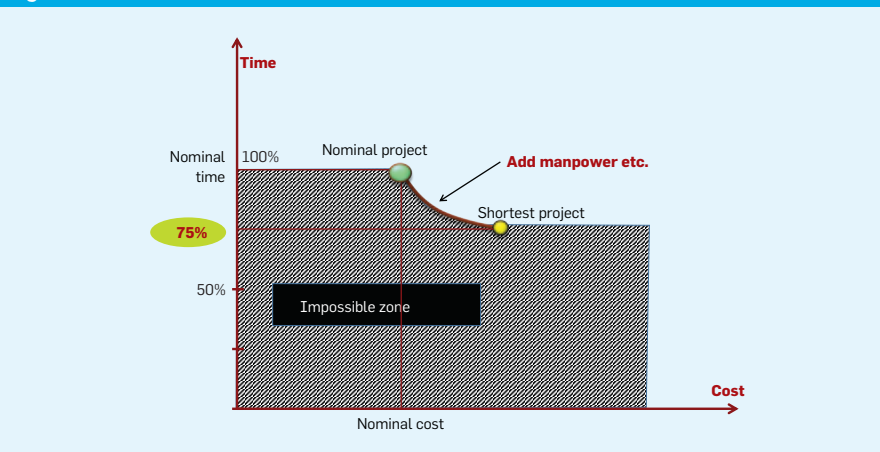
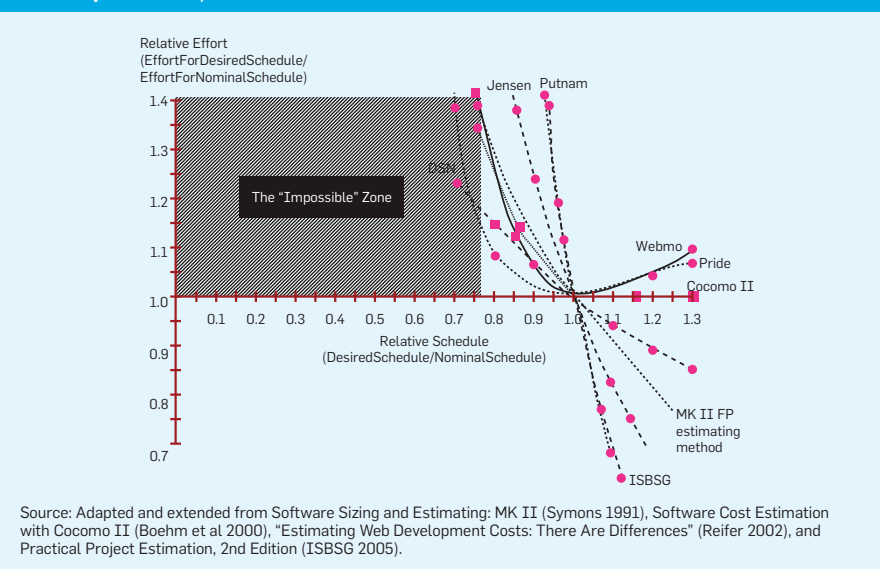


Figure 2. Original illustration of the Shortest Possible Schedule (reproduced with the author’s permission)



Source: Adapted and extended from Software Sizing and Estimating: MK II (Symons 1991), Software Cost Estimation with Cocomo II (Boehm et al 2000), “Estimating Web Development Costs: There Are Differences” (Reifer 2002), and Practical Project Estimation, 2nd Edition (ISBSG 2005).

opers. Assume nominal analysis tells us the project will take four developers a year and, correspondingly, cost 600K (choose your currency). We have a budget of 400K. Can we spend less by hiring fewer developers, accepting it will take longer?

On that side, right of the nominal point in Figure 2, McConnell’s survey of surveys shows no consensus. Some studies and models do lead to decreased costs, others suggest that with the increase in time, the cost will increase, too.

(Here is my interpretation, based on my experience rather than on any systematic study: you can achieve the original goal with a somewhat smaller team over a longer period, but the effect on the final cost can vary. If the new time is $t' = t + T$ and the new team size $s' = s - S$, t and s being the nominal values, the cost difference is proportional to $Ts - t'S$. It can be positive as well as negative, de-

pending on the values of the original t and s and the precise effect of reduced team size on project duration.)

The firm result, however, is the left part of the figure. The Shortest Possible Schedule theorem confirms what good project managers know: you can, within limits, shorten delivery times by bringing all hands on deck. The precise version deserves to be widely known. ■

References

1. Boehm, B.W. *Software Engineering Economics*, Prentice Hall, 1981.
2. McConnell, S. *Software Estimation Demystifying the Black Art*, Microsoft Press, 2006.
3. McConnell, S.: Brooks’ Law Repealed, in *IEEE Software*, vol. 16, no. 6, pp. 6–8, November-December 1999.

Bertrand Meyer is a professor of software engineering (emeritus) at ETH Zurich (Switzerland), chief technology officer of Eiffel Software (Goleta, CA, USA), professor at Politecnico di Milano (Italy), and head of the software engineering lab at Innopolis University (Russia).

SHAPE THE FUTURE OF COMPUTING. JOIN ACM TODAY.

www.acm.org/join/CAPP

SELECT ONE MEMBERSHIP OPTION

ACM PROFESSIONAL MEMBERSHIP:

- Professional Membership: \$99 USD
- Professional Membership plus ACM Digital Library: \$198 USD (\$99 dues + \$99 DL)

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 computing. Membership in ACM-W is open to all ACM members and is free of charge.

PAYMENT INFORMATION

Name _____

Mailing Address _____

City/State/Province _____

ZIP/Postal Code/Country _____

- Please do not release my postal address to third parties

Email Address _____

- Yes, please send me ACM Announcements via email
- No, please do not send me ACM Announcements via email

- AMEX VISA/MasterCard Check/money order

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

By joining ACM, I agree to abide by ACM's Code of Ethics (www.acm.org/code-of-ethics) and ACM's Policy Against Harassment (www.acm.org/about-acm/policy-against-harassment).

I acknowledge ACM's Policy Against Harassment and agree that behavior such as the following will constitute grounds for actions against me:

- Abusive action directed at an individual, such as threats, intimidation, or bullying
- Racism, homophobia, or other behavior that discriminates against a group or class of people
- Sexual harassment of any kind, such as unwelcome sexual advances or words/actions of a sexual nature

BE CREATIVE. STAY CONNECTED. KEEP INVENTING.



ACM General Post Office
P.O. Box 30777
New York, NY 10087-0777

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
www.acm.org/join/CAPP

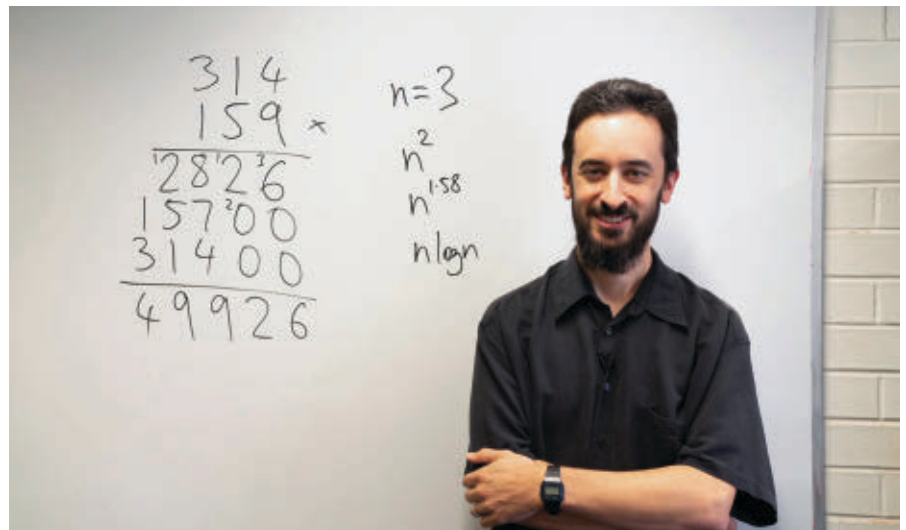
Multiplication Hits the Speed Limit

A problem “around since antiquity” may have been resolved by a new algorithm.

APAPER POSTED ONLINE in March 2019 presents what may be essentially the fastest possible algorithm for one of the oldest problems in mathematics: whole number multiplication. The new algorithm, which can multiply two n -digit numbers in approximately $n(\log n)$ steps, breaks all previous records and reaches what mathematicians conjectured decades ago should be the fundamental speed limit for any multiplication procedure.

“This problem has been around since antiquity,” said Joshua Cooper of the University of South Carolina in Columbia. “It’s extraordinary to see the state of the art reach what people believe is the truth” about how fast multiplication can be carried out.

The new algorithm outstrips other algorithms only for extremely large numbers, so for now its practical applications are limited. Its theoretical implications, however, are vast. Multiplication lies at the core of nearly every mathematical operation: its speed is as central to arithmetical complexity theory as the speed of light is to physics, said Joris van der Hoeven of the French National Center for Sci-



David Harvey, above, demonstrating how standard multiplication is impractical when multiplying astronomically large numbers together. Below, Joris van der Hoeven, who worked with Harvey on the new algorithm to speed multiplication of such numbers.

entific Research in Paris, who created the new algorithm along with David Harvey of Australia’s University of New South Wales in Sydney.

The new paper immediately implies, for example, that it is possible to calculate the first n digits of the reciprocal or square root of a number in approximately $n(\log n)$ steps, and the first n digits of transcendental constants such as π and e in roughly $n(\log^2 n)$ steps.



“Now we know that all these algorithms that depend on multiplication are the time complexity that we thought they were,” Cooper said.

Too Many Logs

The standard multiplication algorithm children learn in elementary school takes approximately n^2 steps, since every digit of the first number must be multiplied by every digit of the second number. For millennia, no one knew any significantly faster multiplication procedure than this simple method. In 1960, Andrey Kolmogorov—one of the preeminent mathematicians of the 20th century—challenged attendees of a seminar at Moscow State University to prove there is no multiplication algorithm that runs in fewer than about n^2 steps.

Anatoly Karatsuba, a 23-year-old student attending the seminar, set out to meet this challenge, but instead proved the opposite. Karatsuba came up with a clever but elementary way to combine the digits of two numbers to compute their product in approximately $n^{1.58}$ steps.

“It’s one of these incredible things that seems so simple once you see it, but no one saw it until Karatsuba did,” Harvey said.

Other mathematicians quickly found improvements to Karatsuba’s algorithm. Then in 1971, Arnold Schönhage and Volker Strassen made another huge leap, devising an algorithm whose running time is about $n(\log n)(\log(\log n))$ —vastly more efficient than $n^{1.58}$ for large values of n . A streamlined version of Schönhage and Strassen’s algorithm lies at the heart of the GNU Multiple Precision Arithmetic Library used by all the standard arithmetic software packages (although for numbers smaller than a few hundred thousand digits, the library uses other approaches, including Karatsuba’s algorithm).

Schönhage and Strassen’s algorithm, which laid the groundwork for the new algorithm announced this past March, leverages the fast Fourier transform, a procedure for sampling and reconstructing polynomials that is used widely in signal processing. It is easy to translate an integer multiplication problem into a problem about polynomials: Simply use the digits of

the two numbers as the coefficients of two polynomials. So, for example, if you want to multiply 635 and 258, you can convert the two numbers into the polynomials $6x^2+3x+5$ and $2x^2+5x+8$. Multiplying these two polynomials gives $12x^4+36x^3+73x^2+49x+40$, and if we plug in the value $x=10$, the polynomial outputs the product of 635 and 258, namely 163,830.

If you calculate the polynomial product by multiplying the two polynomials term by term, as students learn to do in algebra class, this translation doesn’t achieve any speedup over the n^2 integer multiplication algorithm. Yet there is a way to vastly speed up polynomial multiplication. Any polynomial whose highest exponent is k is completely determined by its values at $k+1$ different inputs. So in the example above, the product polynomial, whose highest exponent is 4, is uniquely determined by its values at any five inputs. That means that if we choose our five favorite x values, evaluate $6x^2+3x+5$ and $2x^2+5x+8$ at those five values and then multiply the corresponding outputs, those five multiplications already give us enough information to reconstruct the product polynomial (compared with nine multiplications if we multiply the two polynomials term by term).

What this analysis sweeps under the rug, though, is the cost of first evaluating $6x^2+3x+5$ and $2x^2+5x+8$ at the five inputs, and then reconstructing the product polynomial at the end of the

No one thought it would be possible to bring the running time of integer multiplication down to roughly n steps, which would put multiplication on a par with addition.

procedure. That’s where the fast Fourier transform comes in: It provides a speedy way to do such polynomial evaluations and reconstructions, provided the five input values are chosen carefully (specifically, they should be the five complex numbers whose fifth powers equal 1).

Using the fast Fourier transform, combined with a more sophisticated way of converting numbers into polynomials than the naïve digit-by-digit translation above, Schönhage and Strassen were able to achieve their $n(\log n)(\log(\log n))$ algorithm. For more than three decades after their work, no one could come up with anything significantly faster.

Yet computer scientists found the $n(\log n)(\log(\log n))$ running time disturbingly inelegant. For that reason, Schönhage and Strassen’s algorithm didn’t seem like the final word on the subject.

No one thought it would be possible to bring the running time of integer multiplication all the way down to roughly n steps, which would put multiplication on a par with addition. However, Schönhage and Strassen, along with many others, suspected the true complexity of multiplication—the running time of the fastest possible algorithm—should be $n(\log n)$, not $n(\log n)(\log(\log n))$.

“Everybody feels like multiplication is more complicated than addition,” said Martin Fürer of Pennsylvania State University in University Park. “But everyone thought the $\log(\log n)$ term should not be necessary. From an aesthetic point of view, it doesn’t look nice. Such a fundamental task as multiplication should have a nice complexity.”

The End of the Story

In 2007, Fürer finally managed to whittle down the $\log(\log n)$ term in Schönhage and Strassen’s algorithm to something slightly smaller. Fürer’s algorithm was impractical for any multiplications people might want to carry out in real life, but the theoretical advance electrified Harvey and van der Hoeven. Over the past five years, they have collaborated on a series of about 10 papers further improving Fürer’s bound. “There were twice as many papers that never got written, and algorithms that never saw the light of

day because they were superseded by something else,” Harvey said.

Finally, in March 2019 the pair figured out how to eliminate the $\log(\log n)$ term completely. Their new algorithm uses a higher-dimensional version of the fast Fourier transform, combined with a method they devised for increasing the number of sampling points to take advantage of additional speedups when the number of sampling points is a power of two. “It’s definitely the hardest paper I ever worked on,” Harvey said.

The algorithm entails rounding off the complex numbers involved in the fast Fourier transform to achieve a balance of speed and precision that is “kind of amazing,” Cooper said. “They’re performing exact integer multiplication, but in the process they’re passing into this other world using complex numbers and polynomials and doing an approximate computation, then coming back and getting an exact answer.”

The $n(\log n)$ bound means Harvey and van der Hoeven’s algorithm is faster than Schönhage and Strassen’s algorithm, or Fürer’s algorithm, or any other known multiplication algorithm, provided n is sufficiently large. For now, “sufficiently large” means almost unfathomably large: Harvey and van der Hoeven’s algorithm doesn’t even kick in until the number of bits in the two numbers being multiplied is greater than 2 raised to the 1729¹² power. (By comparison, the number of particles in the observable Universe is commonly put at about 2⁷⁰.)

Harvey and van der Hoeven made no efforts to optimize their algorithm. This was partly because they were focused on the theoretical advance, and partly because they were tickled when their back-of-the-envelope calculations led them to the number 1729, which, in a famous anecdote, the mathematician Srinivasa Ramanujan called a “very interesting” number (because it is the smallest number that can be written as a sum of two cubes in two different ways). “When I saw this, I burst out laughing,” Harvey recalled.

Other researchers will likely find ways to tweak Harvey and van der Hoeven’s algorithm so it outperforms other algorithms at smaller and smaller

The algorithm entails rounding off the complex numbers in the fast Fourier transform to achieve a balance of speed and precision that is “kind of amazing,” Cooper said.

numbers. What they are unlikely to do, many researchers agree, is come up with any algorithm qualitatively faster than $n(\log n)$. No one knows how to prove this—as a rule, establishing there are no algorithms faster than some bound is much harder than coming up with a new fast algorithm.

Nevertheless, “It would really surprise us if it is possible to do better than $n(\log n)$,” van der Hoeven said. “We feel that the story for integer multiplication ends here.” **□**

Further Reading

Fürer, M.
Faster Integer Multiplication. *SIAM J. Comput.*, Volume 39 Issue 3, 2009, pages 979-1005
<http://bit.ly/2CFvmG6>

Harvey, D. and van der Hoeven, J.
Integer Multiplication in Time $O(n \log n)$.
<http://bit.ly/2QcOrr6>

Karatsuba, A.
The Complexity of Computations. *Proceedings of the Steklov Institute of Mathematics*, Volume 211, 1995, pages 169-183
<http://bit.ly/32M1oed>

Schönhage, A. and Strassen, V.
Schnelle Multiplikation grosser Zahlen. *Computing*, Volume 7, Issue 3-4, September 1971, pages 281-292.
<https://link.springer.com/article/10.1007/BF02242355>

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

© 2020 ACM 0001-0782/20/1 \$15.00

Milestones

ACM Recognizes Distinguished Members

ACM recently inducted 62 Distinguished Members for their outstanding contributions to the field.

The 2019 inductees are long-standing ACM members selected by their peers for a range of accomplishments that have contributed to technologies that underpin how we live, work, and play.

“Each year, it is our honor to select a new class of Distinguished Members,” explains ACM president Cheri M. Pancake. “In everything we do, our overarching goal is to build a community wherein computing professionals can grow professionally and, in turn, contribute to the field and the broader society. We are delighted to recognize these individuals for their contributions to computing, and we hope that the careers of the 2019 ACM Distinguished Members will continue to prosper through their participation with ACM.”

The 2019 ACM Distinguished Members have made contributions in a wide range of technical areas, including artificial intelligence, human-computer interaction, computer engineering, computer science education, cybersecurity, graphics, and networking.

The ACM Distinguished Member program recognizes up to 10% of ACM worldwide membership based on professional experience and significant achievement in the computing field. Candidates must have at least 15 years of professional experience in computing, five years of continuous professional ACM membership, and have achieved a significant level of accomplishment or made a significant impact in computing, computer science, and/or information technology.

Distinguished Members serve as mentors and role models, guiding technical career development and contributing to the field beyond the norm.

The list of new Distinguished Members can be viewed at <http://bit.ly/2CBLJDB>.

How the Internet Spans the Globe

The modern Internet is made possible by hundreds of thousands of miles of undersea cables.

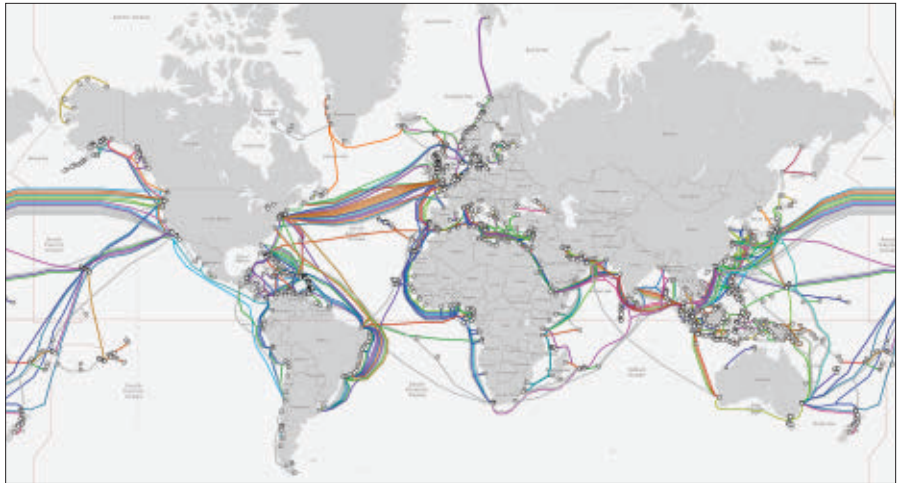
UNDERSEA CABLES ARE RESPONSIBLE for moving data between countries and continents at high speeds, making everything from photo sharing to financial transactions possible. These cables use fiber optics to move data at high speeds to land, where the data is then conveyed via fiber optics to homes and businesses.

Yet, despite the billions of people relying on the data moved by undersea cables, there are only about 380 of them worldwide as of 2019, according to CNN estimates, though they span more than 745,000 miles—or more than three times the distance to the moon.

Given the sheer scope of undersea cables, which often span entire oceans, *The New York Times* estimates an individual undersea cable project can cost up to \$350 million. That is why these cables have historically been laid by global telecommunications firms with the deep pockets and technical expertise necessary to undertake these projects. However, tech companies are increasingly dominating both the use and implementation of undersea cables.

“Some of the main investors in new cables now are Google and Facebook,” says Alan Maudlin, research director at TeleGeography, a telecommunications research firm, because tech giants are increasingly the ones using all that undersea bandwidth.

When it comes to global bandwidth usage, just a handful of services account for more than a third of all bandwidth, according to research from network intelligence company Sandvine. The company’s Global Internet Phenomena Report identifies Web-based media streaming, Netflix, and YouTube as the top three users of global traffic, comprising 34.1% of traffic as



There are more than 745,000 miles of submarine cables supporting the billions of people who rely on the Internet.

of September 2019. Facebook is also in the top 10.

Tech giants like Google (which owns YouTube) and Facebook “have surpassed Internet backbone providers—the traditional telecom carriers—as the largest users of international capacity,” according to TeleGeography. The firm estimates the amount of international capacity deployed by tech companies grew eight times from 2014 to 2018.

These undersea cables connect almost the entire globe. Google is a part or sole owner of 15 different undersea cables, with end points terminating in places that include Chile and France. Facebook is a major capacity buyer or part owner of 10 cables with end points in Singapore, China, and the U.S. (among other countries). Amazon is part owner or a major capacity buyer of five cables, with end points in countries that include Malaysia. Microsoft is part owner or a major capacity buyer of four different cables, with end points in countries that include Spain.

One of the fastest undersea cables in operation today is the Marea cable,

partially owned by Microsoft, Amazon, and Facebook. It transmits data at 160 terabits per second, which is “16 million times faster than the average home Internet connection,” says Microsoft. The Marea cable has enough capacity to “stream 71 million high-definition videos simultaneously,” according to the company.

Despite this ongoing change in cable dominance, undersea cables often force traditional telecoms and the new guard of tech giants to play nice.

The Marea cable is the result of a partnership between Microsoft, Facebook, Amazon, and Telxius, a spinoff that manages telecom infrastructure for multinational telecom Telefonica. That is because advanced cables like Marea are critical to meeting rising demand from the consumers and businesses both telecoms and tech giants serve.

“Marea comes at a critical time,” says Microsoft president Brad Smith. “Submarine cables in the Atlantic already carry 55% more data than transPacific routes, and 40% more data than between the U.S. and Latin

America.” Smith says there is “no question” demand for transAtlantic data flows will increase.

“Only Google has built/funded cables themselves, but even these cables are used by other parties besides Google,” says Maudlin. “For example, the new Dunant cable [owned by Google] between the U.S. and France will have a fiber pair used by the French carrier Orange.”

Tech giants may battle each other for Internet dominance, all while replacing old guard telecoms, but when it comes to laying undersea cable, playing nice seems to be the order of the day.

More Fibers in More Places

Companies may collaborate to build undersea cable networks, but they still must defend these networks from threats. Damage to a single cable can cause widespread disruption. In fact, the potential for cable outages was actually the reason Microsoft helped to create the super-fast Marea cable.

In 2012, Hurricane Sandy (also known as Superstorm Sandy) hit the east coast of the U.S., inflicting nearly \$70 billion in damage to the infrastructure of New York and New Jersey. In addition to the homes and businesses damaged, the storm also damaged the places where undersea cables came ashore, causing service disruptions. Microsoft learned the hard way that New York and New Jersey were home to an unnaturally high convergence of undersea cables making landfall.

That made Microsoft think differently about potential dangers to undersea cables, both below the surface and on land. Today, the Marea cable makes landfall in the U.S. much further south, in Virginia Beach, VA, to avoid disruption if New York or New Jersey ever get hit by a storm of Sandy’s magnitude again.

Being more strategic about where cables make landfall certainly helps. So does building the cables to withstand physical damage and destruction. The fiber optics of undersea cables are wrapped in protective material to withstand harsh underwater environments, natural disasters like submarine earthquakes, and man-made damage from fishing boats and anchors.

The number of undersea cables being laid is increasing to meet demand, but each cable still requires multiple years and millions of dollars to lay.

Typically, “the fibers are wrapped in urethane and wrapped in copper and wrapped again in urethane,” said Byron Clatterbuck, CEO of international telecom company Seacom. The urethane protects the optical fibers, and the copper conducts electricity that powers repeaters designed to regenerate the signal regularly through the cable. Depending on the environment, cables may also be sheathed in extra protective material, such as iron or steel.

The number of undersea cables being laid is increasing to meet demand, but cables still take years and millions of dollars to lay, which puts a ceiling on how many companies can lay cable at a time to expand global bandwidth. That fact has companies exploring ways to get more out of each cable laid.

“One key development is space-division multiplexing,” says Maudlin. “It’s basically a method to increase the number of [spatially distinct] channels that can be put in a single fiber. Google is using this in their new Dunant cable between France and the U.S.”

Dunant, which Google says is scheduled to go live this year, will use space-division multiplexing to provide transmission capacity of 250 terabits of data per second. The search giant says that is enough bandwidth to “transmit the entire digitized Library of Congress three times every second.”

ACM Member News

SOLVING REAL-WORLD PROBLEMS WITH MATH



“I love problem-solving, mathematics, and the practical application of

math to solve interesting, real-world problems,” says Wendi Heinzelman, a professor of electrical and computer engineering, and dean of the Hajim School of Engineering and Applied Sciences, at the University of Rochester in New York.

Heinzelman’s father was an electrical engineer who worked on voice-recognition systems at AT&T Bell Laboratories; his work spurred her interest in the field.

She received her undergraduate degree in electrical engineering from Cornell University, and both her M.S. and Ph.D. degrees from the Massachusetts Institute of Technology in both electrical engineering and computer science. On completing her doctorate, Heinzelman joined the faculty of the University of Rochester in the department of electrical and computer engineering, where she has been ever since.

Her research interests lie in wireless communications, networking, and mobile systems. “Recently, I’ve been working on mobile ad hoc networks through the use of existing protocols, and a standard called Wi-Fi Direct,” Heinzelman says.

“The Wi-Fi Direct standard typically is used where a user is required to set up an ad hoc connection,” she explains, “and I am trying to make it automatic so that large-scale networks can be developed using this protocol.”

Heinzelman also has been exploring mobile health applications beyond simply using mobile phones for their sensors. The possibility of creating user-friendly, online mobile-type applications has captured her interest.

“I think there’s going to be a lot of interesting research questions in how to do that,” she says.

—John Delaney

**Toward ML-Centric
Cloud Platforms**

**Fuzzing:
Hack, Art, Science**

**Directions for
Professional Social
Matching Systems**

**Opening Up the
Baseboard Management
Controller**

**Optimization in
C++ Compilers**

**Guiding Students to
Develop Essential Skills**

**When Human-Computer
Interaction Meets
Community
Citizen Science**

**Automating Visual
Privacy Protection
Using a Smart LED**

Plus the latest news about
how quantum expands
the universe, virtual fraud,
and ways to track shoppers.

In the case of Dunant, which Google is building in tandem with telecom SubCom LLC, the companies are using an approach to space-division multiplexing in which the cable doubles the fiber pairs used from six pairs to 12. The power transmitted across the original six pairs would now be split, resulting in a “two-thirds capacity increase,” according to Georg Mohs, CTO of SubCom LLC.

The technologies incorporated into newer cables like Dunant will make older cables look like dinosaurs—and cause them to go extinct.

“New cables have higher potential capacities than older cables,” says Maudlin. “Eventually older cables, especially those built pre-2003, will be decommissioned as they become economically obsolete compared to higher-capacity cables.”

However, older cables do not always make it to a peaceful retirement. Natural and man-made events can damage or destroy undersea cables.

“Cables routinely have faults all over the world each week,” says Maudlin. He cites some of the primary causes of damage as fishing and anchors. Underwater earthquakes and landslides also can destroy or disrupt undersea cables, he says.

Cable disruption also can affect military operations, as occurred in 2008, when three of the world’s largest undersea cables, in the Mediterranean Sea, were severed by accident. The incident was described in a report by U.K. think tank Policy Exchange that observed, “In a matter of hours, disruptions to regional connectivity had knocked out 80% of the connectivity between Europe and the Middle East,” including communications for American and British military forces in Iraq that were enabled by these cables.

That incident suggests another potential danger to undersea cables: intentional sabotage. Coordinated cable destruction, notes Policy Exchange, could cause major disruptions that target specific nations. For example, cutting just three cables would cause India to lose 70% of its data traffic with Europe. “Cables are not only easily cut, but maps providing the exact locations of all the world’s commercial cabling are freely available in the public domain.”

In mid-2019, a fire broke out on a Russian AS-12 nuclear-powered submarine, killing 14 sailors. It was a tragedy, to be sure. However, the incident also put U.S. officials on edge; they claimed the submarine’s real mission before the fire broke out was to cut undersea cables to disrupt or intercept communications.

According to the BBC, the television channel of Russia’s defense ministry confirmed the country could both cut undersea cables and scan their data, back in 2015. Somewhat discomfortingly, the channel said, “Such an attack on underwater communications is possible only in the event of active hostilities.”

One thing is certain. In wartime and peacetime, the undersea cables that power the Internet are too big to fail. ■

Further Reading

Bach, D.
Microsoft, Facebook and Telxius complete the highest-capacity subsea cable to cross the Atlantic, *Microsoft News*, September 21, 2017, <http://bit.ly/34UrkFO>

Finley, K.
How Google Is Cramming More Data Into Its New Atlantic Cable, *WIRED*, April 5, 2019, <https://www.wired.com/story/google-cramming-more-data-new-atlantic-cable/>

Griffiths, J.
The global internet is powered by vast undersea cables. But they’re vulnerable, *CNN*, July 26, 2019, <https://cnn.it/32KxDdn>

Maudlin, A.
A Complete List of Content Providers’ Submarine Cable Holdings, *TeleGeography*, November 9, 2017, <http://bit.ly/2Qjv3IK>

Satariano, A.
How the Internet Travels Across the Oceans, *The New York Times*, March 10, 2019, <https://nyti.ms/33JVOKC>

Sunak, R.
Undersea Cables, Indispensable, Insecure, *Policy Exchange*, 2017, <http://bit.ly/33JmXx5>

Vusirikala, V.
A quick hop across the pond: Supercharging the Dunant subsea cable with SDM technology, *Google*, April 5, 2019, <http://bit.ly/2KgXGYL>

Logan Kugler is a freelance technology writer based in Tampa, FL, USA. He has written for over 60 major publications.

Will Deepfakes Do Deep Damage?

The ability to produce fake videos that appear amazingly real is here. Researchers are now developing ways to detect and prevent them.

IT HAS BEEN said that the camera doesn't lie. However, in the digital age, it is also becoming abundantly clear that it doesn't necessarily depict the truth. Increasingly sophisticated machine learning combined with inexpensive and easy-to-use video editing software are allowing more and more people to generate so-called *deepfake* videos. These clips, which feature fabricated footage of people and things, are a growing concern in both politics and personal life.

"It's a technology that is easily weaponized," observes Hany Farid, a professor at the University of California, Berkeley.

Not only can deepfakes be used to depict a political candidate or celebrity saying or doing something he or she never said or did, they can depict false news events in an attempt to sway public opinion. And then there are the disturbing issues of blackmail and porn, including revenge porn. A number of deepfake videos have surfaced showing a person's ex-partner nude or en-

"Today's deep neural nets and AI algorithms are becoming better and better at creating images and video of people that are convincing and not real."

gaged in sex acts he or she did not commit. The person creating the deepfake video simply transposes the victim's face onto the body of another person, such as a porn star.

"Today's deep neural nets and AI algorithms are becoming better and better at creating images and video of people that are convincing but

not real," says Siwei Lyu, professor of computer science and director of the Computer Vision and Machine Learning Lab (CVML) of the University at Albany, which is part of the State University of New York. As a result, researchers in digital media forensics, computer scientists, and others are now examining ways to better identify fake videos, authenticate content, and build frameworks to help thwart the rapid spread of deepfakes on social media.

"It's a problem that isn't going to go away," Lyu says.

Deep Trouble

Deepfake technology bubbled to general public awareness in early 2018, when former U.S. president Barack Obama spoke out about the growing dangers of false news and videos. "We are entering an era when our enemies can make it look like anyone is saying anything at any point in time," he stated in a video clip. Except it *was not* Obama actually making the video appearance; it was a deepfake created by comedian Jordan



Paul Scharre of the Center for a New American Security views a deepfake video made by BuzzFeed, which changed what was said by former U.S. president Barack Obama to what is spoken by filmmaker Jordan Peele (right on screen).

Peele in conjunction with online publication BuzzFeed. The goal was to help educate the public about the potential dangers of deepfakes.

Other examples abound. Parkland shooting survivor Emma González was depicted tearing up a copy of the U.S. Constitution, instead of a shooting range target. Celebrities such as Gal Gadot, Emma Watson, Hilary Duff, and Jennifer Lawrence have been inserted into porn scenes. A political party in Belgium depicted Donald Trump taunting Belgium for remaining in the Paris Climate Agreement. In India, a journalist who reported corruption in Hindu national politics found her image inserted into a fake porn video along with her home address and telephone number; she received numerous death and rape threats.

Deepfakes also can incorporate audio. For instance, crooks recently impersonated the voice of a U.K.-based energy company's CEO in order to convince workers to wire \$243,000 cash to their account.

Although photo, video, and audio manipulation techniques have been around for years (evolving from dodging and burning photos in the darkroom to using photo editing software to alter digital images), computer-altered video raises the stakes to a new level. "Advances in artificial intelligence technology have allowed for the creation of fake audiovisual materials that are almost indistinguishable from authentic content, especially to ordinary human senses," said Jeffrey Westling, a Technology and Innovation Policy Fellow with the non-profit, non-partisan R Street Institute, at a U.S. House of Representatives hearing in June 2019.

Westling is not the only person sounding alarms. Says Matt Turek, program manager in the Information Innovation Office of the U.S. Defense Advanced Research Projects Agency (DARPA), "This can affect the political process, law enforcement, commerce and more. There are broad impacts if people can easily manipulate images and video ... and significantly reduce society's trust in visual media."

Yet it is not only fake news that is the problem. Trust in real news and videos is diminished as well when there is a

high degree of uncertainty about what constitutes reality.

Concerns are especially high as the 2020 U.S. presidential election approaches. The rapid spread of false news on social media—such as a May 2019 doctored video of Nancy Pelosi that made the House Speaker appear intoxicated and slurring her words—have demonstrated the power of fake images and social media. The Pelosi video, which was viewed more than 2.5 million times on Facebook and shared many times by prominent political leaders, was actually dubbed a "cheapfake" video because the sound was simply slowed to about three-quarters speed.

Image Is Everything

Deepfakes potentially represent the next frontier in propaganda wars. They could depict fake murders or frame a person for a crime he or she did not commit, they could provide falsified evidence of a weapons system that does not exist, and they could be used to stage news events that never happened, such as immigrants rioting in order to sow discord. They could also be used to falsify evidence in a automobile crash and other events. "Right now, deepfakes tend to be around people, but many other possibilities are plausible," Turek says.

Driving deepfake videos is a growing array of easily downloaded programs with names like AI Deepfake and DeepNude, that allow users to plug in images and synthesize fake content. This includes face-swapping, lip-syncing, and a technique called puppet-master that allows a person to manipulate a video with their own movements and expres-

"There are broad impacts if people can easily manipulate images and video ... and significantly reduce society's trust in visual media."

sions. The computer maps the targeted area, say the face, and transposes that face onto another person's head. A generative adversarial network (GAN) compares the two sets of images or video—the system synthesizing content attempts to fool a second system that's inspecting it. The second machine learning system indicates when it meets a threshold of appearing real. It's then a matter of syncing audio and video. "You provide the images and the machine does the heavy lifting," Farid says.

Over the last few years, computer scientists have fed more and more images into these GANs, so they are able to produce increasingly ultrarealistic fake images. Powerful image manipulation techniques that were once solely the province of movie studios producing scenes in films like *Forrest Gump* and *The Matrix* are filtering onto the desktop and into apps. Farid points out these applications don't require enormous computing resources to accomplish the task. In fact, it often is possible to focus on a specific portion of the body. For example, when Peel created the Obama deepfake video, he only had to manipulate the area around the mouth, and use his voice to impersonate the president's.

For now, there's been plenty of alarm and, except for some documented cases of revenge porn, more hype than reality. However, as Lyu points out, "Awareness is a form of inoculation. The goal is not to stifle innovation—there are a number of positive and innocuous uses for the technology—it's to create an environment where it is not misused and abused."

Spotting Fakes

It often takes a trained eye to spot irregularities in deepfake videos—and human detection is not totally effective, even among experts. That is leading researchers down a path that focuses on machine detection, forensics, authentication, and regulation. Lyu, a pioneer in deepfake detection research, says many approaches, including those he has developed, focus on a basic reality: "A fake video is generated by an algorithm, while a real video is generated by an actual camera. As a result, there are clues and artifacts that can usually be detected. In many cases, there is image warping, lighting

inconsistencies, smoothness in areas, and unusual pixel formations.”

DARPA also is working to elevate detection methods through advanced forensics algorithms that are part of its Media Forensics (MediFor) program. It, too, taps machine learning—much of its research involves the use of GANs. A convolutional neural network (CNN) trains a recurrent neural network (RNN) to spot abnormalities and anomalies. In practical terms, a computer might examine pixels in a photo or video and determine whether the laws of physics were violated in the making of the video. “Is lighting consistent, are shadows correct, does the weather or lighting match the date the video was captured?” Turek asks.

Of course, battling deepfake algorithms with detection algorithms using CNNs, RNNs, and other methods ultimately leads to a perpetual machine-learning cat-and-mouse game.

Another area of research revolves around the use of digital watermarks, which could be embedded in news content, business videos, and other materials where a high level of trust is required. However, a big problem with visual watermarks, Lyu says, is that they can be manipulated easily, and veracity is often based on the goodwill of the user. “It’s ultimately a voluntary system and those creating fake videos aren’t likely to use them,” he says.

Researchers also are exploring ways to embed more sophisticated certificates or tokens into videos to authenticate them using cryptographically signed hashes that are stored on a blockchain. One startup, San Diego-based Truepic, is now working with chip maker Qualcomm to extract a signature from an image so it can be verified later. Another firm, U.K.-based Serelay, also has created a verification process to which a number of insurance companies have signed on already, in order to verify claims. Farid, who is a consultant for Truepic, says these validation systems could prove particularly valuable for users of social media sites, where there currently is minimal oversight and high levels of false news circulating because it is highly profitable to these organizations.

Farid believes social media sites also need to do a better job of inhibiting the spread of false news, including

Battling deepfake algorithms with detection algorithms using CNNs, RNNs, and other methods ultimately leads to a perpetual machine-learning cat-and-mouse game.

deepfakes. “They cannot wash their hands of all responsibility. When they know something is fake—and it has been clearly proven—they need to put monetization aside and do something to curb the spread.”

The Legal Picture

Not surprisingly, deepfakes are also testing the legal system and prompting the U.S. Congress, states, and other entities to take action. For example, the “Malicious Deep Fake Prohibition Act of 2018” (S.3805) would introduce penalties for those who create, with intent to distribute, fake videos that “facilitate criminal or tortious conduct.” In September 2019, Texas passed a law specifically prohibiting deepfake videos aimed at harming candidates for public office or to influence elections. A month later, California had passed a ban on sharing deepfake videos within two months of an election; the state also enacted a separate bill that makes it easier to sue over deepfake porn videos.

“Laws must be updated to protect against clear cases of digital harassment, such as revenge porn, but government entities must avoid legislating for or against specific features because the technology is evolving rapidly,” says Albert Fox Cahn, executive director of the Surveillance Technology Oversight Project at the Urban Justice Center, a not-for-profit organization that promotes privacy and civil rights.

However, not everyone agrees that new laws are needed. Electronic Fron-

tier Foundation civil liberties director David Greene noted in a February 2018 online post: “If a deepfake is used for criminal purposes, then criminal laws will apply. This includes harassment and extortion ... There is no need to make new, specific laws about deepfakes in either of these situations.” In civil cases, he noted, the tort of False Light invasion of privacy is applicable. “False light claims commonly address photo manipulation, embellishment, and distortion, as well as deceptive uses of non-manipulated photos for illustrative purposes.”

The one thing everyone can agree on is that a framework of fairness is essential. While it is impossible to stamp out every piece of false news and every deepfake video, it’s entirely possible to rein in the chaos.

Concludes Cahn: “It’s really difficult to have meaningful discussions and debates as a society when we can’t even agree on the underlying facts. When you consider that deepfake videos have the potential to deceive huge numbers of people, you wind up in a very dark place ... a place that could significantly disrupt society and create a great deal of instability.”

Further Reading

Li, Y. and Lyu, S.

Exposing DeepFake Videos by Detecting Face Warping Artifacts. Computer Vision Foundation, November 1, 2018, <http://bit.ly/2pbASwP>

Long, C., Basharat, A., and Hoogs, A.

A Coarse-to-fine Deep Convolutional Neural Network Framework for Frame Duplication Detection and Localization in Forged Videos, Computer Vision Foundation <http://bit.ly/33GP7ZL>

Güera, D. and Delp, E.J.,

Deepfake Video Detection Using Recurrent Neural Networks, 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 27-30 Nov. 2018, <https://ieeexplore.ieee.org/abstract/document/8639163>

Koopman, M, Macarulla, R., and Geradts, Z. Detection of Deepfake Video Manipulation, Proceedings of the 20th Irish Machine Vision and Image Processing conference, August 29-31, 2018, <http://bit.ly/2NGd7qh>

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

© 2020 ACM 0001-0782/20/1 \$15.00

► James Grimmelman, Column Editor

Law and Technology

Increasing Automation in Policing

Seeking the delicate balance between civil liberties and policing public safety.

WE KNOW HOW artificial intelligence works in our lives: it helps in picking movies, choosing dates, and correcting misspellings. But what does it mean in policing? Is AI replacing traditional police tasks? Does the police use of AI present novel challenges? Should increasing police reliance on AI concern us? The answer to these questions is “Yes.” In the past decade the increasing reliance by police on artificial intelligence tools raises questions about how to strike the right balance between public safety and civil liberties.

Think of policing and you are likely to imagine a uniformed patrol officer scanning the environment for suspicious activity. The most powerful tools an officer once possessed were a gun, experience, and training. But new technologies are changing the way the police approach the streets. Automated license plate readers that identify hundreds of plates a minute are commonplace. The Chicago Police Department uses

an algorithm that identifies which city residents may be at especially high risk as perpetrators or victims of gun violence.¹ The police in Fresno, CA, piloted an alert system that tells an officer whether the driver the police officer just pulled over to the side of the road poses a threat.⁴ To this list we can also add facial recognition, suspect profiling, and financial anomaly detection.

Policing has always relied upon large amounts of information. But the scale and speed of its processing is different.

These technologies are transforming the police. There is the sheer amount of data now potentially available to the police, including all our online activity, digitized analog information, and our movements through space and time. And artificial intelligence transforms this data into actionable predictions and identifications. Policing has always relied upon large amounts of information. But the scale and speed of its processing today is different, and therefore warrants new scrutiny.

We might say policing is becoming increasingly automated.² The identification of suspicious activity—a skill we typically associate with police officers—can increasingly be handed off to artificial intelligence. Just as companies use AI to identify bad credit risks and good employment prospects, police departments are using these tools to figure out which people and places they think deserve scrutiny. Dozens of police departments, for instance, use PredPol, which uses a machine learning algorithm to predict those 500-by-500 foot sections of the city where



crime is more likely to happen.^a We live immersed in a world of scores and predictions—we should not be surprised the police do, too.

But when the police turn to artificial intelligence, we have far different concerns. After all, the police can stop and question even the unwilling, and perform searches and seizures that can begin the criminal process. And in a democratic society, we expect accountability and oversight over these government actors who have so much power over our lives. In the 20th century, that oversight could have been as simple as a bystander reporting potentially abusive behavior. Even the resource limitations of the police themselves once served as a potent check; it is impossible for most police departments to conduct around-the-clock surveillance of the population.

Artificial intelligence removes these checks. Technological tools powerful enough to gather every bit of available data around us and to make inferences about us as a result do what no human

police department could ever do. Every purchase, trip, online post, and more can be endlessly identified, sorted, and combined cheaply. In this sense, artificial intelligence vastly expands the potential pool of people and activities the police can watch.³

AI also allows policing to be less visible. The unrelenting collection of information is made possible because of both the digital trails we leave online and the sensors that capture all our physical world selves. Neither of these things requires the presence of the police. This poses unique challenges for how we regulate policing.

In criminal investigations, police must abide by the Fourth Amendment's prohibition on unreasonable searches and seizures. The Supreme Court has adapted its interpretation of the Fourth Amendment as the world has changed, but two core concepts have shown themselves to be particularly ill-equipped to address the transformations in policing.

First, since the 1960s, the Supreme Court has reiterated that what we knowingly expose to the public is not protect-

ed by the Fourth Amendment. For instance, we have no Fourth Amendment protection over our physical characteristics. We know our hair color, eye color, and other features are there for the world to see, so we can hardly expect special protection for this information. The Supreme Court has said the same of our movements on the public roads.

In today's world that legal idea is more complicated. Sure: a police officer's quick glance at your face may not raise concerns, but what about a hundred officers, or a thousand officers doing the same? What if those thousand officers were replaced by cameras equipped with facial recognition? Then your knowingly exposed self can be mapped in space and time: a map that would provide the government with all kinds of sensitive information, such as your religious affiliation or your political leanings. Yet, the conventional view is that no matter whether the government has taken one or a thousand snapshots of your face, you have given up your privacy rights.

Closely related to the idea of voluntarily exposed information is what is known as the third-party doctrine.

^a See <https://www.predpol.com/how-predictive-policing-works/>

The Supreme Court has long recognized that you lose Fourth Amendment protection in information you provide to third parties, such as banks and phone companies. In a 1979 decision, the Court held that the Fourth Amendment did not apply when the government intercepted the numbers dialed out by a suspect by installing a pen register at the phone company. Once handed over to a third party, that information lost the protection that would normally have required the police to obtain a warrant beforehand.

The phone numbers dialed out by landlines in the 1970s are a far cry, though, from our relationship to data today. It is impossible to live a normal life now without providing all kinds of information to third parties. Indeed, both our heavy reliance on the Internet and the Internet of Things means we are constantly streaming information to third parties.

Despite the dramatic changes in the technologies used by everyone, including the police, these legal concepts about knowing exposure and third parties remain robust parts of Fourth Amendment law. What then, can we expect from the courts as the police rely even more on artificial intelligence?

It turns out the U.S. Supreme Court has already hinted at how it might one day approach the question. These hints come from an unlikely source: the Court's 2018 decision in *Carpenter v. United States*.^b *Carpenter* is not a case about artificial intelligence. It involves the investigation of a string of cell-phone store robberies in the Midwest. Looking for evidence connecting Timothy Carpenter to the crimes, FBI agents asked his wireless services providers for his phone's cell-site location information during the times of the robberies. The FBI eventually received more than 12,000 location points that showed Carpenter's phone—and by implication Carpenter himself—near the robberies during the times they occurred.

The Supreme Court ultimately decided in Carpenter's favor and ruled the collection of this location information amounted to a "search" under the Fourth Amendment. That conclusion meant the government should have obtained a warrant before collecting the

b 138 S. Ct. 2206 (2018).

Artificial intelligence vastly expands the potential pool of people and activities the police can watch.

data. What was remarkable about the decision was the Court's expansion of Fourth Amendment protection to information that was held by Carpenter's wireless carrier, not Carpenter himself. Instead, the Court focuses in *Carpenter* on the nature of information sought: "the qualitatively different category of cell-site records."^c

This much has been commented upon widely. But the Court also said more in *Carpenter* that has implications for artificial intelligence: it also focused on three distinctive characteristics of the policing involved.^d First, what concerned the *Carpenter* majority was a policing technology that was both superhuman and cheap. Unlike the "nosy neighbor who keeps an eye on comings and goings," the technology used by the police was "ever alert, and [its] memory is nearly infallible."^e Few practical limitations exist when police can rely on "tireless and absolute surveillance methods."^f

Second, the Court noted that with the vast amount of data collected all of the time, "police need not even know in advance whether they want to follow a particular individual or when."^g In Carpenter's investigation, the government was able to "access each carrier's deep repository of historical location information" "[w]ith just the click of a button."^h This passive form of surveillance vastly expands the power of the police.

Finally, the way the government collected information in *Carpenter* rep-

c 138 S. Ct. at 2216.

d These observations are adapted from E.E. Joh, "Artificial Intelligence and Policing: Hints in the Carpenter Decision," *Ohio State J. Crim L.* 16, 281 (2019).

e 138 S. Ct. at 2219.

f 138 S. Ct. at 2218.

g 138 S. Ct. at 2218.

h 138 S. Ct. at 2218.

resented a decreasing reliance on human skill in favor of automation. As the Supreme Court itself observed, no one in Carpenter's position—anyone with a cellphone—could escape the "inescapable and *automatic* nature of its collection."ⁱ This represents a change in the scale and scope of policing tasks that augment, not just supplement, what the police do.

The government's case against Carpenter involved plotting points on a literal map for a jury, but the reasoning of the Court's opinion points to concerns that could easily be applied to the police use of artificial intelligence. The Court was concerned about tools that had extended beyond "augmenting the sensory faculties bestowed upon [the police] at birth."^j Tools that are "remarkably easy, cheap, and efficient compared to traditional investigative methods"^k merited new ways of applying the Fourth Amendment's protections.

To be clear: *Carpenter* is not a case explicitly about artificial intelligence. And the Supreme Court was adamant about trying to limit its decision to the collection of cell site location information. But that self-conscious restraint is not likely to last. In describing those changes in policing that called for a new Fourth Amendment approach, the Court happened to describe some of the key features of artificial intelligence tools being adopted by police departments: they are cheap, powerful, ubiquitous, automated, and invasive of privacy in ways that are novel and alarming. Whether the courts will embrace this potential approach to artificial intelligence tools in policing remains to be seen. ■

i 138 S. Ct. at 2223 (emphasis added).

j *United States v. Knotts*, 103 S. Ct. 1081, 1086 (1983).

k 138 S. Ct. at 2219.

References

1. Gornier, J. Chicago police use 'heat list' as a strategy to prevent violence. *Chicago Tribune* (Aug. 21, 2013).
2. Joh, E.J. Automated policing. *Ohio St. J. Crim. L.* 15, 559 (2018).
3. Joh, E.J. The new surveillance discretion: Automated suspicion, big data, and policing. *Harv. L. & Pol'y Rev.* 10, 15 (2016).
4. Jouvenal, J. The new way police are surveilling you: Calculating your threat 'score.' *Washington Post* (Jan. 10, 2016).

Elizabeth E. Joh (eejoh@ucdavis.edu) is a Professor of Law at the University of California, Davis School of Law, USA.

Copyright held by author.



Michael A. Cusumano

DOI:10.1145/3372918

Technology Strategy and Management

‘Platformizing’ a Bad Business Does Not Make It a Good Business

Transaction platforms link third-party applications and services providers with users.

UBER AND LYFT, as well as Airbnb, WeWork (We Co.), and other sharing-economy startups, offer valuable services, albeit with different levels of financial success (see “The Sharing Economy Meets Reality, *Communications*, January 2018). What most of these ventures have in common is they function as *transaction platforms*. That is, they bring together two or more market sides to exchange information, goods, or services, including advertisements. The businesses can grow rapidly through the power of network effects whereby one market actor (for example, sellers) attracts another side (for example, buyers) in a self-reinforcing positive feedback loop. The more populated one side becomes, the more value and participation we see on the other side. Transaction platforms contrast with *innovation platforms*, such as Microsoft Windows, Apple iOS, Google Android, or the Facebook and WeChat APIs. These foundational products or technologies generate network effects by linking users with third-party providers of applications and services.⁵

All platforms connect multiple market participants and can get big fast if they generate strong network effects and do not have a lot of digital or conventional competition. But there



are significant differences in business models. Microsoft and Apple mostly sell products. Google gives away software and digital services but then sells advertisements. Airbnb matches people with rooms for rent to possible renters and charges a fee when a match occurs. WeWork is really a “one-sided company platform” in that it leases office space and then resells it on short-term arrangements.

The most valuable publicly listed firms in the world today—Microsoft,

Apple, Amazon, Alphabet-Google, Facebook, Alibaba, and Tencent—are all “hybrids” that combine transaction and innovation platforms. Several colleagues and I recently measured their performance, comparing the largest 43 publicly listed platform companies to 100 of the largest firms in the same businesses over a 20-year period. They all had comparable annual revenues in 2015 (the year we compiled our list) of between \$4.3 billion and \$4.8 billion. But platforms achieved these sales with half the

Median values for Forbes Global 2000 industry control sample and platforms, 1995–2015.

Variable	Industry Control Sample	Innovation and Transaction Platforms
Number of Firms	100	43
Sales (Million\$)	\$4,845	\$4,335
Employees	19,000	9,872*
Operating Profit %	12%	21%*
Market Value (Million\$)	\$8,243	\$21,726*
Mkt Value-Sales Multiple	1.94	5.35*
Sales Growth vs. Prior Year	9%	18%*
Observations	1,018	374

Source: Michael A. Cusumano, Annabelle Gawer, and David B. Yoffie, *The Business of Platforms* (2019), p.23.

Notes:

* Differences significant at $p < 0.001$ for Industry Sample vs. Platforms using two-sample Wilcoxon rank-sum (Mann-Whitney) test.

Mkt Value-Sales Multiple = ratio of market value compared to prior year sales.

Average of 13 years of data for 18 innovation platforms and five years for 25 transaction platforms.

number of employees, generated nearly twice the operating profits, had market values more than twice as high, and were growing about twice as fast (see the accompanying table).

Not surprisingly, investors and entrepreneurs keep looking for the next blockbuster platform. In fact, we estimate 60% to 70% of the billion-dollar private “unicorn” companies are platform ventures.⁵ But “platformizing” (creating a platform) in a “bad” business (an industry with low profit margins due to high costs, low entry barriers, or other structural factors) does not make it a good business. Profits depend on supply and demand, which impact prices, as well as economies of scale and scope or other efficiencies, which impact costs. Moreover, a business that delivers a physical good or service is unlikely to generate the same high profits as a platform that sells digital goods such as software, content, or advertisements, which have close to zero marginal costs.

Let’s look more carefully at Uber, the most valuable sharing-economy platform. It went public in May 2019 and has since seen its stock price drop sharply, even though Uber remains a compelling and convenient service. Uber users can summon a ride, usually within minutes, track the driver’s progress, and then pay, all via a smartphone app. Uber and other ride-sharing companies also usually charge less than taxis to attract riders. Each additional driver adds value because transportation options for riders increase. Uber also owns no cars, so its capital costs are minimal. Nevertheless, Uber lost \$4.5 billion in 2017 and \$1.8

billion in 2018 (with income boosted from selling some overseas operations). For the first six months of 2019, Uber reported sales of \$6.2 billion and operating losses of \$6.5 billion, including costs related to the IPO.¹² Uber has been raising prices and cutting staff, yet operating losses in the billions are expected to continue.³ Why does a company with such a valuable service lose so much money?

First, Uber (like Lyft and WeWork) is not really a digital business. Only the transaction process is digital. Transporting people or goods is a physical service with potentially high costs. In this case, Uber must pay a lot of money to find drivers, and it keeps prices low to attract riders. In addition, Uber’s economies of scale and scope are primarily local since each area needs its own drivers.

Second, a platform rather than a traditional product or service company should bring together two or more market sides, rely on network effects to grow, and then charge for a product or transaction fee without the expense of having the same number of employees or large capital investments as a conventional business. Uber’s platform works best if the driver side is heavily populated so that customers can always get a ride quickly when they need one. However, Uber drivers frequently quit because of long hours and low compensation, with no employee benefits since they are independent contractors.

According to data released prior to its IPO, Uber drivers were quitting at a rate of 12.5% per month and the company had to pay approximately \$650 to hire each new driver. With at least three mil-

lion drivers, this meant Uber had to find 375,000 new drivers every month and replace all its drivers every eight months. Driver costs before commissions on rides were almost \$250 million per month or nearly \$3 billion per year.² In short, Uber has been massively subsidizing both sides of its platform—large payments to drivers as well as low fares to riders. This is a great way to lose a lot of money.

Lyft loses less money (\$911 million in 2018 on sales of \$2.2 billion, and \$1.14 billion on sales of \$776 million in the first quarter of 2019, including IPO-related stock-compensation charges) mainly because it is smaller.⁴ WeWork lost \$1.9 billion in 2018 on \$1.8 billion in revenues and postponed its IPO after losses of \$690 million on revenues of \$1.5 billion in the first six months of 2019.⁶ By contrast, Airbnb, which plans to go public in 2020, consistently reports profits or at least a positive cash flow.⁹ Why? Because Airbnb can charge both sides of its platform. It does not pay people to list rooms or to rent real estate on its supply side, and it allows renters to charge market prices. Uber pays drivers even when they do not have riders. WeWork pays for real estate even when it does not have renters (and, like Uber, it generally has charged well below the market price in order to grow quickly).

Why would investors put their money into businesses that consistently lose money? In the case of Uber, investors must be hoping for a “winner take all or most” outcome. Once competitors have disappeared, then Uber can raise prices to riders and reduce payments to drivers. But Uber already has 70% of the U.S. ride-sharing market and it still does not have enough market power or operating efficiencies to turn a profit. At least in part this is because of high driver turnover and the fact that, in most cities, there are still too many transportation options (including people using their own vehicles). Uber makes a profit mainly in a few cities where taxis and other transport options are limited and expensive.

Uber has also attracted investors with the promise it will become the “Amazon of transportation” and move into nearly every transportation segment.¹ But expansion into more “bad” businesses simply means the bigger the platform gets, the more money it will lose. Amazon took many years to make a profit and

funded much of its growth through cash flow from its global online store and then a marketplace matching buyers and sellers, with high fees for fulfillment services and advertising. Still, most of Amazon's profit (nearly 60% in 2019) comes from a highly profitable digital service and innovation platform: Amazon Web Services (see "The Cloud as an Innovation Platform for Software Development," *Communications*, October 2019). Transporting food is slightly more profitable than transporting people.² But Amazon tried and failed to make money in food transport and recently closed down Amazon Restaurants.¹¹ Deliveroo is the market leader in the U.K. and Europe but it also has failed to earn a profit in food transport.¹⁰

Another option for Uber is to get rid of drivers (its main cost) by investing in driverless vehicles. This strategy may work someday in some locations. But Uber will most likely run out of cash long before driverless vehicles become safe and commonplace. Moreover, someone still has to own the vehicles. Let's say Uber buys two million vehicles to replace its three to four million drivers. Even at \$50,000 per vehicle, that would be a massive expense of \$100 billion. Then there are maintenance, recharging, replacement, or leasing costs. The economics may still be better than constantly subsidizing drivers and riders. However, there will also likely be more competition. Every major automaker in the world, as well as Lyft and Google Waymo, are in or planning to enter the markets for ride-sharing and driverless vehicles, with the goal of "selling miles, not cars."⁷

Uber and Lyft should stay with the platform strategy rather than own or lease their vehicles, but their ongoing losses may force them to get smaller before they can get bigger again. Ride-sharing platforms should focus on the few cities where taxis and transportation options are expensive and in short supply, and then expand gradually into other geographies and transportation segments, such as partnerships with public transportation authorities to bring people from their homes to mass-transit facilities. Uber already is exploring such partnerships, and it has sold several unprofitable overseas subsidiaries. It also seems to be making some money matching shippers with truckers in a new marketplace venture. However,

Expansion into more "bad" businesses simply means the bigger the platform gets, the more money it will lose.

there is nothing to stop more companies and entrepreneurs from entering the transportation business. Taxi companies can also develop their own smartphone apps and expand their operations and partnerships (see "How Traditional Firms Must Compete in the Sharing Economy," *Communications*, January 2015). As for WeWork, it must raise prices much more to cover its costs, and that will depress demand. Landlords are also now hesitant to lease properties to WeWork, putting the whole business at risk.⁸ Meanwhile, Airbnb is likely to continue its strong business model, although there is nothing to stop traditional hotel companies from aggressively entering roomsharing—if it is a "good" (that is, profitable) business.¹³ □

References

1. Bond, S. Uber aims to be the 'Amazon of transportation.' *Financial Times* (Oct. 25, 2018).
2. CB Insights. How Uber makes money. (2018), 6–23.
3. Clark, K. Uber lost more than \$5B last quarter. *Techcrunch* (Aug. 8, 2019).
4. Conger, K. Lyft's first results after I.P.O. show \$1.14 billion loss. *New York Times* (May 7, 2019).
5. Cusumano, M.A., Gawer, A., and Yoffie, D.B. *The Business of Platforms: Strategy in the Age of Digital Competition, Innovation, and Power* (2019), 18–20.
6. Farrell, M. WeWork IPO filing reveals huge revenue and losses. *Wall Street Journal* (Aug. 14, 2019).
7. Gindrat, R. Automakers envision a business model for AVs: Selling miles, not cars. *Axios.com* (Apr. 17, 2019).
8. Grant, P. and Morris, K. Virtually no one will lease to WeWork. That's a drag on NYC's office market. *Wall Street Journal* (Sept. 29, 2019).
9. Prang, A. Airbnb plans to go public next year. *Wall Street Journal* (Sept. 19, 2019).
10. Satariano, A. Deliveroo takes a kitchen sink approach to food apps. *Bloomberg.com* (Jan. 15, 2018).
11. Soper, T. Amazon to shut down its Amazon restaurants business in the U.S. *Geekwire* (June 10, 2019).
12. Uber Technologies, Inc. *Form 10-Q* (June 30, 2019).
13. Weed, J. Blurring lines, hotels get into the home-sharing business. *New York Times* (July 2, 2018).

Michael A. Cusumano (cusumano@mit.edu) is a professor at the MIT Sloan School of Management doing research on computing platforms for business and coauthor of *The Business of Platforms: Strategy in the Age of Digital Competition, Innovation, and Power* (Harper Business, 2019).

Copyright held by author.

Calendar of Events

Jan. 6–8

Group '20: The 2020 ACM International Conference on Supporting Group Work, Sanibel Island, FL, Sponsored: ACM/SIG, Contact: Louis Barkhus, Email: barkhus@itu.dk

Jan. 13–16

ASPDAC '20: 25th Asia and South Pacific Design Automation Conference, Beijing, China, Contact: Huazhong Yang, Email: yanghz@tsinghua.edu.cn

Jan. 19–25

POPL '20: The 47th Annual ACM SIGPLAN Symposium on Principles of Programming Language, New Orleans, LA, Sponsored: ACM/SIG, Contact: Brigitte Pientka, Email: bpientka@cs.mcgill.ca

Jan. 20–21

CPP '20: 9th ACM SIGPLAN International Conference on Certified Programs and Proofs, New Orleans, LA, Sponsored: ACM/SIG, Contact: Jasmin Christian Blanchette, Email: j.c.blanchette@vu.nl

Jan. 21

PLMW@POPL '20: Programming Languages Mentoring Workshop at PLMW, New Orleans, LA, Sponsored: ACM/SIG, Contact: Justin Hsu, Email: email@justinh.su

Jan. 27–31

AFIRM '20: ACM SIGIR/SIGKDD Africa School on Machine Learning for Data Mining and Search, Cape Town, South Africa, Sponsored: ACM/SIG, Contact: Tamer Mohamed Elsayed, Email: telsayed@qu.edu.qa

Jan. 27–30

ACM FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, Sponsored: ACM/SIG, Contact: Carlos A. Castillo, Email: chato@chato.cl

Historical Reflections

Von Neumann Thought Turing’s Universal Machine was ‘Simple and Neat.’ But That Didn’t Tell Him How to Design a Computer.

New discoveries answer an old question.

COMPUTER ARCHITECTURE AND theoretical computer science have different roots. Architecture grew out of projects begun in the 1940s to design high-speed electronic computing machines able to complete elaborate sequences of operations without human intervention. Its symbolic founding text is John von Neumann’s 1945 “First Draft of a Report on the EDVAC,”^a though early computer builders relied more directly on a series of lectures and reports disseminated the next year. Theoretical computer science grew from an academic desire to theorize about the fundamental characteristics and capabilities of automatic computing. The theoretical foundation of computer science was laid during the late-1950s and 1960s using intellectual materials scavenged from different fields. Alan Turing’s 1936 paper, “On Computable Numbers, With an Application to the Entscheidungsproblem” provided the most prominent building block. In it, Turing introduced a definition of com-

putability based on the operations of imaginary automata.

Popular imagination only has room for one “great man” per invention, and Turing’s prominence in computer science has created a market for arguments that he must therefore have invented the computer itself. The fact that Turing and von Neumann knew each other has led to considerable speculation about the possible influence of Turing’s paper on von Neumann’s architectural approach. Yet no hard evidence has yet come to light showing that von Neumann had read or appreciated Turing’s paper during the crucial period from early 1945 to mid-1946.

In this column, we present newly discovered archival evidence: the text of three lectures on “High Speed Computing” written by von Neumann in 1945.^b This demonstrates von Neumann was well aware of Turing’s work while he worked to define modern computer architecture. It also suggests von Neu-

mann found Turing’s work interesting as a model of computability but not as a source of ideas on computer architecture, a formalism with which to describe the design of a computer, or a way of justifying the construction of actual computers by pointing to their “universal” capabilities.

Von Neumann and Turing

Between 1943 and 1945, a team working at the University of Pennsylvania’s Moore School of Electrical Engineering designed and built ENIAC, the first programmable electronic computer. Before it was finished, they partnered with John von Neumann to propose and begin to design a successor. By April 1945, von Neumann had prepared a long document outlining a new approach in which programs were coded, along with the data they manipulated, as numbers in a large, addressable, and rewritable electronic memory. This “First Draft of a Report on the EDVAC” was widely distributed, as were the notes of a lecture series held at the Moore School in 1946 and reports issued by von Neumann’s new team working at the Institute for Advanced Studies. These inspired the designers of the first generation of mod-

^a See <http://bit.ly/2LbZtcl>

^b J. von Neumann, “High Speed Computing,” n.d. circa 1945, in the Herman Heine Goldstine Papers, American Philosophical Society, Philadelphia, PA (hereafter HHG-APS), box 49, folder “JvN undated #5”.

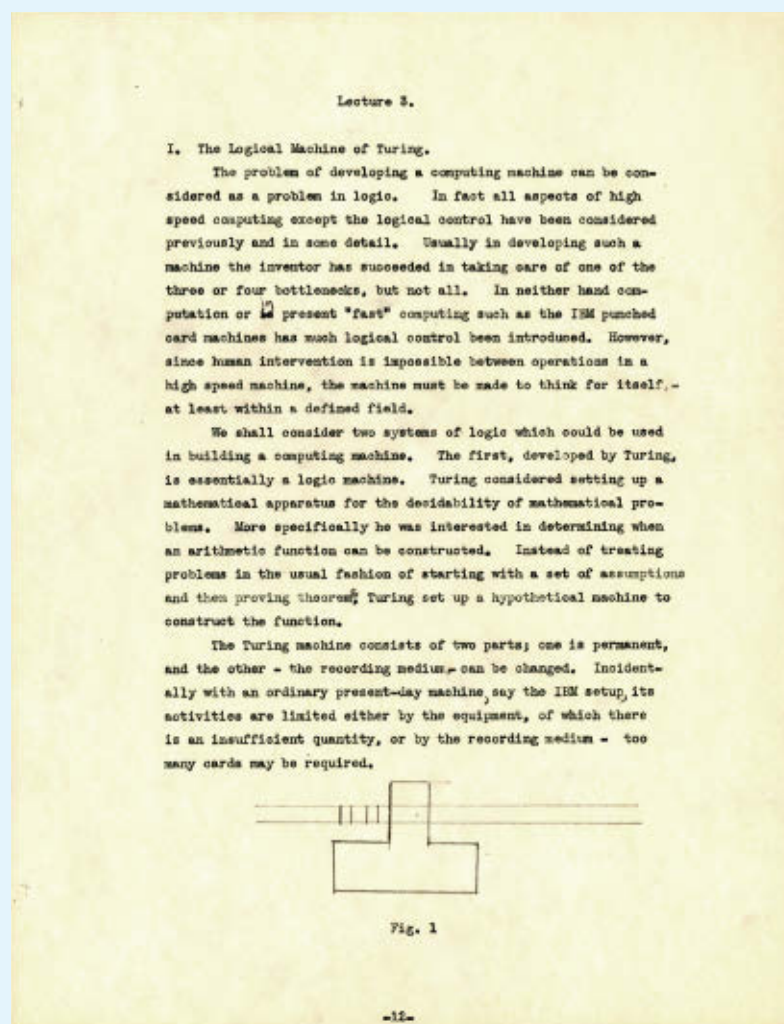
ern electronic computers, including those at the universities of Manchester and Cambridge, and Turing's own design for the ACE at the National Physical Laboratory. During the 1950s the term "stored-program computer" emerged to describe what had previously been called "EDVAC-like computers." This term mutated into an abstract "stored program concept" taken to characterize the essence of these machines, making it easy to assume that modern computers are the embodiment of a single novel idea.⁹

Some have suggested von Neumann took this single novel idea from Turing. The plausibility of this hinges on the philosophical question of what kind of innovation the modern computer was. Martin Davis asserted that Turing devised the "stored program concept" in his 1936 paper, implying that the invention of the computer was more than anything else an advance in mathematical thinking. This is clear in the title of his book: *Engines of Logic: Mathematicians and the Origin of the Computer*.⁶

Such claims have drawn attention to the relationship between Turing and von Neumann. Although the *First Draft* was (as its name suggests) not a finished publication it did include one citation, to a 1943 paper by Warren McCulloch and Walter Pitts on a connection between mathematical logic and neuron nets.¹³ That paper in turn cited Turing's 1936 paper and referred explicitly to his machine-based definition of computability, raising the question of whether von Neumann had read Turing's paper prior to his work on the First Draft. Andrew Hodges looked for direct evidence of this when researching his landmark biography of Turing.¹⁰ Von Neumann was indisputably aware of Turing, having written a reference in 1937 in support of the fellowship that allowed Turing to spend a year at Princeton University, in close proximity to von Neumann's own base at the Institute for Advanced Study. Hodges noted that the reference praised Turing's work in two areas of mathematics, but not with the 1936 paper. Hodges was able to contact one of von Neumann's Los Alamos collaborators, Stanislaw Ulam, who expressed a suspicion that von Neumann had read the paper by 1939 but could "not answer for sure."^c

c <https://www.turing.org.uk/sources/vonneumann.html>

Figure 1. Von Neumann introduced Turing machines at the start of his third 1945 lecture on "high speed computing machines."



Jack Copeland has made the stronger assertion that von Neumann himself "repeatedly emphasized that the fundamental conception was Turing's." He had merely "placed Turing's concept into the hands of American engineers." Copeland quoted a short passage from a November 1946 letter von Neumann sent to Norbert Wiener, the founder of cybernetics, showing *awareness* of Turing's demonstration of a universal machine. He quoted at length passages from lectures delivered in 1948 and 1949 showing von Neumann's admiration for Turing's paper.^d

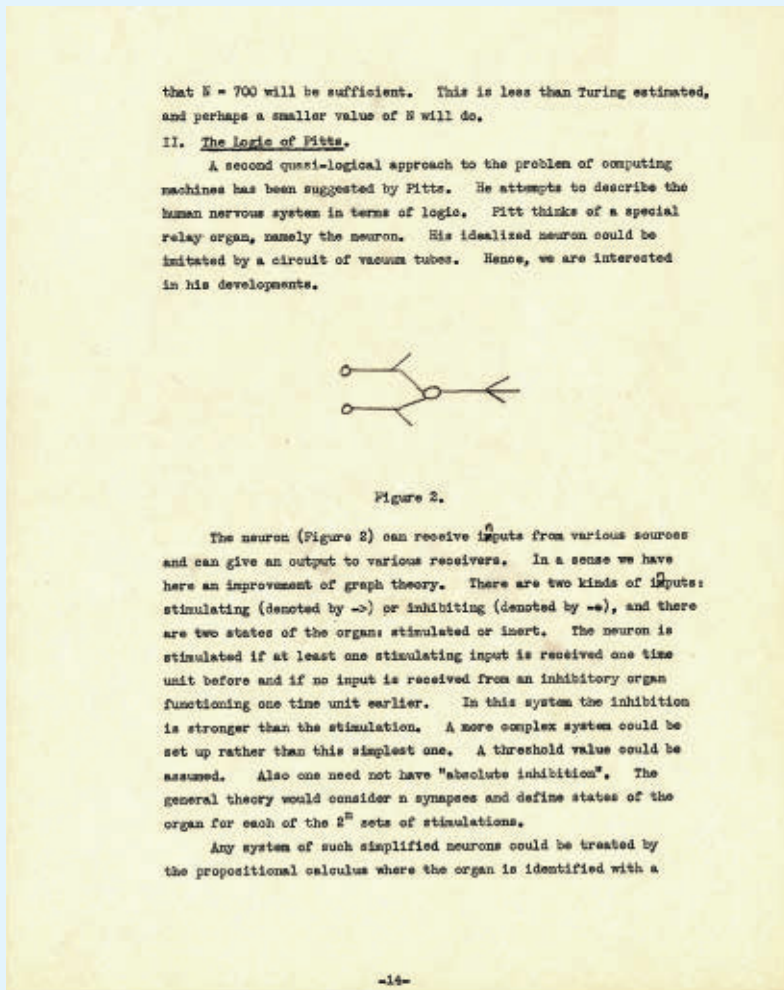
We do not share Copeland's interpretation of these sources and see no

d Copeland has made similar points in several venues, but for accessibility we are working here from his online publication "Turing, Father of the Modern Computer" with Dianne Proudfoot, <http://www.rutherfordjournal.org/article040101.html#chapter06>.

solid evidence that von Neumann ever credited Turing with having inspired the design of EDVAC. Our own position was summed up in the title of an earlier *Communications* Historical Reflections column, "Actually, Turing Did Not Invent the Computer."⁸ The EDVAC design centered on what is often called the "von Neumann architecture" in which instructions were retrieved from memory, decoded, and executed using a single connection to main memory and a single arithmetic unit.^{9,15} We cannot point to any important features of this architecture that von Neumann might have derived only from Turing's paper.

Neither did the world need to read Turing to appreciate the potential of automatic computers. Before writing the First Draft von Neumann had visited groups at Harvard, Bell Labs, and the University of Pennsylvania that

Figure 2. After discussing Turing, von Neumann moved on to explain the use of abstract neurons to represent digital switching circuits.



had initiated computer-building projects in complete ignorance of Turing's work. But such claims underline the importance of finding out what, if anything, von Neumann felt in 1945 about the relevance of Turing machines to computer-building projects.

Before proceeding to answer that question, we should acknowledge another controversy. Even those historians of early electronic computing who see the First Draft as a crucial and original document have disagreed about whether its key ideas should be credited to von Neumann or to the original ENIAC team. It is common to read claims that von Neumann was merely writing up ideas formulated by J. Presper Eckert and John Mauchly. Those who give full credit to the ENIAC design team often focus on the computer as a product of innovations in electrical engineering. Those, including Arthur Burks, an-

other of the ENIAC team, who felt von Neumann made a crucial contribution, point to his abstraction from engineering details to produce the first coherent proposed architecture for EDVAC. Surviving evidence is inconclusive, but in our book *ENIAC in Action* we did our best to plausibly divide credit for different aspects of the EDVAC design.⁹

"High Speed Computing," by John von Neumann

After the completion of *ENIAC in Action*, one of us (Priestley) returned to the archive of Herman Goldstine's papers at the American Philosophical Society in Philadelphia. Goldstine had been von Neumann's closest collaborator within the ENIAC group, and chose to have the First Draft typed up and widely distributed.

Hiding in Goldstine's papers was the typescript of a series of three short

lectures by von Neumann on "High Speed Computing." They provide a more complete and explicit discussion of the connection of Turing's 1936 paper to the design of actual computers than the documents historians have previously relied upon. We believe they predate any prior documented reference by von Neumann to Turing's 1936 paper. Through not dated, from internal evidence and comparison with other documents from the period, we conclude the text dates from mid-to-late 1945; we justify that assertion below.

As shown in Figure 1, Lecture 3 began with the words "The problem of developing a computing machine can be considered as a problem in logic," which in this context referred to "logical control" or the automatic sequencing of operations. Von Neumann's approach to computer architecture was deeply shaped by his background in logic. His plan for EDVAC was a simpler, cleaner, and more practical design than any of the earlier attempts to build a general-purpose automatic computer.

The text shows that von Neumann knew Turing's 1936 paper and fully appreciated the significance of the universal machine described in it. He first described Turing's machine concept and its connection to the question of effective calculability: "We shall consider two systems of logic which could be used in building a computing machine. The first, developed by Turing, is essentially a logic machine. Turing considered setting up a mathematical apparatus for the decidability of mathematical problems. More specifically he was interested in determining when an arithmetic function can be constructed. Instead of treating problems in the usual fashion of starting with a set of assumptions and then proving theorems, Turing set up a hypothetical machine to construct the function.

"The Turing machine consists of two parts; one is permanent, and the other—the recording medium—can be changed ... It is composed of a long paper band with symbols recorded and an apparatus to sense these symbols, put on new ones, and erase old ones. There are a finite number of states of the machine. Let the range of the indication i of those states be $i = 1, 2, \dots, N$. Let the state of a square of tape be j , where $j = 1, 2, \dots, M$. At every moment the machine inspects the tape and then

does something. That is, from every state (of machine and tape) (i,j) , they move to another (i',j') and the tape is moved right or left by one unit. If we think of this in terms of graphs we have an arrow from one point to another and a sign.”

This description is easier to follow than Turing’s own, and von Neumann’s description of state transitions in terms of a graph anticipates the later development of state transition networks as a visual model for finite state automata.

Von Neumann was clear on the limited usefulness of this model of computation, which had been designed to prove a theoretical point about mathematics, as a guide to the capabilities of actual automatic computers. “Suppose this machine is provided with a tape with a finite number of symbols, the question is whether there is a permanent apparatus which will solve the problem by the Turing method. There is a finite number of different steps that the machine can do. However, the machine can do steps an infinite number of times. Hence, a problem whose solution can be broken down into a finite or infinite number of parts, but involving only a finite number of different steps, can be done on this machine. Here the analogy to a high-speed computing machine breaks down, for one cannot wait for the machine to go all eternity for his answer.”

Computer builders, von Neumann realized, must be concerned more with practical than theoretical limits to computability.

Universal Machines

Before moving on to the next topic, von Neumann then described Turing’s universal machine concept: “A Turing machine is defined as ‘adequate’ for a particular problem if it can be solved by means of a suitable tape and apparatus. A ‘universal’ machine is one which can construct any arithmetic function that can be done by a particular Turing machine. Common sense might say that a universal machine is impossible, but Turing proves that it is possible. The idea of a universal machine is simple and neat. To build this machine one decides on a code to describe each particular Turing machine. Then one puts the definition of each Turing machine to a tape. The new machine reads the definition of a

Computer builders, von Neumann realized, must be concerned more with practical than theoretical limits to computability.

Turing machine and then imitates it.”

He proceeded to make the first contribution to what later became a popular game of identifying the smallest number of states and symbols a Universal Turing Machine could operate with.

Later popularizers have focused on the universal machine as the heart of Turing’s paper, to the extent that many people believe that a “Turing Machine” is necessarily a universal machine. It takes a conscious effort to notice how unusual this focus was in 1945. Turing’s paper had been little cited, and the attention it did receive, most famously a short review by Alonzo Church that introduced the phrase “Turing Machine,” treated it as a contribution to work on decidability and ignored the universal machine part of the paper.⁴ In that context the universal machine was almost a diversion, developed in more detail than necessary to prove Turing’s mathematical argument. We are not aware of any earlier recapitulation of the universal machine concept by any author.^e Even Martin Davis, who in 1958 was one of the first to note the potential of the universal machine as a model for what could be calculated by real computers, nevertheless advised readers that the page he devoted to the topic was “a digression and may be omitted without disturbing continuity.”⁵

Neurons and Cybernetics

Von Neumann then moved to the lecture’s main topic, “The Logic of Pitts,”

spending more than twice as much space on neuron networks as he had on Turing machines. This reflects von Neumann’s deep involvement in the establishment of what would soon be called cybernetics. In January 1945, von Neumann, Howard Aiken, and Nobert Wiener convened a meeting of people “interested in communication engineering, the engineering of computing machines, the engineering of control devices, the mathematics of time series in statistics, and the communication and control aspects of the nervous system.” Wiener himself had been working during the war with electronics, trying to produce an automatic control unit for anti-aircraft guns able to predict the trajectory of enemy pilots.⁷

Historians remember the January 1945 event as one of several that laid the groundwork for the emergence of cybernetics a few years later.¹² The meeting had a strong focus on computation and applied mathematics: as well as McCulloch and Pitts and other leading brain researchers, invitees included mathematicians and statisticians with links to practical and automated computing, such as Herman Goldstine and Leland Cunningham of the Ballistics Research Lab. Immediately afterward, participants were filled with excitement for follow up plans for collaborative research, including the establishment of a Teleological Society. Working groups were formed to explore the application of automatic computers to statistical problems and to differential equations.^f But when plans for a second teleological meeting faded, work shifted to the well-known series of “Macy Conferences” organized by McCulloch and others from 1946 onward. Von Neumann remained involved, but the series settled down with a mix of participants featuring fewer applied mathematicians and more high-profile participants from disciplines such as sociology, psychology, psychiatry, and anthropology.¹⁰ In 1948, Wiener published *Cybernetics: Or Control and Communication in the Animal and the Machine* which pushed the new way of thinking onto the pages of newspapers and magazines.

e We are deeply grateful to Andrew Hodges for his advice on this point. The spread of Turing’s ideas in the 1950s is discussed by Lisebeth De Mol in <https://plato.stanford.edu/entries/turing-machine/>.

f J. von Neumann to Aiken et al, Jan 12, 1945, in the John von Neumann Papers, Library of Congress, box 7, folder 14.

Figure 3. The neuron notation and design of this adder suggests the lecture was written after May 1945.

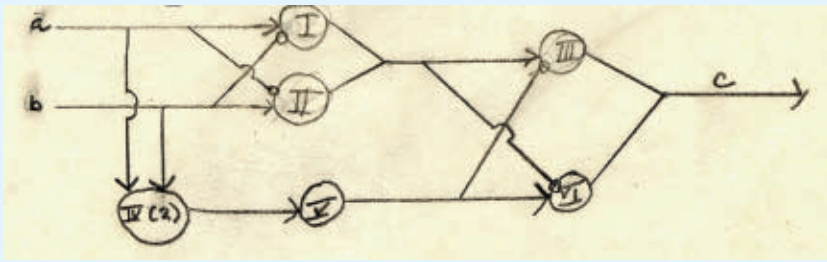
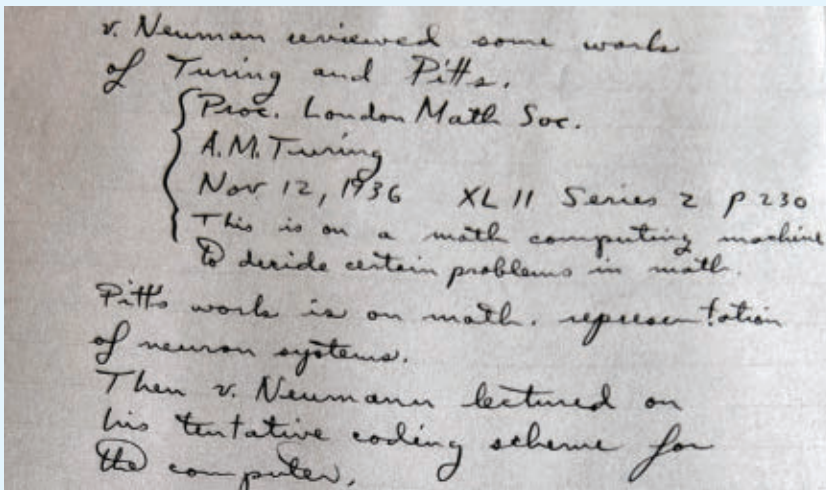


Figure 4. Calvin Mooers' notebook records topics covered during his meeting with von Neumann on October 28, 1945: Turing's 1936 paper, Pitts' mathematical representation of neurons, and von Neumann's work on instruction set design.



The interchangeability of organisms and mechanisms remained one of the central ideas of cybernetics.

Weiner had introduced von Neumann to the 1943 McCulloch and Pitts paper we mentioned earlier in this column. Work by Claude Shannon had already established an equivalence between digital circuits and expressions in propositional logic. McCulloch and Pitts went further, asserting that their “nets” of abstract neurons, coupled with “tapes” and suitable “scanners,” had equivalent computational capabilities to Turing machines and other computational agents: “If any number can be computed by an organism, it is computable by these definitions, and conversely.” Initial interest in “nervous nets” was thus quite different from the later shift to neural nets in artificial intelligence, which was motivated by a desire to avoid symbol processing and propositional logic rather than to implement them.

To the cyberneticians, digital control circuits, brains, and logical propositions were different ways of expressing the same relationships between inputs and outputs. The excitement of this idea runs through the First Draft, and is reflected in von Neumann’s use of biological language (EDVAC’s major components were called “organs” and its storage “memory”) and deployment of an abstract neuron-inspired notation to represent computing circuits (see Figure 2).

In the lecture von Neumann used the same notation to illustrate the applicability of “the logic of Pitts” to computer design. His examples showed simple configurations, including a chain of neurons for memory (“we would like to design a circuit which would learn and unlearn”), a binary adder, and a counter. The lecture stops abruptly, without getting to the issue of automatic (“logical”) control which von Neumann had earlier suggested was the major unsolved challenge. More space is given

over to the neuron notation than to anything else, and relatively little to electronic storage and its organization (a major topic in the First Draft).

Dating the Manuscript

The lecture contains several implicit references to ENIAC and EDVAC, evidence that von Neumann was writing for a broad audience at a time when ENIAC remained secret, that is, before its February 1946 press launch. As in the First Draft, he preserved a deliberate vagueness, presenting as theoretical possibilities things the ENIAC team had already proven experimentally. For example, he stated that a “fast” machine able to multiply two ten-digit numbers in 0.001 second was not yet a reality but was achievable with “existing objects of computing.” Ten decimal digits was the size of ENIAC’s standard numbers, placing an upper bound for the lecture date at some point before the full ENIAC’s first successful use in December 1945.

Noting the simplicity and generality of the examples in the lecture as opposed to the more complex networks presented in the First Draft, we originally suspected that the lectures contained von Neumann’s first experiments in using the neuron notation. A plausible venue for their delivery would then have been the January 1945 meeting of the nascent Teleological Society on the first day of which, according to Wiener, “von Neumann spoke on computing machines.”^g That would make the adder circuit in the lecture, which differs significantly from the First Draft version, an early, discarded design.

However, we then re-examined a letter in which Herman Goldstine sent von Neumann comments on the text of the First Draft, including a sketch of the design of “an adder that Pres [Eckert] and John M[auchly] are patenting.”^h Von Neumann’s adding circuit in the lectures (see Figure 3), closely resembles this sketch. Goldstine also suggested some changes to the notation, such as adding arrowheads to the lines

g Wiener, letter to Rosenblueth, 1945 Jan 24, quoted in S. J. Heims, *John von Neumann and Norbert Wiener: From Mathematics to the Technologies of Life and Death*. MIT Press, Cambridge, MA, 1980, 185–186.

h Goldstine to von Neumann, 15 May 1945 (HHG-APS box 21).

linking the neuron symbols. These appear in the lecture but not in the First Draft, the text of which was not altered before it was circulated in June.ⁱ These facts suggest that the lecture was written after von Neumann received Goldstine's letter in mid-May 1945 and used simple diagrams for pedagogical reasons only.

Thus we are confident the notes date from the summer or fall of 1945. This was a crucial period in von Neumann's work on computing, during which he continued to revise his EDVAC code and worked on a complex sort routine designed to test out the potential of the new approach to automatic computing.¹⁵ By the end of the year his attention had shifted to the IAS computer project, which settled on a revised and highly influential version of the EDVAC design. Combined with his multiple consultancies and contributions to IBM's early efforts in electronic computing, this reminds us that von

Neumann's contributions to early electronic computing go far beyond simply writing the First Draft, and were made possible by his movement between different groups and communities.¹

We then combed through what we could reconstruct of von Neumann's calendar to identify a possible venue for the lectures. He gave a talk with a similar title, on "High-speed computing devices and mathematical analysis," at the Canadian Mathematical Congress in June 1945, but Garrett Birkhoff recalled that the focus of that talk was on numerical methods and simulation in fluid dynamics.³

Calvin Mooers

The closest match to the lecture we found was in a memoir from mathematician Calvin Mooers, who was part of a computer building project at the Naval Ordnance Lab headed by John Atanasoff. Von Neumann was an initiator of and consultant to that project. Mooers recalled that "very early" in the project, which he joined in August 1945, the team met with von Neumann who "cordially received us, and then

jumped into an advanced (for us) logical discussion about the design of a computer using, as I recall the Pitts and McCulloch symbolism for neural connections." They were not "intellectually ready" for this, because the "language and concepts were not from electronics and circuits, which we might better have assimilated..."¹⁴

Mooers' diary and notebooks document several meetings with von Neumann, the earliest on August 29 when Mooers learned about ENIAC and plans for EDVAC.^j On October 28, Mooers and other members of the NOL team traveled to Philadelphia to visit ENIAC, and then moved on to a meeting at the IAS where "von Neumann reviewed some work of Turing and Pitts" before he "lectured on his tentative coding scheme for the computer." Mooers' notebook includes the citation for Turing's paper, with the comment "This is on a math computing machine to decide certain problems in math"; see Figure 4. On

i Arrow symbols are used in the First Draft on some connections to indicate delays. Goldstine proposed an alternative notation for this.

j "NOL Notebook 2," in box 27 of the Calvin R. Mooers Papers, Charles Babbage Institute, Minneapolis, MN.



Advertise with ACM!

Reach the innovators and thought leaders working at the cutting edge of computing and information technology through ACM's magazines, websites and newsletters.

Request a media kit with specifications and pricing:



Ilia Rodriguez
+1 212-626-0686
acmm mediasales@acm.org



ECSEE
European Conference
Software Engineering Education

18 and 19
June 2020

Seeon Monastery
Germany

Full Paper
Submission Deadline:
16 March 2020

www.ecsee.eu

Proceedings will be published in:



Published by ACM

his return to Washington, Mooers requested a photostat copy of Turing's paper from the library but, unlike the First Draft, he never recorded reading it.^k

Mooers' experience of von Neumann spontaneously launching into similar material in October builds our confidence in a mid-1945 date for preparation of the lecture text. Von Neumann gave many lectures and assisted many computing groups, undoubtedly recycling material between them. He obviously hadn't prepared the text specifically for this occasion, not least because it is coy about the details of projects he had already discussed with Mooers. The material von Neumann followed it with on that occasion, describing his new approach to logical control, would have been the obvious content for a "lecture 4" to solve the problem posed at the start of lecture 3.

Mooers did not reference Turing in his efforts to design a computer and instruction set. In March 1946, however, he did track down the work of McCulloch and Pitts and used the neuron notation and cybernetic terminology to sketch out plans for a "thinking machine." He discussed it with Pitts that summer, noting in his diary that he told him "how by use of a magnetic edvac type machine a device could be made which would trace through a nervous net. Showed him how a 'Turing Machine' (which it is) can be elaborated to do the job."^l Not long after Mooers' boss, John V. Atanasoff, ordered him to work on more useful matters.

Conclusion

Even before learning of the lecture notes on "High Speed Computing" we believed that von Neumann was almost certainly familiar with Turing's 1936 paper prior to beginning work on the First Draft. His later remarks showed that he fully understood its use as an abstract model of computation, and there was no reason to believe he suddenly developed this understanding after 1945. The new evidence confirms that von Neumann had read and fully understood Turing's paper around the

time he was creating and refining his abstract design of EDVAC.

More interestingly, the lecture gives us a sense of the contexts in which von Neumann did, and did not, find Turing machines relevant. They appear in a tutorial role, introducing an abstract model of computation to make a point about the equivalence of different possible computers with respect to the computations they could perform "given all eternity." That resembles the use of the concept by modern computer science popularizers, though its coupling with discussion of neurons and the creation of machines able to learn reminds us that for von Neumann, as for other founders of cybernetics, enthusiasm for theories of computation was bound up with much grander visions. The example of Mooers suggests that von Neumann successfully communicated this cluster of ideas to at least one recipient.

While Turing machines do not appear in von Neumann's work on computer architecture and logical control, they are prominent in his later work toward a "general and logical theory of automata."² His lecture at the 1948 Hixon symposium showed, in the words of historian William Aspray, that he "had in mind the McCulloch-Pitts and Turing contributions as the foundation for his new theory." Von Neumann developed these ideas further over the next few years, culminating in a major work on cellular automata and a series of lectures on the computer and the brain, both of which remained unfinished at the time of his death.

Von Neumann freely acknowledged the contribution of Turing's 1936 paper to his work on automata theory but made no such connections in his discussion of computer design. Now that we know how clearly and concisely von Neumann could explain the universal Turing machine in 1945, its absence in his other reports, lectures, and letters of the 1945–1946 period speaks loudly, like the dog that Sherlock Holmes realized had failed to bark during the night. Von Neumann deployed it in an introductory lecture, but not for other purposes. He made no reference to any feature of the Turing machine in the First Draft or his other 1945–1946 detailed writings on computer architecture and instruction sets. Neither did

he mention Turing's demonstration of universality (or related work by Post and Church) in his other speeches and letters lobbying for the construction of EDVAC-like computers.^m In contrast the neuron notation and cybernetic language did make it into the First Draft, though as a convenient way to describe digital logic rather than as a source of architectural inspiration. The abstraction from constraints of space and time that eventually made Turing machines so useful for computer scientists looking to lay theoretical foundations for their new field made them irrelevant to people trying to design the first real electronic computers. ■

^m For example, a lengthy 27 August 1945 report to the Navy Ordnance Department on "Computer Services" (HHG-APS, box 9). See Priestley¹⁴ (2018, 92-3) for a discussion of the term JvN used instead, namely "all-purpose," and its difference from Turing's notion of "universal."

References

1. Akera, A. *Calculating a Natural World: Scientists, Engineers, and Computers During the Rise of U.S. Cold War Research*. MIT Press, Cambridge, MA, 2007.
2. Aspray, W. *John von Neumann and the Origins of Modern Computing*. MIT Press, Cambridge, MA, 1990, 189–212.
3. Birkhoff, G. Computer Developments 1935–55, as Seen from Cambridge, USA. In *A History of Computing in the Twentieth Century*, N. Metropolis, J. Howlett and G.-C. Rota, Eds., Academic Press, New York, 1980, 21–30.
4. Church, A. Review of Turing "On Computable Numbers..." *Journal of Symbolic Logic* 1 (1937), 42–43.
5. Davis, M. *Computers and Unsolvable Problems* (Dover, 1958).
6. Davis, M. *Engines of Logic: Mathematicians and the Origin of the Computer*. Norton, New York, NY, 2001.
7. Galison, P. The Ontology of the Enemy: Norbert Wiener and the Cybernetic Vision. *Critical Inquiry* 21, Autumn 1994, 228–266.
8. Haigh, T. Actually, Turing did not invent the computer. *Commun. ACM* 57, 1 (Jan. 2014), 36–41.
9. Haigh, T., Priestley, M., and Rope, C. *ENIAC In Action: Making and Remaking the Modern Computer*, 2016.
10. Hodges, A. *Alan Turing: The Enigma*. Simon and Schuster, New York, NY, 1983.
11. Kline, R. *The Cybernetics Moment, Or Why We Call Our Age the Information Age*. Johns Hopkins University Press, 2015.
12. McCulloch, W.S. and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* 1943, 115–133.
13. Mooers, C.N. The computer project at the Naval Ordnance Laboratory. *IEEE Annals of the History of Computing* 23, 2 (Apr.–Jun. 2001), 51–67.
14. Priestley, M. *Routines of Substitution: John von Neumann's Work on Software Development, 1945–1948*. Springer, Cham, Switzerland, 2018.
15. Priestley, M. and Haigh, T. The media of programming. In *Exploring the Early Digital*. Springer, Cham, Switzerland, 2019, 135–158.

Thomas Haigh (thaigh@computer.org) is a Professor of History at the University of Wisconsin—Milwaukee and Comenius Visiting Professor for the History of Computing at Siegen University in Germany. He has written widely on the history of computing—read more at www.tomandmaria.com/tom.

Mark Priestley (m.priestley@gmail.com) is Research Fellow at the U.K.'s National Museum of Computing, Bletchley Park, U.K. See www.markpriestley.net for more information.

Copyright held by authors.

^k "NOL Notebook 3," in box 28 of the Calvin R. Mooers Papers, Charles Babbage Institute, Minneapolis, MN.

^l Lengthy extracts from Mooers' diary are in box 28 of his papers at CBI.

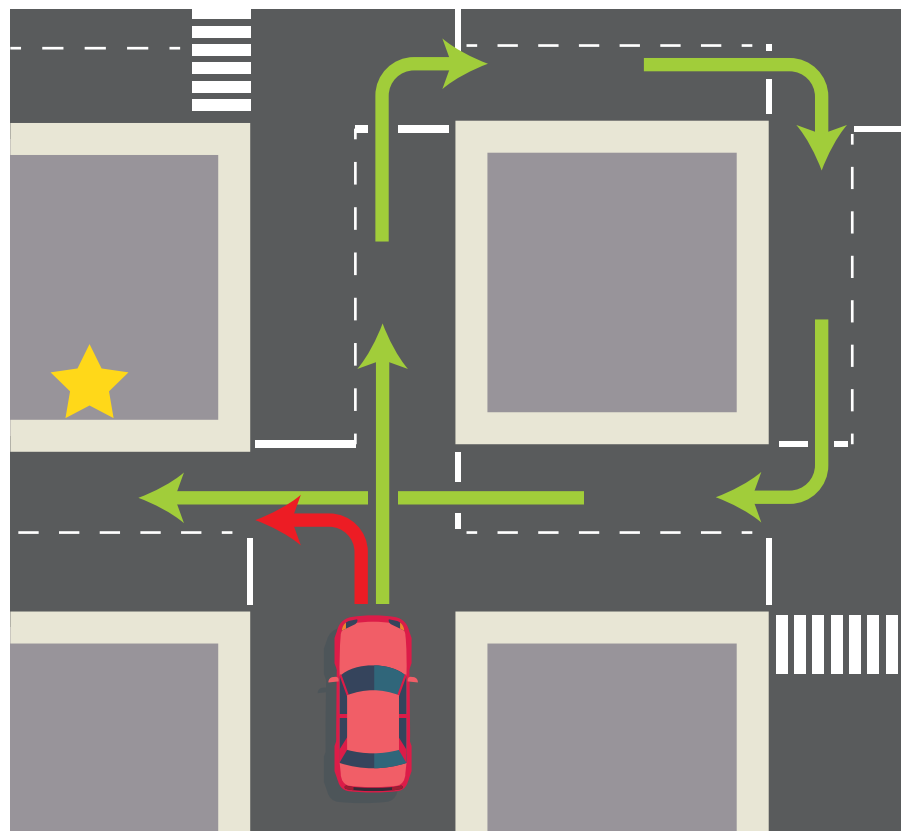
Viewpoint

Ethics of Technology Needs More Political Philosophy

Incorporating considerations of reasonable pluralism, individual agency, and legitimate authority.

AS A DRIVER, have you ever asked yourself whether to make left turns? Unprotected left turns, that is, left turns with oncoming traffic, are among the most difficult and dangerous driving maneuvers. Although the risk of each individual left turn is negligible, if you are designing the behavior of a large fleet of self-driving cars, small individual risks add up to a significant number of expected injuries in the aggregate. Whether a fleet of cars should make left turns is a question that any developer of self-driving cars and any designer of mapping and routing applications faces today.

A more general issue is at stake here: the decision of whether to make left turns involves a trade-off between safety and mobility (the time it takes to get to a destination). You gain safety at the expense of mobility by driving around the block and thereby avoiding left turns. But you gain mobility at the expense of safety by designing self-driving cars to zip through small gaps in oncoming traffic. Other situations that exemplify this mobility—safety trade-offs include merging onto highway lanes, driving through crosswalks with limited visibility, or avoiding detours by routing through school zones. Such maneuvers are very common, and we will soon be able to regulate them



centrally via software as cars become increasingly automated.

How should we make this trade-off between mobility and safety? What is the right balance? These are hard questions about the values that we build into self-driving cars. The ongoing debate on the ethics of self-

driving cars typically focuses on two approaches to answering such questions: moral philosophy and social science. Although each has its place, both approaches fall short.

The first approach turns to moral theories. For example, a utilitarian theory that seeks to maximize overall



Digital Threats: Research and Practice

DTRAP is a peer-reviewed journal that targets the prevention, identification, mitigation, and elimination of digital threats. DTRAP promotes the foundational development of scientific rigor in digital security by bridging the gap between academic research and industry practice. DTRAP welcomes the submission of manuscripts that address extant digital threats, rather than laboratory models of potential threats. To be accepted for publication, manuscripts must demonstrate scientific rigor and present results that are reproducible.



For further information
or to submit your
manuscript,
visit dtrap.acm.org

happiness would probably recommend left turns, assuming that getting to your destination quickly creates more happiness than driving around the block to avoid left turns. But answering value questions purely by deduction from moral theories is a bad idea. First of all, it is an open question whether one moral theory is correct.^a And even if we had one correct or uniquely best supported moral theory, we still should not rush to implement the answer that this theory gives. We value human agency and individual autonomy. People should be free to do what they believe is right even if that involves making certain mistakes.⁶

The second way of answering ethical questions turns to psychology and the social sciences. We could gather data on how people in fact make this trade-off between mobility and safety. We could observe how people drive today; or we could survey people on how they think self-driving cars should operate; or we could look to how much people value mobility and safety in other forms of transportation such as air travel, and use this value elicitation to inform the balance between safety and mobility for self-driving cars.

But, although understanding all relevant empirical data is crucial, relying exclusively on empirical facts is also a bad idea. People often reveal discriminatory or unfair preferences. For example, how likely a driver is to yield at a crosswalk may differ with a pedestrians' age, race and gender.^{2,3,5} Although such data is illuminating and informative, we should be hesitant to build such preferences into our design.

Even if people had fair and unbiased preferences, insights from social science would still not be sufficient. First, although social science is useful for surfacing and quantifying disagreements, it provides only limited guidance on resolving disagreements. How we should deal with disagreements is itself a moral question about which people will disagree. Second, people are to some extent self-interested, which creates a compliance or incentive problem. For example, if you are driving in a self-driving car you might

want the car to protect your own safety at the expense of the safety of others.¹ Sometimes you should be allowed to prioritize your own interests to some extent and not put what you believe is morally right for others before what you believe is best for yourself. The balance between mobility and safety needs to be struck in a way that is compatible with individuals' fair pursuit of self-interest. This raises the questions: To what extent should we control, compel or force people to do what they themselves believe to be morally correct? Is the majority endorsement of a policy enough to force everyone into compliance? We should not answer these questions by majority opinion.

In sum, the two approaches of answering questions about the ethics of self-driving cars are both lacking. We should neither deduce answers from individual moral theories nor should we expect social science to give us complete answers. What further ways of addressing ethical questions can we draw upon?

I argue that we should turn to political philosophy in addition to moral philosophy. The issues we face are collective decisions that we make together rather than individual decisions we make in light of what we each have reason to value.⁹ A basic mistake in the ethics of self-driving cars is asking only what an individual should do. This is the domain of moral philosophy. Whether you should eat meat or maintain a vegetarian diet is an example of a moral question. But in addition to such questions, we also need to ask what makes for good

**The balance
between mobility
and safety needs
to be struck in a way
that is compatible
with individuals'
fair pursuit
of self-interest.**

^a This is not to say that any moral theory is as good as the next one. You can believe that some theories rest on better arguments than others.

policies and institutions. Policies and institutions result from collective decisions and form the domain of political philosophy.

Political philosophy adds three basic concerns to our conceptual toolkit. First, political philosophy starts with the idea of reasonable pluralism, or the recognition that people often disagree for good reasons.⁸ Almost all issues are complex, and a large amount of considerations and evidence can be brought to bear on the issues. At the same time, every one of us has different life experiences that inform our values. We might disagree for good reasons—perhaps we should cherish such a pluralism of values.

Second, political philosophy needs to balance values with a respect for human agency and individual autonomy. Even if everyone agreed that eating meat is morally bad, we would still need to ask how eating meat should be regulated. Within reasonable limits, individuals should be free to decide for themselves. We should not confuse questions on the individual level—What is wrong with eating meat? How should your self-driving car drive?—with questions on the collective level: How should we regulate meat consumption? How should we regulate self-driving cars?

Third, political philosophy worries about legitimate authority and why we are obliged to abide by decisions of our political institutions. What is regulated, how it is decided and by whom makes a difference. For example, even if we all agreed that eating meat is wrong, we might still hold that a private company lacks the authority to regulate meat consumption. Maybe certain issues need to be left unregulated. Maybe certain things should be done only by the government. These are all issues of legitimate authority.

These three concerns—reasonable pluralism, individual agency, and legitimate authority—are central themes in political philosophy. They have so far been largely overlooked in the debate on the ethics of self-driving cars. To illustrate this idea, think of the cases that have captivated the public discussion on the ethics of self-driving cars.⁴ These so-called trolley cases ask you to imagine that a self-driving car needs to choose who has to die. If a car has a choice between running over five

When we think about good regulation, we can import concepts from political philosophy to inform our collective decision making.

pedestrians or only one, when all pedestrians are identical in all relevant respects, what should the car do?

A major problem with such trolley cases and other such dilemmas is that they look at these choices as if they were exclusively a moral problem even though they raise a distinctively political problem. Trolley cases ask: What is the right thing to do? What would you do? What should the car do? But instead we need to think more broadly about value pluralism, individual agency, and political legitimacy when developing self-driving cars.

Self-driving cars—whether it is about trolley cases or left turns—raise the question of how we get along as a community or people. We now have a chance to regulate traffic systems to a degree that was just technically impossible before. When we think about good regulation, we can import concepts from political philosophy to inform our collective decision making. For example, letting passengers of self-driving cars set at least some of the driving parameters themselves would be one way of achieving greater respect for reasonable pluralism, individual autonomy, and legitimacy.

This point generalizes to other issues, such as the ethics of artificial intelligence. How do recently heralded AI ethics principles respect reasonable pluralism, individual agency, and legitimate authority? How do corporate ethics principles reflect the diversity of values in society?

To respect reasonable pluralism, our responses to ethical challenges of technology should find common

ground between the fundamentally different values that people hold and consider processes of incorporating and resolving value conflicts.⁷

To respect individual agency, we should hesitate to replace human judgment and decision making. And if we make decisions that affect others, we should keep their interests in mind and try to decide for them as they would for themselves.

To respect legitimacy, corporations should democratize decisions that affect basic goods such as individuals' liberty, safety, or opportunity. Options to democratize corporate decisions can involve government regulation, stakeholder participation, or deliberative input from the public, among others.

How such political concerns bottom out in policies in detail is a question ripe for interdisciplinary cooperation. We should no longer overlook political philosophy when deliberating about the social challenges that arise from new technologies but make pluralism, agency, and legitimacy central pillars of the discussions. ■

References

1. Bonnefon, J.-F., Shariff, A., and Rahwan, I. The social dilemma of autonomous vehicles. *Science* 352, 6293, (Jun. 2016), 1573–1576.
2. Coughenour, C. et al. Examining racial bias as a potential factor in pedestrian crashes. *Accid. Anal. Prev.* 98, no. Supplement C, Jan. 2017), 96–100.
3. Goddard, T., Kahn, K.B., and Adkins, A. Racial bias in driver yielding behavior at crosswalks. *Transp. Res. Part F Traffic Psychol. Behav.* 33 Supplement C, (Aug. 2015), 1–6.
4. Himmelreich, J. Never mind the trolley: The ethics of autonomous vehicles in mundane situations. *Ethical Theory Moral Pract.* 21, 3 (May 2018), 669–684.
5. Kahn, K.B. et al. Racial bias in drivers' yielding behavior at crosswalks: Understanding the effect. National Institute for Transportation and Communities, Tech Report NITC-RR-869, Oct. 2017.
6. Millar, J. Technology as moral proxy: Autonomy and paternalism by design. *IEEE Technol. Soc. Mag.* 34, 2 (Jun. 2015), 47–55.
7. Rahwan, I. Society-in-the-loop: Programming the algorithmic social contract. *Ethics Inf. Technol.* 20, 1 (Mar. 2018), 5–14.
8. Rawls, J. *Justice as Fairness: A Restatement*. Harvard University Press, 2001.
9. Susskind, J. *Future Politics: Living Together in a World Transformed by Tech*. Oxford University Press, 2018.

Johannes Himmelreich (jrhimme@maxwell.syr.edu) is an assistant professor of Public Administration and International Affairs at Syracuse University's Maxwell School of Citizenship and Public Affairs, Syracuse, NY, USA.

Copyright held by author.



Watch the author discuss this Viewpoint in the exclusive *Communications* video. <https://cacm.acm.org/videos/more-political-philosophy>

Viewpoint

A* Search: What's in a Name?

A search for algorithmic answers returns unique results.

ORIGINALLY PUBLISHED IN 1968 by Hart, Nilsson, and Raphael,² the well-known A* search algorithm is a foundational pathfinding algorithm in computer science and artificial intelligence (AI) for traversing trees and graphs. The method provides the optimal path from the initial state to the target goal state, given the use of an admissible heuristic (must not overestimate the remaining distance to the goal). The A* algorithm is included in nearly all AI textbooks and courses worldwide. Given its widespread fame, however, there is no reliably documented evidence as to the origin of the name “A*”: What does it really stand for and what does it mean? This *Communications Viewpoint* answers the question.

At Ohio State University, we offer a specialty course on AI designed for non-computer science students¹ (due to the large interest and demand in AI from multiple disciplines). During the course offering this year while teaching the A* algorithm, a student raised his hand and asked: “What does A* stand for?”. Out of all my years teaching AI, I have never been asked that question! I realized I did not know the answer myself, and therefore would need to seek the explanation. After an exhaustive search online (including reading the original publication²) and a thorough inspection of the classic AI textbooks on my bookshelf, I could not find a definitive answer. During the next class, I stated the lack of evi-



dence available and therefore put out a challenge to the class to see if anyone could find a credible source for the answer.

Jeff Hachtel, an ambitious Management Information Systems undergraduate in my course, took that challenge to heart and began his search. One potential source was found stating the A* name is based on the (plausible) use of Kleene star syntax³ (used in regular expressions), however it was not properly verifiable. Jeff then boldly decided to take the question straight to the authors of A* themselves. He found the email addresses of Hart, Nilsson, and Raphael and sent them a brief inquiry asking how the name was selected and if it was related to Kleene star syntax.³ Given the paper was published in 1968, it was not expected any response(s) would be forthcoming.

Incredibly, the very next day responses from the authors sharing their recollection and historical perspective began to appear in Jeff’s email inbox. Jackpot!

Bert Raphael was the first to respond,^a and began with the statement “A* was a quick, easy, and rather arbitrary name for the algorithm we came up with.”

Raphael continued, recalling: “As I remember, Nils called me into his office (in about 1968) to show me his proposed algorithm for how ‘Shakey the Robot’ could plan its route through a room containing obstacles. For lack of anything better, he called it Algorithm A. I suggested a modification of Algorithm A that I had a hunch would be more efficient. Peter then dropped in, looked at

a B. Raphael, Email communication, February 7, 2019.

our proposals, and suggested that the modified algorithm was not only better, but (under certain conditions) might be unique and optimal. Then, for the next week or two, we studied and established under what conditions the revised algorithm was provably optimal, and we called it A^* *just to distinguish it from any simpler algorithm A*. That's about it!"

Nils Nilsson's initial response to Jeff's email supported a relationship to the Kleene star interpretation.^b

Peter Hart then gave a detailed reply^c to everyone, initially saying this may be the first time all three of them have participated together on this topic, and then stated "we seem to have different recollections about some details of nomenclature and notation."

"Only a few years earlier, I had completed a Ph.D. dissertation at Stanford in the area of nonparametric statistical decision theory. So when I started working on the mathematical proofs with Nils and Bert, I adapted some standard nomenclature and notation I was familiar with from the field of mathematical statistics. This comes up in several places: 'Admissibility' itself is a standard concept in statistics and statistical decision theory, you can easily find lots of examples and explanations. From an intuitive point of view, it usually means that something has a 'good' property. From a mathematical point of view, it limits consideration of things (like say decision rules) to a 'good' class, about which interesting theorems can be proven.

"The circumflex (^) or 'hat' notation is commonly used in statistics to denote an estimate, as of a random variable. So, if for example \hat{a} might be used to denote an estimate of a random variable 'a'. I introduced the hat notation for the f, g and h functions in A^* to suggest they were estimates of the true, but unknown, underlying values (in particular of the look-ahead function, h). That brings us to the asterisk, the 'star' in A^* .

b N. Nilsson, Email communication, February 7, 2019. Sadly, Nils Nilsson passed away soon after we submitted this material for publication in *Communications*; see <https://stanford.io/333qB3A>. We are grateful we had the opportunity to correspond with Nils; our sincere thanks to Peter Hart and Bert Raphael for their participation and historical perspectives contributing to the development of this Viewpoint.

c P. Hart, Email communication, February 7, 2019.

Decades after the A^* algorithm was initially published we finally have our answer(s).

The star notation is probably a bit over-used in statistics, with different authors employing it in different ways. But it can mean a special or optimal value of a parameter, such as one that minimizes some cost or loss. I introduced the A^* notation with the thought that our algorithm (A^*) was better than any other algorithm, better than anyone else's algorithm A, and we're gonna prove it! So from my PoV, the star has nothing whatsoever to do with Kleene star syntax."

Lastly, Nilsson clarified his position to support the statement made by Hart, affirming "that's my understanding too". (Afterward, a Nilsson reference was actually found that similarly supports a "special property" interpretation.⁴)

Decades after the A^* algorithm was initially published we finally have our answer(s). Though there remain slight differences in opinion behind the true meaning and use of the star (distinction vs. optimality), it has nonetheless been an interesting and illuminating historical journey through A^* . Perhaps it is time to update the textbooks. ■

References

1. CSE 5052. Survey of Artificial Intelligence for Non-Majors. Department of Computer Science and Engineering, Ohio State University, 2019.
2. Hart, P.E., Nilsson, N.J., and Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics SSC4* 4, 2 (Feb. 1968), 100–107.
3. Kleene, S.C. Representation of events in nerve nets and finite automata. In *Automata Studies*. Princeton University Press, Princeton, New Jersey, 1956, 3–41.
4. Nilsson, N. *The Quest for Artificial Intelligence*. Cambridge University Press, 2009.

James W. Davis (davis.1719@osu.edu) is a Professor in the Department of Computer Science and Engineering, Ohio State University, Columbus, OH, USA.

Jeff Hachtel (jeffrey.r.hachtel@accenture.com) received a Management Information Systems degree from Ohio State University and is currently a Consulting Analyst for Accenture, Columbus, OH, USA.

Copyright held by authors.



Association for
Computing Machinery

ACM Transactions on Internet of Things

ACM TIOT publishes novel research contributions and experience reports in several research domains whose synergy and interrelations enable the IoT vision. TIOT focuses on system designs, end-to-end architectures, and enabling technologies, and on publishing results and insights corroborated by a strong experimental component.



For further information
or to submit your
manuscript,
visit tiot.acm.org

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

Understanding enterprise reliability.

BY SANJAY SHA

The Reliability of Enterprise Applications

ENTERPRISE RELIABILITY IS a discipline that ensures applications will deliver the required business functionality in a consistent, predictable, and cost-effective manner without compromising core aspects such as availability, performance, and maintainability.

While achieving a high level of reliability is a common goal of most enterprises, reliability engineering involving third-party applications can be a complex landscape. First-party software affords the luxury of building a modular and extensible application that integrates seamlessly with an enterprise's IT ecosystem. Third-party software does not always have the same flexibility. Incorporating an off-the-shelf enterprise application within an existing IT ecosystem, without compromising functionality and reliability, is a classic engineering and philosophical problem the CIO's office has to deal with all the time.

Despite this complexity, enterprises still pursue and select third-party software to power their business verticals such as human resources (HR), legal, and



finance, since it makes economic sense to pay for an enterprise application rather than building the software in-house. Enterprises sometimes base their buying decisions only on the required business functionality, however, and tend to overlook the application's overall reliability. This can compromise the availability and supportability of the application and increase the cost of managing it in the long run.

This article describes a core set of principles and engineering methodologies that enterprises can apply to help them navigate the complex environment of enterprise reliability and to deliver highly reliable and cost-efficient applications and that can help them navigate the complex environment of enterprise reliability.

Reliability axioms are a set of principles that emphasize the values and behaviors that help foster and maintain the culture of enterprise reliability.

Culture = Reliability Axioms (**Values**)
x Reliability Engineering (**Behaviors**)

These five core axioms define en-



terprise reliability and form the basis of this article: Focus on the customer; select the right vendor; invest in a common application platform; engineer reliability to be cost effective; and build an engineering-centric support organization (or site reliability engineering, SRE).

Focus on the customer. Customer objectives determine the reliability of an application. Having a well-defined set of customer objectives is foundational, as these translate into tangible and measurable goals. These goals, also known as service-level objectives (SLOs), drive the overall reliability posture of an application such as availability, performance, data integrity, monitoring, and responding to incidents. SLOs ensure an application is engineered to meet the precise needs of the customer.

Select the right vendor. The choice of vendor impacts the reliability of the core application. Choosing an enterprise application involves much more than just buying software that meets the business functionality. It involves

partnering with a vendor that thinks and builds software with similar principles and values to the enterprise (for example, secure by design, scalable software components, open APIs for extensibility, and ease of support and maintenance).

Invest in a common platform. The overall reliability of an application is the sum total of the reliability of the business's core application and all its dependencies. Transforming the baseline dependencies into a common platform can help standardize and bring consistency in how an application is managed. Using a common platform can drastically decrease technical silos and increase overall reliability and efficiency. A common platform could mean having a shared deployment manager, CI/CD (continuous integration/continuous delivery) frameworks, or shared service management workflows for monitoring, logging, backups, and so on.

Engineer reliability to be cost effective. Over-engineering reliability breaks the ROI (return on investment)

curve. Reliability is a function of how mature an application is and, as a result, its overall availability. Imagine you have a service with a 99.9% SLO. Adding an extra nine (99.99%) sublinearly increases the availability of your service, as shown in Figure 1.

While improving the reliability of your service from 43.2 minutes of downtime per month to 4.32 minutes can be tempting, it can represent a significant engineering feat with a hefty price tag. Therefore, when specifying the required availability of a service, the decision should be based on the business requirements: “How available (how many nines) does my system need to be, in order to meet the business objectives?”

Build an engineering-centric support organization. Application reliability is preserved by SREs. Designing a perfect application doesn't guarantee a high-quality production experience—at least not without the support of an SRE organization. Both the application and the IT ecosystem where it runs change constantly—with developers

Terminology

Business owner: The owner or person leading a business vertical such as legal, finance, or HR; business owner and customer are used interchangeably.

Customer: The business owner of a business vertical such as legal, finance, or HR.

Enterprise applications: Software owned by an external company, also known as third-party software.

SLO: Service-level objective; a quantifiable objective that measures the effectiveness of business functionality.

SRE: Site reliability engineering; the enterprise's support organization. Some enterprises may not have a dedicated SRE team; instead, they have support teams such as DevOps and system or IT admins. The principles and methodologies outlined here are generic enough that they can be applied to any support organization.

User: An employee of an organization who represents the consumer of an enterprise application.

Vendor: A provider of third-party software.

pushing new code, vendors publishing new security patches, or infrastructure teams updating the software of the underlying platform.

Reliability is not a “build once and forget for life” construct; it is a continuous process of maintaining and upholding its principles and methodologies. Enterprises that recognize the need and invest in developing SRE skills stand out from the rest because they recognize that without these skills, enterprise reliability cannot be sustained.

Designing Enterprise Reliability Engineering

Designing for enterprise reliability is a multidimensional problem that spans multiple entities: customer, vendor, platform engineering, cost, and the SRE

organization. The rest of this article expands on these axioms and describes the behaviors, principles, and methodologies that influence and shape the discipline of enterprise reliability.

Customer objectives. “If you don't understand your customer objectives, then you do not need to exist as an org.” Whether you are a traditional IT organization or a mature SRE org, this fundamental principle holds true.

Translate customer objectives to SLOs. In an enterprise setting, a typical customer is the owner of a business vertical such as legal, finance, or HR trying to accomplish a specific business goal. Having a well-defined set of business objectives lays the foundation for developing concrete functional requirements, allowing you to effectively translate those requirements into quantifiable and measurable outcomes, also known as SLOs.

Defining SLOs early on leads to a better design and implementation of the overall system. Arriving at a clear set of measurable SLOs, however, is an exhaustive process with a lot of considerations (for example, what is technically feasible vs. infeasible, expensive vs. cost effective, reliable vs. fragile). Closely involving the customer and vendor throughout this process is crucial, as it develops a shared understanding of requirements, constraints, and trade-offs, and helps reconcile the gap between aspirational and achievable SLOs.

Documenting the SLOs, including a strong rationale for the established targets and thresholds (for example, 99.9 uptime) is key, as this becomes the contract among all the parties (SRE team, software vendor, and customer). This rigor also creates a culture of transparency and openness to inform how the system should be designed and how the service should operate. For a deep dive into engineering SLOs effectively, refer to the SLO chapter in the SRE book.¹

Empathy toward customer and vendor is key. Customers (business owners) may not always have the same level of understanding of the problem space.

Their approach could be purely business driven, and they may expect the application never to go down. Likewise, the vendor may not entirely understand how the IT ecosystem is designed and cannot operate independently to deliver the system. The SRE team should become a true partner to bring alignment between the customer and vendor and develop a shared understanding of the overall objective, specific requirements, and constraints of the domain.

Given the nature of third-party domains, it may be difficult to find a perfect system that meets 100% of the business functionality, as there are many variables in the equation (for example, third-party software, hardware, cost, and vendor). Therefore, working closely with the customers in developing a set of detailed requirements and distinguishing core vs. optional requirements helps with the trade-off analysis—for example, if the application has constraints, evaluating their impact on business objectives or revisiting and adjusting customer requirements without compromising the business objectives, or finding a new vendor altogether.

Taking customers through this journey from beginning to end helps them better understand the space and weigh in on all important considerations, ultimately allowing them to make effective business-driven decisions.

SLOs as a means to customer happiness; solve for SLOs. Solving for customer happiness based on objective goals is key; it is better to cater to functionality based on the customers' objectives in a measurable way (SLOs). Customers have only one fundamental criterion: Is the system able to translate business objectives into business functionality in a cost-effective and reliable manner?

Having this objective view creates a transparent and blameless culture. The key point to remember, however, is that SLOs are not fixed for life: As business needs evolve, the system SLOs need to be revisited. Therefore, having a strong discipline of revisiting the SLO agreements periodically with the customer helps tackle these changes and adjust the scope and expectations as business needs evolve.

Vendor selection. Enterprise-application engineering with a vendor is a

Figure 1. Reliability as a function of availability.

Availability Percent	Nines	Downtime / Month
99.9	3	43.2 minutes
99.99	4	4.32 minutes

long-term investment that goes beyond the application itself. Therefore, it's important to select a vendor that aligns with the values and principles of the enterprise—for example, software design discipline (scale and performance), data security and privacy management, use of open standards, and ease of operations and maintenance.

To ensure a vendor meets its requirements, an enterprise needs a rigorous evaluation and validation process. Two distinct sets of evaluations determine and shape the reliability of an application:

- ▶ **Functional evaluation:** represents the business functionality required by the customer.

- ▶ **Infrastructure evaluation:** represents the application's IT requirements.


Functional evaluation. Functional requirements are derived directly from customer objectives and form the basis of the evaluation process. Each functional requirement has a set of key functional characteristics. The goal of the evaluation process is to do an in-depth analysis of these characteristics and assess the feasibility of third-party software.

To understand this, consider the following scenario. Assume your enterprise is evaluating a third-party IT inventory system to manage your corporate IT asset information. One of your business objectives is to predict the supply and demand for your inventory in realtime. This could result in a requirement for a centralized global inventory database that updates in realtime every time a checkout happens.


Based on this scenario, let's analyze the core characteristics that a functional evaluation should delve into.

Functional specification. Does the vendor understand the functional requirement and the expected outcome? In the scenario just described, the functional requirement is to maintain a global inventory database for all asset information. The expected outcome is the ability to track asset information and update the global inventory database in realtime.

Dependencies and constraints. Does the vendor need to be aware of any core dependencies or constraints? For example, does the global inventory database depend on any external entities? Is a centralized database required for



Defining SLOs early on leads to a better design and implementation of the overall system. Arriving at a clear set of measurable SLOs, however, is an exhaustive process.



reads and writes, or is a distributed setup required? What are the pros and cons of both approaches?

Functional interfaces. Does the vendor understand all the end-to-end functional interfaces involved in this requirement? For example, does the inventory database have any reporting interfaces? How does the admin interact with the database? How do the users interact with the database when they do a checkout? What is the end-to-end flow?

Geographic requirements. Does the enterprise have a presence across the globe? Will users access this inventory system from different regions? What are the specific performance and latency requirements for these users?

Scale and load requirements. How many users are going to use the inventory system, both globally and per region? What are the QPS (queries per second) or load requirements for these users? Are there any peak or off-peak volume requirements or considerations?

Security requirements. Does the vendor understand the security posture of the system? Are there any specific access restrictions based on user type (for example, admin vs. normal user)? What is the authentication and authorization mechanism? Does the application depend on a centralized authorization service such as LDAP (Lightweight Directory Access Protocol) or AD (Active Directory)? Is there a single sign-on dependency?

Compliance requirements. Does the vendor understand and meet the compliance requirements for this application?

Handling requirements. Does the vendor understand the key failure modes based on the design of the system? How does the vendor's software handle exceptions (for example, request timeouts, retries during write failures, and connection resets)?

Release management. What software release management discipline does the vendor use? What is the release cycle? How are changes tested before being released to the customer? What is the QA/qualification process?

Testing and validation. Does the vendor have a holistic testing plan that covers the end-to-end workflow, and does it include all the edge cases? What is the testing plan for measuring load and performance?

Infrastructure evaluation. Infrastructure requirements create the foundation for the whole application. Therefore, ensuring the end-to-end reliability of this base layer is critical.

Every enterprise is unique and has its own set of infrastructure requirements and constraints. When evaluating an enterprise application, you want to ensure the vendor can comply with the requirements of the enterprise's IT ecosystem. For example, suppose your enterprise has fully adopted virtualization for internal efficiencies and other business reasons. In this case, the vendor's application should be compatible with

and supported on VMware. Otherwise, the application could become a nonstandard model in your IT organization, driving up costs related to infrastructure, licensing, hardware, and support.

Following are a set of key infrastructure requirements to ensure a vendor's software is compatible with an enterprise's IT ecosystem.

Core infrastructure. The vendor must meet an enterprise's hardware, network, and operating system requirements. This includes specific hardware models, enterprise databases, software and operating systems versions that the IT team supports.

Networking. The vendor must meet the authentication and authorization requirements of the network—for example, LDAP or AD, or single sign-on requirements.

Infrastructure security. The vendor should understand and meet the enterprise's security policies related to access management, perimeter security, and data encryption.

Infrastructure sizing. The enterprise should derive a concrete sizing plan including the number of environments and compute and storage requirements based on its functional requirements, and evaluate the vendor closely to ensure its software can scale and meet those sizing needs.

High availability and disaster recovery. The SRE team should have a clear understanding of reliability requirements based on SLOs and customer objectives. Deciding on the high-availability design such as active-active or active-passive, disaster-recovery requirements and strategy, and data recovery (recovery point objective) and restore (restore point objective) are all critical when engaging the vendor. The enterprise must ensure the vendor's application can meet its requirements, or that the vendor is willing to collaborate with the SRE team to provide the needed reliability.

Data management. The vendor should have a clear data management discipline and methodologies when it comes to data integrity, backup, recovery, and retention. Does the vendor have a strong data security discipline such as encryption of data both in transit and at rest?

Integrations. Make a list of all the dependent systems and necessary integrations that the IT ecosystem requires—for example, authentication services such as LDAP or AD; corporate mail service and the necessary integrations; and service management workflows such as centralized backups, monitoring, and logging.

Operability. Ensure the vendor has a strong discipline of software updates/upgrades, clearly defined maintenance windows, and so on.

These requirements provide an overview of the core aspects and characteristics you should evaluate when choosing an enterprise application. Note this is not an exhaustive list, and requirements may vary among enterprises.

Figure 2. Common platform.

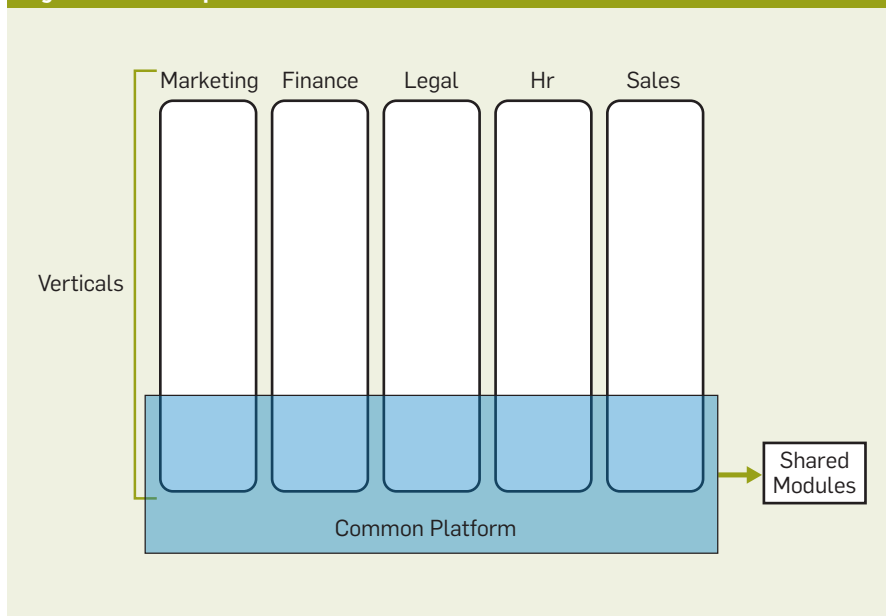
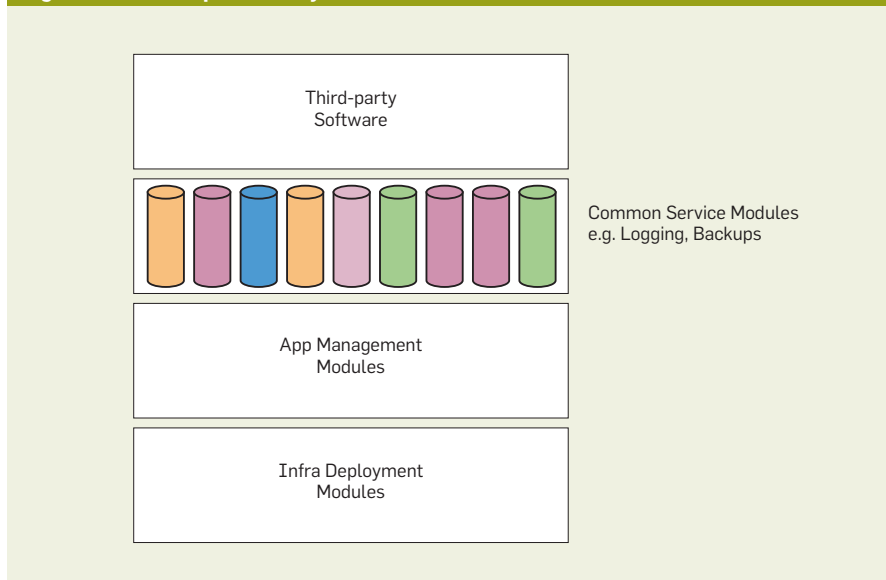


Figure 3. Common platform layout.



Functional and infrastructure requirements can heavily influence the design and delivery of an application. Therefore, evaluating the feasibility of these requirements is a crucial step in engineering the reliability of an application.

Common application platform. Most enterprises rely on third-party software to support the operations and needs of their business verticals (Figure 2). Running different third-party applications, however, can lead to a large number of disparate systems within an enterprise. Not having a common baseline across applications makes maintaining the reliability and efficiency of service more difficult over time. This creates a lot of overhead for the SRE team and increases the organization’s operational costs.

A common platform provides a standard operating environment in which to run all of a system’s applications, enhancing the overall reliability and efficiency of an enterprise. The key principle of implementing a common platform is to identify, build, and enforce a set of shared modules and standards that can be reused across the applications that support the business verticals.

On the other side, overengineering a common platform can have a negative impact. If a platform has many standards in place or becomes too rigid, an enterprise’s delivery and execution speed can decrease significantly.

The goal is to develop a strategy that allows enterprises to find the right balance between optimizing for reliability and maintaining the development speed needed to deliver and support business functionality. Finding this balance requires a careful analysis of the trade-offs and net benefits.

Common platform layout. An application platform consists of a set of modules that can be grouped into three main categories (Figure 3):

- ▶ Infrastructure deployment modules.
- ▶ Application management modules.
- ▶ Common service modules.

Infrastructure deployment modules provide intent-based deployment of an end-to-end application environment based on a set of resource requirements such as CPU, memory, operating systems, and the number of instances. This mechanism is highly efficient since the workflows only need

to be configured once and can be triggered as needed. It also provides a standardized, consistent, and predictable environment, which improves overall reliability.

Many enterprises are already embracing open-source technologies to help them manage the underlying infrastructure of their applications. Tools such as Terraform provide abstractions to handle the provisioning and deployment of end-to-end environments agnostic to the underlying platform (for example, on premises vs. cloud).

Application management modules handle critical workflows during the life of an application. A few examples of these workflows include:

- ▶ Configuration management workflows to deploy application configuration.
- ▶ Release management workflows to manage software releases and rollbacks.
- ▶ Security management workflows to manage secrets and certification deployments.

Software solutions such as Puppet, Chef, and Ansible provide frameworks and solutions for enterprises to orchestrate these workflows across their applications.

Common service modules manage the standardized workflows that can be shared across all applications, such as logging, monitoring, and reporting. This layer can also include custom service modules for the specific needs of an enterprise, such as a custom web front end or a single sign-on service.

Some examples of common service modules include:

- ▶ Monitoring module to collect and publish metrics for reporting and alerting.
- ▶ Backup module to execute backups, retention, and recovery.
- ▶ Log collection module to securely ship logs to a centralized log service.
- ▶ Custom Weblog/Tomcat as a ser-

vice offering middleware capabilities.

- ▶ Managed DBaaS (database as a service) module to manage database workflows.

Combining infrastructure deployment, application management, and common service modules creates a platform that enables enterprises to move away from managing monolithic applications and into a new realm of modular, extensible, and reusable applications.

Cost engineering. When enterprises opt for third-party software, they are making a cost- and ROI-based decision to use a “reliable” enterprise application that delivers the business functionality in a cost-effective manner. Determining the right reliability-to-cost trade-off that sustains the ROI curve is the crux of cost engineering.

Reliability-to-cost trade-off. Figure 4 illustrates how reliability (the number of nines) directly influences the overall availability or reduction in downtime. The reduction with each additional nine is sublinear. While it is extremely tempting to add a nine, it is important to recognize that engineering an additional nine can be expensive, and overengineering reliability produces diminishing ROI. To understand this, let’s look at the following scenario.

Enterprise ABC is looking for a third-party sales application that can provide market analysis and insights. The sales team predicts they can generate an average of \$600/hour of revenue by leveraging those insights. Their revenue target per quarter is approximately \$1.2 million. What is the required uptime (availability SLO) for this application?

If the application was available 100% of the time, the maximum revenue would be:

Net revenue = hours in a quarter (3 months x 30 days x 24 hours = 2,190) * earnings per hour (\$600)

\$1,296,000 (~ \$1.29M) = 2,160 hours in a quarter * \$600 per hour

Figure 4. Availability and reliability.

Percent	Nines	Downtime / Qtr	Downtime / Month
90	1 nine	9 days	3 days
99	2 nines	21.6 hours	7.2 hours
99.9	3 nines	2.16 hours	43.2 minutes
99.99	4 nines	12.96 minutes	4.32 minutes
99.999	5 nines	1.30 minutes	25.9 seconds

The net revenue (~\$1.29 million) clearly exceeds the target revenue of \$1.2 million, but 100% availability is infeasible. Figure 5 illustrates how to choose the perfect availability SLO that meets the ROI.

Here are the key conclusions reached in this scenario:

1. A 90% availability SLO generates ~\$1.16 million in revenue, which falls short of the target revenue of \$1.2 million. This SLO is not feasible.

2. A 95% availability SLO generates ~\$1.23 million in revenue, which comfortably meets (slightly exceeds) the revenue objective of \$1.2 million. This SLO is feasible.

3. A 99% availability SLO generates ~\$1.28 million in revenue, which far exceeds the revenue objective of \$1.2 million, but it comes with additional overhead:

- ▶ A 95% SLO guarantees no more than 36 hours downtime per month and still comfortably meets the target revenue.

- ▶ In contrast, a 99% SLO guarantees no more than 7.2 hours downtime per month, but the cost of engineering and support can be higher.

- ▶ As long as the cost to engineer a 99% SLO does not exceed \$80,000 (\$1.28 million to \$1.2 million), this is a viable option.

4. The net revenue growth for each additional nine provides diminishing returns (delta revenue)—for example, between 99.99% and 99.999%:

- ▶ There is a significant reduction in downtime per month from 4.32 minutes to 25.92 seconds, but the revenue increase is only \$116.64.

- ▶ To choose a 99.999% SLO, the added engineering cost should be <\$116.64.

Account for application dependencies. To design a system with a 99.9% SLO, the rule of thumb is to have all critical dependent systems provide an addi-

tional nine (that is, 99.99). This means you have to factor in the reliability investment (additional cost) for your application and all of its critical dependencies, because a system is only as available as the sum of its dependencies.²

Choose a SLO that fits the ROI curve. The ideal SLO is one that delivers the required functionality with a degree of reliability that fits within the ROI curve. In the previous scenario, the best SLO would be 95%, because it is the least expensive option that meets the business goal (\$1.2 million).

Overengineering reliability produces diminishing ROI. From the previous scenario, it is evident that increasing the availability of a service does not always translate to a significant growth in revenue. This is clearly evident from the scenario. In fact, with each additional nine, the benefit of engineering the reliability increases sublinearly, breaking the ROI curve.

Preserving Enterprise Reliability

Reliability is not just a systems design problem. You can have the world's best-designed system, but without proper rigor and discipline, preserving core aspects of the system such as availability, performance, and security can become extremely difficult. Reliability is a responsibility that should be shared across all teams involved in the system, including vendors, development, and SRE. The SRE teams are ultimately accountable, however, since they are responsible for achieving their SLOs. During the lifecycle of an application there are a few critical junctures where maintaining proper rigor can translate into preserving the reliability of the service.

Design for standardization and uniformity. Reliability is preserved when you recognize the importance of uniformity and invest in standardiza-

tion. One of the challenges of enterprise applications is there is no agreement or consensus among vendors on common standards around software technologies, operating systems, and workflow orchestration methodologies, such as release management and patch management. Each vendor provides its own flavor.

The role of SRE is to publish common standards for the portfolio of tools and technology they support (the base operating system, release management, and configuration frameworks) and the minimum operational maturity they expect from the vendor (for example, automated installs and seamless patching workflows).

Mature enterprises that rely on multiple software vendors recognize the importance of having a baseline ecosystem and strong operational maturity. They not only consider business functionality, but also account for ecosystem maturity when looking for third-party applications.

Change management. Change is powerful. You can build a highly reliable system, but one small change (a bad config push or a software bug) can compromise the reliability of the entire system. Preserving reliability comes from having a change-management rigor with a set of checks and balances that can detect, prevent, or minimize the impact of problems. SRE should be responsible for maintaining this rigor. Consider the following checks and balances.

Measure, monitor, and alert. Measure, monitor, and introduce thresholds to alert for everything that is on the critical path of your SLO. This provides the ability to proactively detect and fix issues.

Streamline change. Require all changes to go through validation and regression testing. This should be en-

Figure 5. Selecting the right availability SLO.

Availability SLO percent	Number of Nines	Downtime / month	Uptime / month	Net Revenue	Target Revenue	Delta Revenue
90	1	72 hours	648 hours	\$1,166,400	<1.2M	--
95	1.5	36 hours	684 hours	\$1,231,200	>1.2M	\$64,800
99	2	7.2 hours	712.8 hours	\$1,283,040	>1.2M	\$51,840
99.9	3	43.2 minutes	719.28 hours	\$1,294,704	>1.2M	\$11,664
99.99	4	4.32 minutes	719.928 hours	\$1,295,987	>1.2M	\$1,166.4
99.999	5	25.92 seconds	719.9928 hours	\$1,295,987	>1.2M	\$116.64

forced as a strong requirement across all teams that introduce changes.

Dedicated canary environment. Every critical production application should have a dedicated canary environment as a prerequisite. It should be an exact replica of the production environment. This allows for testing user-facing impact such as load and performance.

Phased rollouts help reveal unforeseen issues (those not uncovered by tests) that are discovered only in production. This provides the agility to roll back the changes quickly and minimize the impact.

Rollbacks and restore. Another key discipline is to ensure every change can be rolled back. It is particularly important to understand the dependency graph of the change and ensure an atomic rollback. This is difficult in complex systems, but in such cases having a clear restore point is key for most critical changes.

Error budgets are a simple concept. Every service has a target SLO, and if it exceeds that SLO, then that positive delta of uptime becomes the budget to use in pushing any changes or releases. This is a powerful concept explained in depth in the SRE book.¹ Sharing this rigor with your application development team is a good way to ensure service reliability.

Outages and incidents. No matter how reliable a system is, you should anticipate and prepare for a disaster. Rather than solving for no outages, which is impractical, the focus should be on effectively managing the outage (minimizing downtime) and learning from it, so the same patterns don't repeat.

Resiliency testing. The goal here is to stress test application resiliency by breaking the system, observing the effects of the breakage, and subsequently improving the reliability of the application.

Incident preparedness. The SRE team should periodically run fire drills to practice incident management that involves extensive coordination with partner teams, timely communication to stakeholders, and restoring the service as soon as possible. Responding to and handling an actual incident without this preparation can reduce the speed and effectiveness of restoring the service.

Learning from outages. A repeated outage is not an outage anymore; it is a

mistake. For every outage there should be a thorough post-mortem that clearly identifies the root cause of the outage and focuses on what went wrong and what can be improved going forward. It is critical for enterprises to foster a blameless post-mortem culture that focuses on improving the reliability of the application.

The Future of Enterprise Reliability

Over the past few years, cloud platform providers have increasingly focused on enterprises, offering a suite of secure, reliable, and cost-effective products from highly scalable compute, storage, and networking services to modernized managed offerings such as container as a service (Kubernetes), serverless, and DBaaS. In addition, cloud providers are delivering advanced services in the realms of AI (artificial intelligence), ML (machine learning), and big data, opening a wide range of possibilities for enterprises to rethink and transform their business verticals.


This shift represents a tremendous opportunity for enterprises to embrace and adopt the cloud. Undertaking such a large-scale migration, however, introduces a new challenge: How can enterprises adapt and rapidly evolve without reducing their reliability?

Cloud migration strategy. Enterprises typically have complex business requirements, so a lift-and-shift strategy to migrate 100% of their workloads to a single cloud provider may not be feasible. A hybrid cloud environment provides the flexibility for workloads to operate seamlessly across both public and private cloud environments. This approach greatly simplifies the cloud adoption strategy and provides a controlled environment that ensures a predictable level of reliability throughout the transition to the cloud.

Enterprises that thoughtfully embrace the hybrid cloud strategy have less risk in terms of overall reliability and have a faster path to cloud transformation. Investing in a common application platform, coupled with the adoption of technologies such as Kubernetes (<https://kubernetes.io/>), Istio (<https://istio.io/>), and serverless computing (https://en.wikipedia.org/wiki/Serverless_computing), provides the flexibility to operate workloads,

agnostic to the cloud provider. Technologies such as the GCP (Google Cloud Platform) Anthos platform (<https://cloud.google.com/anthos/>) can also help enterprises expedite their transition to the cloud in a reliable and efficient manner.

VEC ecosystem. Developing a strong relationship among vendors, enterprises, and cloud providers is pivotal to the future of enterprise reliability. Cloud providers need to motivate software vendors, through partnership programs, to modernize third-party software embracing cloud-based technologies and building certified multicloud-compliant software offerings. This VEC (vendor-enterprise-cloud) ecosystem coupled with the technological shift will bring a rapid transformation shaping the enterprise domain.

Maintaining enterprise reliability is a continuous process that is in a crucial moment with the advent of the cloud. The next decade will be the era of large-scale enterprise transformations leveraging cloud capabilities, and only those enterprises that grasp the discipline of reliability engineering will be able to transform successfully into the realm of cloud-based enterprise computing. 

Related articles on queue.acm.org

Toward Software-defined SLAs

Jason Lango

<https://queue.acm.org/detail.cfm?id=2560948>

Enterprise Software as Service

Dean Jacobs

<https://queue.acm.org/detail.cfm?id=1080875>

Why Cloud Computing Will Never Be Free

Dave Durkee

<https://queue.acm.org/detail.cfm?id=1772130>

References

1. Jones, C., Wilkes, J., Murphy, N. and Smith, C. Service-level objectives. *Site Reliability Engineering*. B. Beyer, C. Jones, J. Petoff, and N.R. Murphy, eds. O'Reilly Media, 2016; <https://landing.google.com/sre/sre-book/chapters/service-level-objectives/>.
2. Treynor, B., Dahlin, M., Rau, V., Beyer, B. The calculus of service availability. *acmqueue 15, 2* (2017); <https://queue.acm.org/detail.cfm?id=3096459>.

Sanjay Sha is an SRE Manager at Google. With more than 14 years' experience running several large-scale systems at Google, he currently leads the Enterprise domain, managing SRE teams supporting Google's key business verticals. He is currently working on the Corp to Cloud initiative to run Google's internal enterprise workloads on GCP.

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

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

Industry's dreams and fears for this new technology.

BY SCOTT RUOTI, BEN KAISER, ARKADY YERUKHIMOVICH,
JEREMY CLARK, AND ROBERT CUNNINGHAM

Blockchain Technology: What Is It Good For?

IN 2008, AN author using the pseudonym Satoshi Nakamoto wrote a white paper describing Bitcoin, a new decentralized cryptocurrency.⁸ Unlike past attempts at forming a cryptocurrency—attempts that relied on preestablished trusted entities for the system to operate correctly—Bitcoin's design runs on the open Internet, with no one in charge, while maintaining tight security. While the building blocks of Bitcoin were not novel, the composition of these properties into a single system was a meaningful contribution,⁹ and Bitcoin became the first cryptocurrency to achieve widespread attention.

In response to Bitcoin's success, the technology was quickly dissected to understand how it works and what is new about it. Its most innovative component

has been labeled blockchain technology, a decentralized mechanism for participants to agree upon data and computation.

Technology news commonly leaves the cheery impression that blockchain technology reduces or even completely eliminates the need for trust. The use cases of such an innovation stretch the imagination. Occasionally, there is a contrarian take.¹²

The truth is, trust is complicated. Blockchain technology does eliminate specific, narrow reliances on trust, but it also requires new assumptions that might be better or worse for specific use cases. Thus, there are not many single-sentence talking points that will be accurate about blockchain technology's efficiency, security, cost, and so on.

It is clear this technology requires a more nuanced discussion. Business executives, government leaders, investors, and researchers frequently ask the following three questions: What exactly is blockchain technology? What capabilities does it provide? What are good applications?

The goal of this article is to answer these questions thoroughly, provide a holistic overview of blockchain technology that separates hype from reality, and propose a useful lexicon for discussing the specifics of blockchain technology in the future.

Methodology. This discussion is based on a rigorous textual analysis of nonacademic sources (hereafter referred to as industry white papers), including but not limited to the technology, financial, and health care sectors—from startups to SMEs (small and medium-sized enterprises) to Fortune 500 corporations. Academics have already systematized deep technical aspects of blockchain technology. Our analysis systematizes a distinct set of knowledge—the institutional knowledge in industry—which helps complete the picture. What industry might lack in technical knowledge, it makes up for in understanding market needs, the true costs of deployment, the intricacies of existing and legacy systems,



stakeholders and their competing interests, and the regulatory landscape.

While there is valuable information to be learned from industry, analyzing these sources also brings challenges, including imprecise terminology and errors in knowledge; inclusion of hype; and researcher bias.

The well-established research method known as *grounded theory*^{3,15} was used to rigorously analyze the data in a way that directly addresses each of these three limitations. Grounded theory helps researchers identify high-level themes and processes within qualitative data sources generated by humans and filled with imprecise terminology and descriptions. Additionally, grounded theory limits the impact of researcher bias, ensuring the themes and processes are derived from the data and not from the researchers' preconceived notions of what the data says.

Materials. The following methods were used to gather materials:

- ▶ Following RSS feeds that track news and publications related to blockchain technology.

- ▶ Downloading materials published by blockchain consortia (for example, Hyperledger, the Decentralized Identity Foundation).

- ▶ Reviewing documents from major accounting firms, banks, and tech companies.

- ▶ Browsing news articles and blog posts related to blockchain technology.

- ▶ Reviewing submissions to the ONC (Office of the National Coordinator of Health Information Technology) for the *Blockchain in Health Care Challenge*.

In reviewing these materials, we also followed references and included those documents if relevant. In total, 132 documents were collected and split into three categories:

- ▶ *High-level overviews.* Often prepared by investment firms, these overviews of blockchain technology provided an enumeration of efforts at using blockchain technology in practice.

- ▶ *System designs.* These papers proposed ways blockchain technology could be used in a specific system (or, less frequently, reported on a pilot study).

- ▶ *Commentaries.* These generally shorter documents discussed specific facets of blockchain technology in greater depth than seen in other documents.

Analysis. Four members of our group participated in the analysis of collected documents. We continued gathering and reviewing documents until each felt that the last three to five documents read revealed no new information; this is a commonly accepted stopping criterion in grounded theory that ensures all core (not one-off) ideas have been identified. A technical companion to this article contains the complete mythological details: the type of coding used at each stage and theory generation.¹¹

Results. The analysis revealed a set of 75 interconnected concepts that define blockchain technology. These concepts are grouped into five broad categories:

- ▶ *Technical properties*—the components that make up blockchain technology. Examples include decentralized governance, a consensus protocol, and an append-only transaction ledger.

- ▶ *Capabilities*—the high-level features provided by the technical properties. Examples include automatic executions of code (such as, smart contracts), internal auditability, and access control.

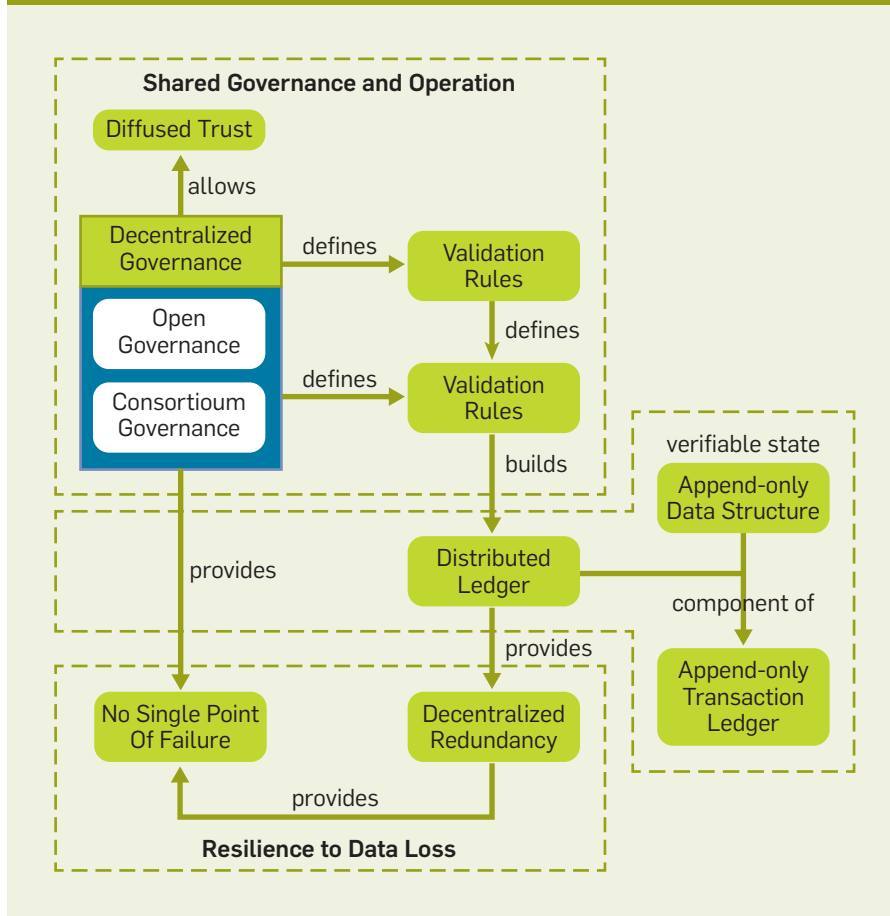
- ▶ *Technical primitives*—the building blocks used to construct the technical properties and capabilities of blockchain technology. Examples include timestamps, hash chains, and peer-to-peer communication.

- ▶ *Use cases*—classes of systems that the literature identified as applications of blockchain technology. Examples include cryptocurrencies, supply-chain management, and identity management.

- ▶ *Normative properties*—representative of what people hope to achieve using blockchain technology. Importantly, these properties are not provided by the use of blockchain technology, as the technical properties and capabilities are. In general, normative properties relate strongly to the hype surrounding blockchain technology. Examples include public participation, trustlessness, and censorship resistance.

While the concepts defining blockchain technology are divided into these

Figure 1. Technical properties for blockchain technology.



five categories, individual concepts are highly interconnected, both inter- and intra-category. This lends credence to the notion that blockchain technology is a cohesive whole, with each of its component concepts serving a purpose in the overall technology. This article focuses on some interesting and useful highlights from the full analysis, while interested readers are directed to the technical companion article and data files for the rest.¹¹

Technical Properties

The first broad category of blockchain technology concepts is technical properties, subdivided into three key groups: shared governance and operation, verifiable state, and resilience to data loss. Figure 1 shows the relationships among them.

Shared governance and operation. Blockchain technology addresses the scenario in which a collection of entities (for example, individuals or companies) want to participate in a communal system but do not trust each other or any third party to operate the system single-handedly. By deciding on the system details (governance) and then deploying networked devices (referred to as *miners*) to run the system, each entity can be assured of correct operation. If a small number of the miners become compromised (within bounds that are highly nuanced), the uncompromised miners can reject the malicious actions taken by the compromised miners and preserve the correct operation of the system. In this regard, blockchain technology provides diffused trust, in which the collective of miners is trusted. This is often given the misnomer trustlessness—trust still exists but has been diffused.

Shared operation is enabled by consensus protocols, which are used by the miners to agree upon which operations—known as transactions—will be executed by the system. A transaction is sometimes what it sounds like, a financial transaction that moves a unit of value from one account to another, but more generally it is a request that a certain function (which itself may be stored in the blockchain system) be executed on a set of inputs given in the transaction. Shared governance exists over what valid transactions look like (for example, the

transaction is digitally signed by the sender) and how the system functions (for example, the size and number of operations in a transaction are less than a certain bound). Shared operation means every miner validates transactions, and consensus among miners is used to ensure only correct outputs of valid transactions are written to the blockchain system (invalid or incorrectly executed transactions can be proposed but will be rejected by the miners).

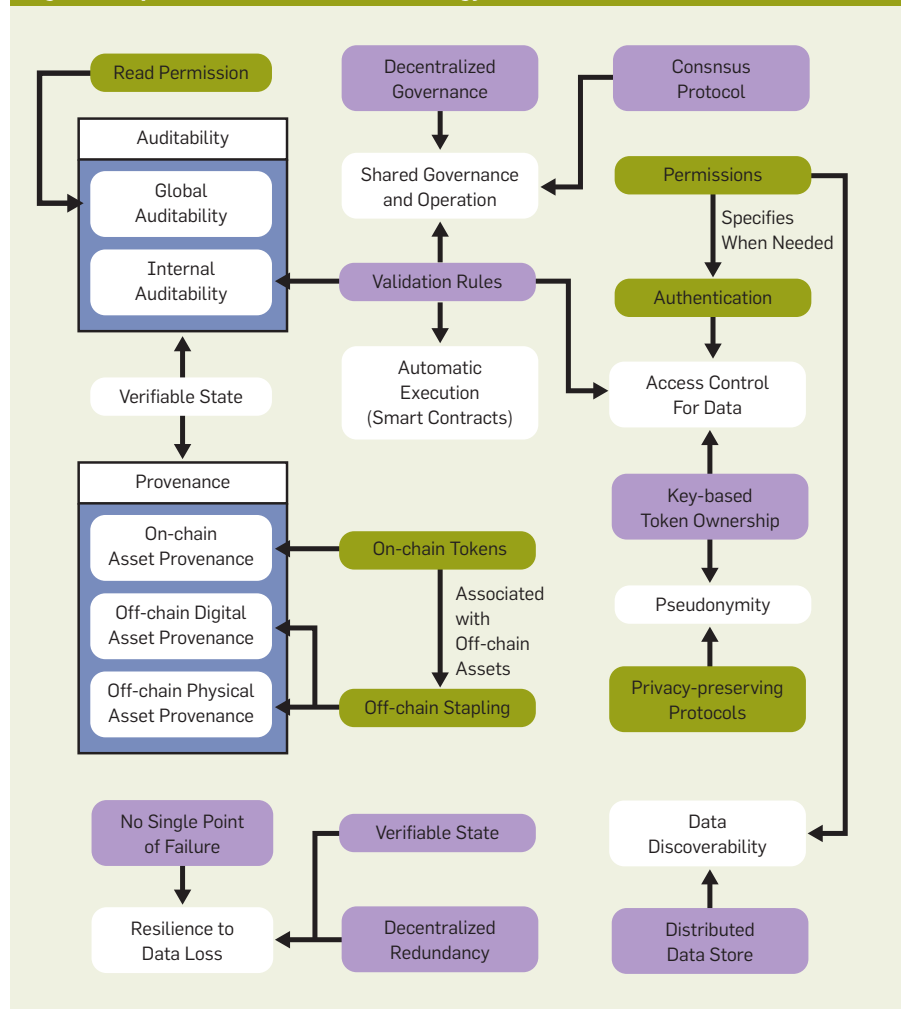
Blockchain systems can be categorized based on who is allowed to act as a miner:

► *Open governance* (that is, permissionless blockchain systems). Any party that is willing to participate in the consensus protocol is allowed to do so, regardless of their identity. To prevent a Sybil attack, in which an attacker creates multiple identities in order to influence the results of the consensus protocol, open governance system rely

on consensus protocols where miners prove ownership and/or expenditure of some costly, finite resource. Proof of work (demonstrating ownership of computing resources) and proof of stake (staking digital assets owned on the blockchain system) are two common methods.^{2,5}

► *Consortium governance* (that is, permissioned blockchain systems). Participation in the consensus protocol is limited to miners approved on a whitelist defined at system initialization. If this set never changes, it is known as a static consortium. Alternatively, in an agile consortium miners change over time, either based on the rules of the system (for example, random selection) or through consensus by the existing miners. Because each miner in a consortium is mapped to a known identity, a traditional byzantine fault-tolerant protocol (from distributed systems) can be used. This sidesteps the wasteful resource expenditure of

Figure 2. Capabilities for blockchain technology.



Sybil-resistant protocols such as proof of work.^{2,5}

For each type of governance, there is a need to reward correct participant behavior. The first type of incentive is intrinsic—such as, miners maintain the system faithfully because they derive value from using it. Next, on-chain incentives exist when the blockchain system provides direct benefits to miners for faithful execution (for example, minting currency and giving it to the miners). Finally, off-chain incentives are those not managed by the blockchain system—for example, contractual obligations or individual reputation. Importantly, off-chain incentives apply only to consortium governance, as they inherently rely on knowing the identity of the miners.

Verifiable state. Entities adopt blockchain technology because they want their trust to be rooted in the system (that the current state of the system accurately reflects the transactions that the consensus protocol allowed to execute in the past). To enable this trust, miners write all transactions to a cryptographically verified append-only ledger,¹⁴ providing full system provenance and allowing miners (or outside parties) to audit the system's current state and past operations.

In many systems, including Bitcoin, this ledger is colloquially referred to as the blockchain (we avoid using this term for the ledger to avoid confusion with holistic references to blockchain technology). In the ledger, all transactions are strictly ordered, and after consensus is reached (and as long as it is maintained) this ordering never changes and transactions are never removed. Thus, all miners who begin at the first entry (called the genesis block) will process all the transactions in the same order and reach the same current state for the entire system.

Resilience to data loss. If the ledger were stored in a single location, deletion or modification of data could be detected by all parties, but there would be no guarantee that the data could be restored. With blockchain technology, the content of the ledger is replicated among all miners to address this single point of failure. When data does need to be restored—for example, if an individual miner's ledger is corrupted or a new miner joins—the replicated data

can be verified to ensure it correctly represents the system state.

Some blockchain systems try to limit the amount of data any given miner needs to replicate by segmenting the data and assigning miners to handle governance and operations for only a subset of the system. This is known as *sharding*, with individual segments of the data called shards. Sharding can drastically reduce the amount of data that miners need to store, while also increasing the performance of the consensus protocol, which often scales based on the number of miners. Still, sharding adds complexity to auditing the system as a whole. Additionally, by reducing the number of miners responsible for any given transaction, sharding reduces the number of miners an adversary would need in order to deceive an end client about a transaction's existence.

Capabilities

Capabilities define the high-level functionality that can be achieved by using blockchain technology in a system's design. Blockchain technology's three core capabilities were described in the preceding section: Shared governance and operation; verifiable state, and, resilience to data loss. In coding, we identified 11 additional capabilities. (In Figure 2 these capabilities are color coded: purple represents capabilities; blue, technical properties; and green, technical primitives. Arrows indicate that the destination depends on the source.)

Provenance and auditability. Blockchain systems provide a complete history of all transactions that were approved by the consensus process (that is, full-system provenance). This information can be used by the miners to audit the system and ensure it has always followed the appropriate rules. Additionally, this information can be used by nonminers to verify that the system is being governed and operated correctly.

If transactions are used to store information regarding digital or real-world resources, then the resources must be stapled to on-chain identifiers. The provenance information for the blockchain system can also be used to provide audit information for those resources. This can be used

to track physical off-chain assets (for example, for supply-chain management), digital off-chain assets (for example, copyrighted digital media), or digital on-chain assets (cryptocurrencies or data files).

Access control and pseudonymity. Data stored in a blockchain system may have limitations on which users can use it as an input to a transaction or modify it as part of the transaction. For example, a financial asset should be a valid input to a transaction only if the owner of that asset approves its use. One approach to providing this functionality is storing access control lists (ACLs) in the ledger and having the appropriate users prove their identity to the miners (for example, using Kerberos or OAuth 2.0) as part of the transaction validation process.

More commonly, access control in a blockchain system is implemented cryptographically: data is associated with a public key when it is created, and the ability to use or modify this data as part of a transaction is granted only to users who can prove knowledge of the corresponding private key (for example, by generating a signature that validates with the public key attached to the data). Ownership of the data can be expanded or transferred by associating it with a new public key.

Key-based (as opposed to ACL-based) ownership of data has another advantage: It allows for pseudonymous ownership and use of data. Still, this requires careful attention in the system design to use appropriate cryptographic techniques (for example, zero-knowledge proofs, mix networks, or secure multiparty computation) to avoid linking real-world individuals to their keys and actions. This remains an open problem.

Smart contracts. In a general-purpose blockchain system, a smart contract or decentralized application (DApp) can be deployed using a transaction that stores the code for a set of functions and the initial state of the contract. These functions can then be called in subsequent transactions. The functions themselves are executed by the miners, and outputs are verified through the consensus protocol. Any entity can execute any function, but the function might be programmed to fail if the conditions under which

it is called are not what the designer intended. The computational power of the scripting language that can be used to specify a function varies from system to system and there are many nuances to ensure functions can be executed by each miner deterministically in a timely fashion. Bitcoin is known for its limited scripting language that enables little beyond financial transactions, while Ethereum strives for highly verbose code capable of general computation.

Data discoverability. If users are allowed to read any record stored in a blockchain system, then it is possible to search for records of interest. This capability is nothing more than what is provided by having a read-only data lake, but it was still discussed frequently in the reviewed literature.

Challenges and Limitations

Our analysis reveals several challenges that need to be considered when developing systems that use blockchain technology.

Scalability and performance. Decentralized governance and operation incur three forms of overhead: The need to run a consensus protocol before state can be updated; the need to store the full system provenance; and, the need for each miner to store the ledger in its entirety. Furthermore, most of today's open governance blockchain systems are based on proof of work, which brings additional challenges. Users must acquire hardware and expend electricity to participate in consensus, the real-world cost of which can be tremendous. For example, it was estimated that as of April 2018 the energy consumed by Bitcoin miners alone was equivalent to the power usage of almost 5.5 million U.S. households.⁴

On-chain correctness. All executable code is subject to bugs, and smart contracts are no exception. The immutability of a blockchain's ledger exacerbates this challenge by impeding rollback of state changes, even those that are clearly malicious. Failure to act can be costly (for example, the DAO attack¹³), but so too can reversing transactions. If miners decide to roll back the ledger to erase a mistaken transaction, confidence in the blockchain system may be lost. The rollback system must be designed carefully, or there is



Trust is complicated. Blockchain technology does eliminate specific, narrow reliances on trust, but it also requires new assumptions that might be better or worse for specific use cases.



risk of further exploitation.¹ Alternatively, if miners can't agree on what to do about errant transactions, it could lead to a *fork*: the creation of two competing blockchain systems.

Off-chain stapling. Many blockchain systems manage off-chain assets by representing them on-chain using digital identifiers, or tokens. A major challenge for these applications is ensuring consistency between on-chain state and the off-chain reality it represents. When dealing with digital assets, consistency can be maintained by code; for example, a smart contract can track transference of ownership for a digital media license. For physical assets, real-world processes must be employed to ensure consistency. These processes are an obvious point of failure, as they rely on correct execution by trusted parties (something that blockchain systems are often deployed to remove). The end users must also be trusted, as they may be able to separate a token and sell it while keeping the asset, causing the token to be attached to an invalid asset (for example, fake goods in luxury markets).

Similar challenges arise when blockchain systems must track real-world events and information (for example, sports scores, Web requests). While such information can be provided by off-chain oracles, these are trusted entities that are difficult to audit.

Security. Because of their decentralized nature, blockchain systems are potentially vulnerable to a number of security threats. Coordinated attacks by a majority (or, often, even a large minority) of the miners can reorder, remove, and change transactions on the ledger. Additionally, blockchain systems are vulnerable to traditional network attacks such as denial of service or partitioning. Such attacks aim to lower the number of participating miners or fracture the network of miners to prevent consensus, lower the bar for attacks, or create an inconsistent state.

Privacy and anonymity. Data in a blockchain ledger is public (at least to all miners) in order to enable verification, meaning that sensitive data is inherently nonprivate. Confidentiality can be provided using a reference monitor that limits access (for non-miners) to data stored in a blockchain system based on access-control lists

stored in the ledger, but this introduces a trusted entity (the reference monitor). Alternatively, the data can be encrypted using advanced cryptographic techniques that allow miners to verify the correctness of encrypted transactions (for example, zero-knowledge proofs, secure multiparty computation, and functional encryption),⁷ though encrypting data limits auditability and the ability to have meaningful shared governance.

Extreme care must be taken when trying to build an anonymous blockchain system. While many existing blockchain systems provide a notion of pseudonymity in which users are identified by their cryptographic keys instead of by their real-world names, this does not provide true anonymity, as attacks that correlate transactions by the same pseudonyms together with other data external to the blockchain system can effectively deanonymize users.⁶

Usability. The availability of user-friendly developer tools varies significantly depending on the maturity of the blockchain platform. Some projects such as Ethereum have mature tools, while others have very little support. Many blockchain platforms are geared toward expert users and lack the experience-focused tools needed for easier use by nonexperts. A related challenge is that some blockchain systems require users to store, manage, and secure cryptographic keys; this requirement is known to be a significant impediment for most users.¹⁰

Legality and regulation. Some benefits claimed by blockchain systems cannot be attributed to the underlying technology, but rather to sidestepping the regulation and oversight that slows existing systems (for example, international payments or raising capital by selling virtual assets to investors). As regulators catch up, compliance is given priority. Blockchain technology is not directly regulated; firms are regulated based on how they use it. The most discussed areas of regulation are taxation, audited financial statements, transaction reporting (know-your-customer/anti-money laundering/anti-terrorist financing), securities law, banking, and custodianship. An extreme case of regulation is prohibition of cryptocurrencies or blockchain assets. At the time of writing, the largest

country to ban Bitcoin is Pakistan, and the largest country to prohibit wide categories of cryptocurrency use is China.

Use Cases

Industry and government can apply blockchain technology in a number of use cases that require shared governance, verifiable state, and/or resilience to data loss.

Financial use cases. It is well known that blockchain technology can be used to build cryptocurrencies; Bitcoin is a working example of this. Blockchain technology enables electronic transactions that are resilient even when large amounts of money are at stake. Bitcoin has notable drawbacks that include low scalability, high-energy consumption, and merely moderate privacy protections. A payment system using consortium governance can address the first two key challenges.

Asset trading. Financial markets allow the exchange of assets. They tend to involve intermediaries such as exchanges, brokers and dealers, depositories and custodians, and clearing and settlement entities. Blockchain-based assets—which are either intrinsically valuable or are claims on off-chain assets (material or digital)—can be transacted directly between participants, governed by smart contracts that can provide custodianship, and require less financial market infrastructure. Two key challenges are: Stapling for tokens that represent something off-chain (for example, equity in a firm or a debt instrument); and government oversight and regulatory compliance.

Markets and auctions. A central component of asset trading is the market itself—the coordination point for buyers and sellers to find each other, exchange assets, and provide price information to observers. Auctions are a common mechanism for setting a fair price; this includes double-sided auctions such as the order books in common use by financial exchanges. The key challenge for a decentralized market is that transactions are broadcast to the consensus protocol and thus nonconfidential, hindering privacy and enabling front-running.

Insurance and futures. Transactions can be arranged that are contingent on future times or events. Examples include a purchase of assets at

a future time for a locked-in price, an insurance payout for a fire, or action on a loan default. The key challenges are: Determining trustworthy oracles to report relevant off-chain events such as fires and exchange rates. (or limiting the contracts to on-chain events); and choosing between a design that locks up so much collateral it can settle all possible eventualities, or a leaner design where the counterparty promises to fulfill its obligations but there is the counterparty risk that it will not.

Penalties, remedies, and sanctions. Legal contracts anticipate potential future breaches and specify a set of penalties or remedies. With blockchain technology, remedies for likely outcomes could be programmed (these could be later overturned through traditional litigation). As with insurance and futures, oracles and counterparty risk are key challenges.

Data storage and sharing use cases. Blockchain technology can be used to track material assets that are globally distributed and valuable, and whose provenance is of interest. This includes standalone items such as artwork and diamonds, certified goods such as food and luxury items, dispersed items such as fleets of vehicles, and packages being shipped over long distances, which will change hands many times in the process. It also includes the individual components of complex assembled devices, where the parts originate from different firms. For heavily regulated industries such as airlines, and for military/intelligence applications, it is important to establish the source of each part that has been used, as well as a maintenance history (that is, its provenance). Blockchain technology provides a common environment where no single firm has the elevated power and control of running the database that tracks this information. Key challenges are the reliable stapling of data, confidentiality, and onboarding all the necessary firms onto the same blockchain system.

Identity and key management. Identities, along with cryptographic attestations about properties for those identities (for example, over 18 years of age, has a driver's license, owns a specific cryptographic key), can be maintained on a blockchain system. This is a spe-

cial case of asset tracking, where the “asset” is a person. The key challenges are the same.

Tamper-resistant record storage. The append-only ledger of a blockchain system can be used to store documents, including the history of changes to these documents. This use case is best suited for records that are highly valuable (such as certificates and government licenses), have a small data size, and are publicly available (as they will be replicated by all miners). If large and/or confidential documents need to be stored, a blockchain system might store secure pointers (that is, binding/hiding commitments) for the documents, while the documents themselves are stored in a different system.

Other use cases. *Electronic voting* is a challenging problem that is often asserted to benefit from blockchain technology’s properties. Shared governance could be used to ensure multiple parties (the government, nongovernmental organizations, international watchdogs) can work together to ensure an election is legitimate. Auditability is important in providing evidence to the electorate that the election was fair. Finally, the resilience of blockchain technology is important in preventing cyberattacks against the voting system. Voting on a blockchain system, however, has many challenges to solve: Blockchain systems offer no inherent support for secret ballots; electronic votes can be changed by the device from which they are submitted (undetectably if a secret ballot is achieved); cryptographic keys could be sold to vote buyers; and key recovery mechanisms would need to be established for lost keys.

Gambling and games. Gambling is already very popular on Bitcoin and Ethereum. Players can audit the contract code to ensure execution is fair, and the contract can use cryptocurrency to handle the finances (including holding the money in escrow to prevent losing parties from aborting before paying). This use case is best suited for gambling games that do not require randomness, private state, or knowledge of off-chain events.

Application

Ultimately, blockchain technology is not a panacea, but it is a useful tool

when the overhead is justified by the system’s needs. A good place to start is by posing the following questions:

1. Does the system require shared governance?

2. Does the system require shared operation?

If both answers to these questions are no, the overhead of blockchain technology is unnecessary. If both answers are yes, there is a good fit. If only one of the answers is yes—if only shared governance or shared operation is needed but not both—then two more questions should be considered:

3. Is it necessary to audit the system’s provenance?

4. Is it necessary to prevent malicious data deletion?

If auditability and data replication are critical, blockchain technology should be considered. This is because meaningful shared governance *and* operation require miners to audit the operations of others and to be able to recover data that a malicious miner might try to delete.

Even though blockchain technology does not solve all the problems that its proponents claim it does, it is nonetheless a meaningful technology that will continue to be used in industry and is deserving of further research and experimentation. ■

Related articles on queue.acm.org

Bitcoin’s Academic Pedigree

Arvind Narayanan and Jeremy Clark
<https://queue.acm.org/detail.cfm?id=3136559>

Research for Practice: Cryptocurrencies, Blockchains, and Smart Contracts

Arvind Narayanan and Andrew Miller
<https://queue.acm.org/detail.cfm?id=3043967>

A Hitchhiker’s Guide to the Blockchain Universe

Jim Waldo
<https://queue.acm.org/detail.cfm?id=3305265>

References

1. Avizheh, S., Safavi-Naini, R., Shahandashti, S.F. A new look at the refund mechanism in the Bitcoin payment protocol. In *Proceedings of Financial Cryptography and Data Security*, 2018; <https://arxiv.org/abs/1807.01793>.
2. Bano, S., Sonnino, A., Al-Bassam, M., Azouvi, S., McCorry, P., Meiklejohn, S., Danezis, G. SoK: Consensus in the age of blockchains. In *Proceedings of ACM Advances in Financial Technology*, 2019; <https://arxiv.org/pdf/1711.03936.pdf>.
3. Corbin, J., Strauss, A. Grounded theory research: Procedures, canons and evaluative criteria. *Zeitschrift für Soziologie* 19, 6 (1990), 418–427.
4. Digiconomist. Bitcoin energy consumption index, 2019; <https://digiconomist.net/bitcoin-energy-consumption>.
5. Garay, J., Kiayias, A. SoK: A consensus taxonomy in the blockchain era. *Cryptology ePrint Archive*, Report 2018/754; <https://eprint.iacr.org/2018/754>.

6. Goldfeder, S., Kalodner, H.A., Reisman, D., Narayanan, A. When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies. *Privacy Enhancing Technologies* 179–199; <https://www.petsymposium.org/2018/files/papers/issue4/popets-2018-0038.pdf>.
7. Kosba, A.E., Miller, A., Shi, E., Wen, A., Papamanthou, C. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *Proceedings of IEEE Symp. Security and Privacy*, 2016, 839–858; <https://ieeexplore.ieee.org/document/7546538>.
8. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system, 2008; <https://bitcoin.org/bitcoin.pdf>.
9. Narayanan, A., and Jeremy Clark, J. Bitcoin’s academic pedigree. *acmqueue* 15, 4 (2017); <https://queue.acm.org/detail.cfm?id=3136559>.
10. Ruoti, S. et al. A usability study of four secure email tools using paired participants. *ACM Trans. Privacy and Security* 22, 2 (2019), 13:1–13:33; <https://dl.acm.org/citation.cfm?id=3313761&preflayout=tabs>.
11. Ruoti, S., Kaiser, B., Yerukhimovich, A., Clark, J., Cunningham, R. SoK: Blockchain technology and its potential use cases. Technical report, 2019; <https://arxiv.org/abs/1909.12454>.
12. Schneier, B. There’s no good reason to trust blockchain technology. *Wired*, 2019; <https://www.wired.com/story/theres-no-good-reason-to-trust-blockchain-technology/>.
13. Siegel, D. Understanding the DAO hack. *Coindesk*, 2017; <https://www.coindesk.com/understanding-dao-hack-journalists>.
14. Tamassia, R. Authenticated data structures. In *Proceedings of European Symposium on Algorithms*, 2003, 2–5. Springer; https://link.springer.com/chapter/10.1007/978-3-540-39658-1_2.
15. Wolfswinkel, J.F., Furtmueller, E., Wilderom, C.P.M. Using grounded theory as a method for rigorously reviewing literature. *European J. Information Systems* 22, 1 (2013), 45–55; <https://link.springer.com/article/10.1057/ejis.2011.51>.

The majority of this work was completed while the authors (other than Jeremy Clark) were working at MIT Lincoln Laboratory.

Scott Ruoti is an assistant professor in the electrical engineering and computer science department at the University of Tennessee in Knoxville. His research includes using blockchain technology to build and secure noncryptocurrency systems, improving the security and accessibility of password managers and two-factor authentication, and helping software developers create secure software.

Ben Kaiser is a Ph.D. student in the Center for Information Technology Policy at Princeton University, Princeton, NJ. He previously worked on applied cryptography as a staff researcher at MIT Lincoln Laboratory and now focuses on issues surrounding disinformation and online speech.

Arkady Yerukhimovich is an assistant professor of computer science at George Washington University. Previously, he was a research staff member at the MIT Lincoln Laboratory. His recent research is focused on developing cryptographic protocols for secure computation and database search.

Jeremy Clark is an associate professor at the Concordia Institute for Information Systems Engineering in Montreal, Canada, where he holds the NSERC/RCGT/Catalaxy Industrial Research Chair in Blockchain Technologies. He collaborates regularly with government agencies and municipalities on voting and blockchain technologies.

Robert Cunningham is Associate Director for Cyber Assurance in the CERT division of the Software Engineering Institute, and an adjunct professor of cybersecurity at Carnegie Mellon University. Previously, he led a series of computer security groups at MIT Lincoln Laboratory. He regularly briefs the U.S. Government on technical matters related to computer security.

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

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

**Combining data from many sources
may cause painful delays.**

BY PAT HELLAND

Space Time Discontinuum

INCREASINGLY, CALCULATIONS ARE based on timed events originating at many sources. By comparing, contrasting, joining, and noodling over these inputs, you can derive some interesting results. If the inputs to these calculations come from disparate computers (or sensors), you can't always be sure of how quickly the information will propagate. Hence, you can't be sure when you will get an answer to the requested calculation.

If you can't promise when you will get the answer, what are you going to do? You can wait to get the perfect answer, or you can give an imperfect answer more promptly by basing it on partial knowledge.

How can you meet your SLAs (service-level agreements)? Sometimes, it's not an easy and seamless continuum of options but more of a discontinuum.

Events and time. In many systems, events come with a timestamp. These may come from temperature sensors, motion detectors, factory floors, your cable box, your security system, automated tollbooths on the freeway, and much more. One source of information

that is increasingly important is the monitoring of events in datacenters. These events may be used by an automated management system or by humans in complex error spelunking. It is common to try to see which events happened close in time to another particular event. Patterns in time proximity are essential to controlling these complex environments.

Events and space. Now, all this is cool except when the information you want is "over there." In a distributed system, if the stuff is not "here," it's "over there." When something is far away and over there, it may take a loooooong time to get information in and out. It can be kind of like when the only road up the canyon gets washed out in a flood. I refer to remote nodes as being "over yonder" whether they are in the same rack in the datacenter or thousands of miles away.

Space, Distance, and Traffic Jams

If it's over there with an *open* queuing network, it will usually get here in a timely fashion. Almost all modern networks are open queuing networks. In such a network, there's no practical limit to the amount of stuff that may try to share the network. Highway 101 through San Francisco is an open



queuing network. It works great until it doesn't.

Right now, I'm writing this on an airplane. I'm glad the modern fly-by-wire systems use *closed* queuing networks. This type of network carefully manages the messages allowed in. By ensuring the network never ingests more work, it can ensure a bounded time for the network to digest the work it already has. Assuming no hardware faults (or a bounded number of hardware faults), the closed queuing network can do what is requested within a specified period of time. I don't want a traffic jam on the fly-by-wire system controlling the flaps on the plane. I'm grateful for a closed queuing network.

In datacenters, on the other hand, you don't see closed queuing networks. Instead, you see a freeway that usually has a lot of lanes. It mostly works OK.

Reviewing our queuing. Delays in propagating events are not just caused by the network. It's common



for events to get pushed into some queuing system as they are published. The published events are then consumed by one or more subscribers. These subscribers typically enqueue the event in yet another queue (ad nauseam) as the event rattles its

way through the datacenter or Internet trying to seek its destination like a fly buzzing around a light on the back porch.

Misconstruing our queuing. Each of these queuing systems presents a whole new opportunity for delay. It's almost as if the event gets frozen in time and, hopefully, gets defrosted within the desired time window. Each stage of these queuing systems is very likely to



propagate the event quickly. Each time you add a "very likely to propagate the event quickly," you also add a "sometimes it will be slow."

Not only can queuing systems be slow, they rarely have end-to-end guarantees. Unless the final point of consumption is coordinated with the originating sender, the event will sometimes go kablooeey and never arrive.

Service levels are a key component of engineering a system for use by a customer. These may deal with system performance, system availability, or quality of the returned results.^{4,5}

► **SLA** (service-level agreement). This is a contract with the customer. It may list financial compensation if you don't meet the numbers. It may be an understanding with your boss that you will miss part of your bonus if things go awry.



► **SLO** (service-level objective). This is your internal target for the metric in question. You always want your SLO to be better than your SLA (or you are not trying to please your customer).

► **SLI** (service-level indicator). This is what you actually measure against the metric. Hopefully, your SLI is better than your SLO, which is better than your SLA.

To ensure many different sources of events are combined, you should leave some room in your SLA to allow for slowness in getting the source data from all those sources. The more sources, the more likely you will blow your SLA as one or more of them blow their SLAs.

Managing Tail Latencies in Idempotent Operations

One of my favorite articles is "The Tail at Scale," by Jeffrey Dean and Luis André Barroso.¹ In this article, the authors outline how a service requiring a whole bunch of inputs can use timeout-based retries to dramatically improve the probability of getting all the inputs needed to meet the desired 99.9% SLA for response. In this case, sufficient replicas exist to get the answer without loss of quality, simply by retrying and consuming more resources. Key to this

is that it is OK to retry the laggard requests because they are idempotent. It does not cause harm to do them two or more times.

Knowing you can't know. The more complex the set of inputs, the more likely you won't see everything in a timely fashion. The more complex the store-and-forward queuing, the more likely stuff will arrive too late or not at all. The more distant the sources of your inputs, the more challenges you may have.

As we have seen, sometimes it can be effective to retry the request for input. In particular, in some systems, retrying can ensure all the inputs are available quickly.


In other systems, the inputs are not simply fetched but are rattling their way through queues similar to Highway 101 through San Francisco. In these environments, the processing probably has to simply cut off with what it has and do the best it can. This means you can't guarantee the stuff is ready when you want it.

So, if you know you can only *probably* know, what's the plan?


Approximating queries. There is some fun new work describing analytics with approximate answers. By expressing sampling operators, some systems can provide really good answers based on a small subset of all the inputs one would normally examine.^{2,3} In the cited systems, there is more focus on sampling for performance when everything is working and timely. Still, it's quite similar to what you would do to build systems that return answers based on what is available in a timely fashion.

Returning partial answers. Many systems work to give *some* answer in a timely fashion even if they are wounded. Many sophisticated websites will dribble out partial answers to the browser as they become available: The text for a product description may arrive before the product image; the product image may arrive before the user reviews. This decoupling yields a faster overall result and, in general, a more satisfied user.

When dealing with answers relying on data from a distance, it's important to consider how to decouple results and, where possible, return a degraded answer quickly when and if that



When dealing with answers relying on data from a distance, it is important to consider how to decouple results and, where possible, return a degraded answer quickly when and if that best meets the needs of the business.




best meets the needs of the business. What does it take to keep on going?

As the Black Knight from “Monty Python and the Holy Grail” says, “’Tis but a scratch” (<https://www.youtube.com/watch?v=ZmInkxbvlCs>; thank you to Peter Voss hall for raising the Black Knight in discussion when we were both younger many years ago).

The Disconcerting Discontinuum

Back when you had only one database for an application to worry about, you didn't have to think about partial results. You also didn't have to think about data arriving after some other data. It was all simply there.

Now, you can do so much more with big distributed systems, but you have to be more sophisticated in the trade-off between timely answers and complete answers. The best systems will adapt and interpret their problems as, “’Tis but a scratch!” 

Related articles on queue.acm.org

The Calculus of Service Availability

Ben Treynor, Mike Dahlin, Vivek Rau and Betsy Beyer
<https://queue.acm.org/detail.cfm?id=3096459>

Toward Higher Precision

Rick Ratzel and Rodney Greenstreet
<https://queue.acm.org/detail.cfm?id=2354406>

A Lesson in Resource Management

Kode Vicious
<https://queue.acm.org/detail.cfm?id=2523428>

References

1. Dean, J. and Barroso, L.A. The tail at scale. *Commun. ACM* 56, 2 (Feb. 2013), 74–80; <https://dl.acm.org/citation.cfm?id=2408794>.
2. Hall, A., Tudorica, A., Buruiana, F., Hofmann, R., Ganceanu, S. and Hofmann, T. Trading off accuracy for speed in PowerDrill. In *Proceedings of the Intern. Conf. Data Engineering*, 2016; <https://ai.google/research/pubs/pub45682/>.
3. Kandula, S., Lee, K., Chaudhuri, S. and Friedman, M. Experiences with approximating queries in Microsoft's production big-data clusters. In *Proceedings of the VLDB Endowment* 12, 2 (2019), 2131–2142; <http://bit.ly/2nUzkXL>.
4. Moguls, J.C., Isaacs, R., Welch, B. Thinking about availability in large service infrastructures. In *Proceedings of the 16th Workshop on Hot Topics in Operating Systems* 2017, 12–17; <https://dl.acm.org/citation.cfm?id=3102980.3102983>.
5. Moguls, J.C. and Wilkes, J. 2019. Nines are not enough: meaningful metrics for clouds. In *Proceedings of the Workshop on Hot Topics in Operating Systems*, 2019, 136–141; <https://dl.acm.org/citation.cfm?id=3321432>.

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/owner.
Publication rights licensed to ACM.

ACM Computing Surveys (CSUR)

2018 JOURNAL
IMPACT FACTOR:
6.131

Integration of computer science and engineering knowledge



ACM Computing Surveys (CSUR) publishes comprehensive, readable tutorials and survey papers that give guided tours through the literature and explain topics to those who seek to learn the basics of areas outside their specialties. These carefully planned and presented introductions are also an excellent way for professionals to develop perspectives on, and identify trends in, complex technologies. Recent issues have covered image understanding, software reusability, and object and relational database topics.

Both surveys and tutorials must develop a framework or overall view of an area that integrates the existing literature. Frequently, such a framework exposes topics that need additional research; a good CSUR article can fill such a void, but that is not its major purpose. Basically, a CSUR article answers the questions, "What is currently known about this area, and what does it mean to researchers and practitioners?" It should supply the basic knowledge to enable new researchers to enter the area, current researchers to continue developments, and practitioners to apply the results.

For more
information
and to submit
your work,
please visit:

csur.acm.org



Association for
Computing Machinery

DOI:10.1145/3362068

Edge computing holds great promise, and almost as many challenges in deployment.

BY SAURABH BAGCHI, MUHAMMAD-BILAL SIDDIQUI, PAUL WOOD, AND HENG ZHANG

Dependability in Edge Computing

EDGE COMPUTING IS the practice of placing computing resources at the edges of the Internet in close proximity to devices and information sources. This, much like a cache on a CPU, increases bandwidth and reduces latency for applications but at a potential cost of dependability and capacity. This is because these edge devices are often not as well maintained, dependable, powerful, or robust as centralized server-class cloud resources.^a

This article explores dependability and deployment challenges in the field of edge computing, what aspects are solvable with today's technology, and what aspects call for new solutions. The first issue addressed is failures—both hard (crash, hang, and so on) and soft (performance-related)—and real-time constraint violation. In this domain, edge computing bolsters real-time system capacity through reduced end-to-end latency. However, much like cache misses, overloaded

or malfunctioning edge computers can drive latency beyond tolerable limits. Second, decentralized management and device tampering can lead to chain of trust and security or privacy violations. Authentication, access control, and distributed intrusion detection techniques have to be extended from current cloud deployments and need to be customized for the edge ecosystem. The third issue deals with handling multi-tenancy in the typically resource-constrained edge devices and the need for standardization to allow for interoperability across vendor products.

We explore the key challenges in each of these three broad issues as they relate to dependability of edge computing and then hypothesize about promising avenues of work in this area.

For the purpose of this article, we consider as edge devices, those that are in the premises of the end user (the home or the industrial campus) as well as those are outside the premises, say at the edge of the Internet (for example, content distribution nodes at the edge) or of the cellular network (for example, base station). This definition is consistent with prior use of the term,^{2,15} though it is wider than some other prior usages.⁶ In terms of geographical spread of the coordinating edge devices, our definition of edge could span

» key insights

- **The scale of highly interconnected, real-time devices places immense pressure on existing best-effort communications infrastructures, leading to dependability concerns especially as deployments scale up.**
- **Edge computing provides a mechanism for relieving communications pressures by moving computation closer to the sensors and control loops, but it introduces new failure modes when edge nodes go down.**
- **Standardization, scalable authentication, and multitenancy techniques for edge computing devices will improve dependability, but ultimately application developers must understand how large-scale edge deployments may fail and prepare adequate contingencies for their use cases.**

^a Terminology: We distinguish between two classes of devices the client devices and the edge computing devices, or simply edge devices. When used without qualification, a device refers to a client device.



IMAGE BY ANDRZJ BORYS ASSOCIATES, USING PHOTO BY DAVID MG

from a small number of edge devices deployed in a neighborhood, those deployed in cellular base station within a city, to a citywide deployment. Local device-level computation is offloaded to nearby edge computing devices (foglets, cloudlets, among others) whenever local processing is either inadequate or costly, or the computation relies on non-local information. For example, a long-enough voice snippet from a phone can be processed at a cell tower rather than on a local device (mobile edge computing), both saving battery and reducing end-to-end laten-

cy due to processing speed differences. In contrast with traditional heavily centralized cloud computing, the edge computers act as a distributed computing infrastructure, providing increased bandwidth and reduced latency but with limited resources when compared with a central cloud.

The edge paradigm supports the large scale of the Internet of Things (IoT) devices, where real-time data is generated based on interactions with the local environment. This complements more heavy-duty processing and analytics occurring at the cloud level.

This structure serves as the backbone for applications, such as augmented reality and home automation, which utilize complex information processing to analyze the local environment to support decision making. In the IoT domain, functional inputs and outputs are physically tied to geographically distributed sensors and actuators. If this data is processed in a central location, immense pressure will be placed on “last mile” networks, and cloud-leveraged IoT deployments will become impractical (see the sidebar “Current Edge Deployment”).

Current Edge Deployment

An example of a current edge deployment, albeit not with the richness of factors described in this article, is the 4G radio access network (RAN) that uses mobile edge computing (MEC) to better deliver content and applications to end users.¹¹ It can adapt the service delivery according to radio link load and avoid long distance transmission by using local content caching. MEC was approved as a formal specification by the European Telecommunications Standard Institute in 2014 and there is ongoing activity in standardizing it. Two popular current use cases with this edge technology are to provide localized video at a stadium or concert venue, and asset tracking in a large enterprise with enterprise small cell networks.

Without drastic network improvements, which seem unlikely in the mid-range future,¹⁷ edge computing is likely to become a cornerstone of IoT. We see there has been a shifting of the envelope of local versus edge computing, based on two dimensions—first, as more demanding applications arise (voice processing to video processing to augmented reality) and second, as the locally available resources increase (processing, storage, networking). The first drives some processing toward the edge (and further, toward the cloud) while the latter drives processing to move closer to local devices. In the context of edge computing dependability, we focus on the five aspects that we deem most significant: large scale, low-latency or soft real-time requirements, authentication and physical security, multi-tenancy on the edge devices, and standardization.

Resiliency Challenges

Applications that benefit from edge computing typically have requirements for low latency and generate high-bandwidth data streams. Two canonical examples are provided by IoT devices and augmented reality (AR) applications. With this model, we examine the resiliency challenges that are posed by edge computing applications.

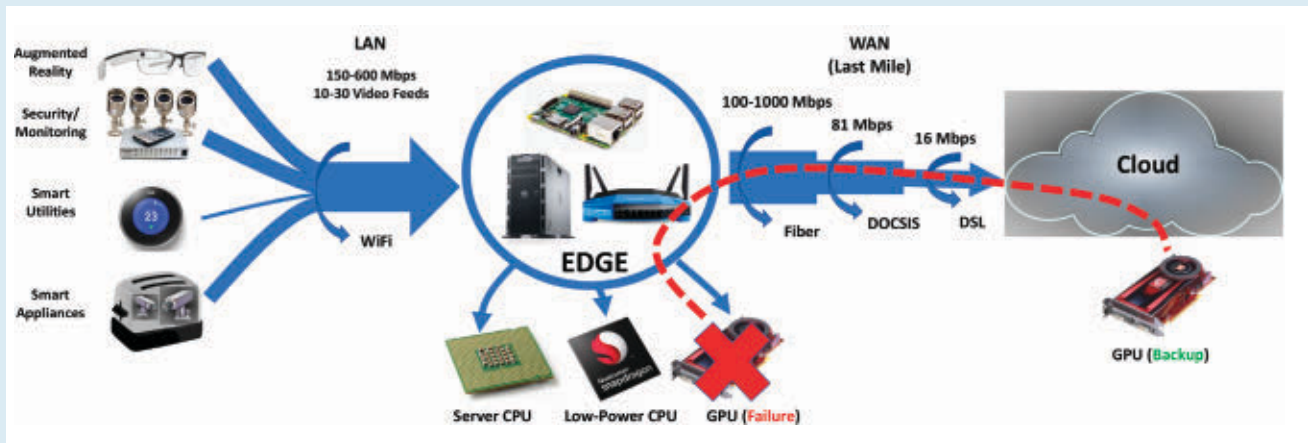
Large scale. Since edge computing applications are in their infancy, many current design decisions seem reasonable at a small scale. However, many practical challenges arise as the scale of edge applications grow, both in terms of the number of client devices and the amount of data being generated by them. The IoT, propelled by low-cost wireless electronics and ease of integration, is greatly increasing the number of addressable computers and the amount of environmental data available for transmission on the

Internet. Shared computing utilities, especially networking resources, can quickly become saturated by scale out of data-intensive applications, and protocols can fail to deliver results in a timely manner when algorithmic complexity is super-linear in terms of the number of endpoints (See the sidebar “Networking Challenges”).

Network impact. Cloud-based computing supports scalability by incrementally adding resources to the computing environment as new devices enter service. Practically, scale of this nature is more easily achieved in centralized datacenters where new server crates can be parked and connected to existing infrastructure. With increasing scale of IoT, offloading all processing to the datacenter becomes infeasible since network operators rely on average case capacity for deployment planning, and network technologies have not kept up with the growth of data. A fixed video sensor may generate 6Mbps of video 24/7, thus producing nearly 2TB of data per month—an amount unsustainable according to business practices for consumer connections, for example, Comcast’s data cap is at 1TB/month and Verizon Wireless throttles traffic over 26GB/month. For example, with DOCSIS 3.0, a widely deployed cable Internet technology, most U.S.-based cable systems deployed today support a maximum of 81Mbps aggregated over 500 home—just 0.16Mbps per home. If

Figure 1. High-bandwidth edge services on the left generate dense video or less-dense environmental data that needs to be processed on the edge.

The high bandwidth of WiFi allows edge computers such as ARM-based Raspberry Pi’s, routers, and traditional tower servers to process the data. When there is a failure, the data can be re-routed to the cloud, but this re-routing is limited by the capacity of the WAN system.



12 users are interacting with AR in the area, the network would be saturated. With edge computing, all 500 homes could be active simultaneously but at a resilience cost: if each home has four active AR users, then only three of the 500 edge computers could fail simultaneously and fall back to cloud-based processing, due to the constraints of the last-mile network.

Lack of failover options. The first resiliency challenge becomes a lack of resources to fail over in case of a failure. In the cloud, individual resource availability becomes less significant due to the presence of hot spares in the same local network. In an edge environment, hot spares are not practical—a common deployment scenario is very lean with a single-edge device, such as a WiFi router providing all the edge services in a home. An alternative solution to this is to negotiate peer-based fail-over or community aggregation of resources (similar to microgrids). In this case, the community bandwidth resources (of the 500 homes) will still limit failover capacity. Without additional network infrastructure support, the 500 home community can only support 12 AR users that are not processing data locally, even if the failover peer is in the same local community. With more bandwidth, additional redundant edge computers would become feasible, but this requires last-mile network support that is both expensive and has historically been slow to deploy.

Failing to meet real-time deadlines. The promise of low latency from edge computing attracts application deployment with soft real-time requirements. AR applications, for example, need to remain below 16ms end-to-end latency to maintain seamless 60 frame-per-second user interactions. Such latency is easily achievable on local devices, but having all client devices have the requisite computing capability is infeasible. Edge computing provides a cost saving aspect, especially if the edge device is idle most of the time (such as a “computer” in the cash register of a store). This leads to real-time applications operating on edge computers instead of on client devices (see the accompanying table).

Inside an edge computing device, a finite amount of CPU, RAM, GPU/APU, and networking support exist.

Networking Challenges

IoT and edge computing are positioning themselves to upend traditional consumer models for network usage. Since the inception of a consumer-level Internet, content has existed in a central location, and consumers have downloaded that content, be it a website, audio file, or movie. This near-constant trend has led to extensive asymmetric network deployments where finite bandwidth resources are partitioned to favor download over upload. For example, the 2016 FCC definition of broadband^a is 25Mbps download and only 3Mbps upload, an 8:1 ratio. Such ratios make sense for traditional Internet applications—users do not produce much content. Slowly this has begun to change with the advent of live streaming video applications such as Periscope and YouTube Live, but even in these deployments, the viewer-to-stream ratio is still quite high. IoT will upend this ratio, so that one Internet user (a home, for example) may produce 5+ video streams (as security cameras for example) for only one real-time viewer. Given the current Internet needs, it will be impractical to scale down the 25Mbps download to make room for the 25Mbps upload, given finite radio spectrum and despite the frantic activity to release white space broadcast TV spectrum for communications needs. Instead, more expensive technologies such as Google Fiber’s active networks will have to replace aging infrastructure. Since such deployments are slow and expensive and unlikely to gain universal penetration, and IoT devices are ready today, edge computing is a viable solution to the constrained network problem.

a <http://bit.ly/2ma2Ved>

Each real-time application needs some slice of these resources to perform its task within the prescribed deadline. Real-time scheduling in a constrained environment—a problem often solved by earliest deadline first—can be complicated by the intermix of delay tolerance levels from different applications, unpredictable user interactions, unpredictable network behavior between the edge device and the client device, as well as drift in the clocks of multiple client devices being served by a common edge device.⁸ Devices may wait to operate until capabilities can be secured from the edge computing network. As an example, consider a video stream from an AR device being analyzed at an edge device and a thermostat now needs analysis of the video stream to determine how many people are in a room and adjust its setting accordingly. If the edge device can only support the analysis of a single video stream due to the demanding nature of such processing, it will have to make a scheduling decision to prioritize one of the two streams. If the edge computing paradigm becomes popular, this almost guarantees contention for the

resources at the edge devices, and the assumption of instant availability of resources that many applications make today, may cause failures of timeliness guarantees for many real-time services. One possible solution approach here is to use the cloud as a failure backup, for delay-tolerant applications. Also, where the application is stateful, and in view of the impermanence of some edge devices, the state may be stored on the client, on the cloud, or on a combination of the two.

Authentication and physical security. Edge computing must address security challenges, especially considering that client devices may be embedded in private physical spaces. In a nod to the anticipated large scale of these systems, the security mechanisms themselves must be scalable and decentralized. We expect the importance of the scalability concern will vary with the scale of the edge device deployment, from where it is local to a neighborhood (less of a concern) to where it is citywide (more of a concern). We expect that economic imperatives will mean that most edge devices will be cheap. This will mean that any security mechanism that re-

Edge computer costs.

	Raspberry Pi	Router	Xeon E3-1220L	Specialized
Video Feeds	1	0	1	1–4
GFlops	1–2	0.5–1	50	100–200
Cost	\$50	\$200	\$300–\$500	\$300–\$500

quires expensive hardware or has large memory footprint will be infeasible except for a small subset of edge devices. This opens up the design space of security mechanisms in a heterogeneous environment with a large number of constrained devices and a few (security) resource-rich devices.

Scalable authentication. Since client devices are placed close to information sources, they are necessarily distributed such that physical access to these devices cannot be protected. An attacker can do invasive probing and install malicious software on these devices. As a result, any cryptographic keys stored on the device are subject to tampering and eavesdropping. These can be made more difficult by hardening methods, but they cannot be eliminated altogether. Consequently, the authentication and trustworthiness of client devices must be validated through existing low-cost hardware security techniques such as code signatures. These techniques mostly rely on some form of public key infrastructure (PKI) which has a somewhat high computational cost, but most importantly, a high management cost.³ The PKI systems may become cumbersome for the low-cost, high-volume OEMs of either client or edge devices to properly implement and manufacturers may opt out of secure system designs in favor of ad-hoc or proprietary mechanisms.

In this model, each device requires a managed key-based authentication system, whereby devices must be marked, signed, and managed after creation. End users must be able to easily identify a device by its public information and verify through the OEM that the device is secure (for example, has an untampered software stack) and properly authenticated before sharing information with the edge infrastructure. PKI and the SSL system used today for secure banking and other services can scale out to the IoT level, but the level of interaction changes. In these traditional systems, a set of root public keys are distributed by operating systems to the end devices on a regular basis. The end device must verify any secure host's published key against this set of root public keys. In this system, the scale of secure hosts is on the order of the number of public-facing websites,

software development companies, and so on. In an IoT scenario, the number of secure host certificates will scale to the billions, placing immense pressure on device manufacturers to manage the process of issuing, storing, and securing certificates and client and edge device keys. Key management itself is often a weak point in both scalability and security,¹⁶ such as keeping root keys safe on the devices or generating keys with enough entropy.

In order to alleviate the concerns raised by public-key systems, biometric authentication can be introduced in a home environment. A central device within a home can be authenticated biometrically and then the authentication can be propagated to all connected devices. As an extra layer of security, different devices can be mapped to different expiration times depending on the functionality of the device. For critical devices such as a locker or heart monitor, the expiration times can be very low and should be authenticated every time just before their use. Multiple expiration times combined with delegation of authentication from the central control in a house leads to scalable secure methods of information transactions.

Decentralized security. The fact that the edge network may be disconnected, or be in degraded connectivity, means that security mechanisms in the edge devices should be autonomous and be capable of receiving on-demand updates, again in a secure manner.¹ Prior work on networkwide, or multi-node, code updates¹³ is relevant to us. However, there is the salient aspect of heterogeneity of the edge devices, which will have to be handled here. Because of high computing power involved in PKI systems, a low-end edge device will need to delegate its computational tasks to a more powerful device nearby. For a time-critical operation, the edge device may not be able to establish connection and verify some operation with a central server connected to the Internet. We can take some inspiration from schemes for self-organized public key management for mobile ad hoc networks,⁴ which also deal with disconnected and decentralized operation, but we will need closer attention to the latency involved. It may be possible to use physical unclonable functions (PUF) to

verify authenticity of the edge devices, without needing to contact a central data source. Hence, in a geo-distributed and mobile environment, with issues ranging from intermittent connectivity with centralized infrastructure to complete connection outages, it should be possible to make security decisions autonomously, perhaps sub-optimal from an efficiency or functionality standpoint but still meeting the privacy or security guarantees. Thus, it is important to design the equivalent of fail-safe modes of failure, analogous to what exists in safety critical systems.

Along with providing authentication and infrastructure for security in a highly scalable edge ecosystem, we also need to develop decentralized security mechanisms tailored for the edge devices. Increasing concerns in privacy and data ownership, and impaired solutions from centralized cloud infrastructures warrants distributed, peer-to-peer security mechanisms to be implemented in the edge ecosystem that also eliminates the need of centralized privacy mediators. We will have to explore the secure distributed mechanisms such as blockchain that eliminates the need of centralized infrastructure as well as protects data privacy from the owners of any centralized infrastructure. Blockchain enables secure distributed peer-to-peer transaction exchange and makes these edge devices autonomously secure. In a home, a private blockchain environment can be created consisting of digital updates and secure data sharing between smart devices. Recently, industry has started to realize the potential of blockchains in the edge ecosystem.⁹ Data from heterogeneous devices in the home can be converted into blockchain ledger format through custom API and then used securely in a home monitoring system.


Multi-tenancy of services and billing. An edge device, much like a computing node in the cloud, will need to support multiple tenants. Clouds perform this by virtualization combined with a per-user public billing system. It has been found in the virtualization literature^{10,12} that while some resources can be well partitioned (like processing cores), some others are notoriously difficult to partition (like cache capacity and memory bandwidth), which leads

to performance interference. In edge computing, since low latency is an important driver, it will be particularly important to prevent performance failures. Further, edge computers will need a similar billing system to properly manage resources in a congested system. For example, a smart thermostat from company A and a smart oven from company B may both wish to use edge computing resources. The billing question will depend critically on whether the edge device is within the premises and under a single ownership, or outside the premises and under shared ownership or providing service to multiple unrelated users. In the first case, the billing question is distinctly easier. Regarding multi-tenancy, deciding which client application gets what portion of the resource at the edge device will require input about ownership and applications' timing requirements.


Consumers may be unable to understand the nuances of contention and therefore an automated solution is necessary to solve this multi-objective optimization problem. Popular virtualization technologies such as virtual machine, or even containers, may be too heavyweight for edge devices. These need a relatively significant amount of hardware resources to execute, for example, VMWare's ESX hypervisor needs a recommended 8GB of RAM, while a Docker container with NAT enabled doubles the latency of a UDP stream⁵ (see Figure 1). Thus, the challenge will be to find a lighter-weight solution for multi-tenancy, possibly at the expense of reducing the isolation among the different applications and limiting the total number of applications.

Standardization. In our daily life, we are increasingly seeing a proliferation of "smart devices." However, these are being built essentially without regard to standardization and thus interoperability. We expect that for edge computing to flourish, this trend will need to be arrested and instead, standardization put in place. Thus, new standards or new mediation layers should be designed to coordinate those devices to provide useful functionality, such that an edge device can seamlessly communicate with, and possibly control, multiple end user devices, fail-over from one device to another is possible.

It will be economically viable to re-



An edge device, much like a computing node in the cloud, will need to support multiple tenants.



alize edge computing if relationships and risks among all parties can be clearly delineated. In the area of cloud computing, many standards have been proposed, such as the NIST Cloud Computing Reference Architecture (CCRA) 2.0, the ISO/IEC standard under the group "Cloud Computing and Distributed Platforms," and the Cloud Standards Customer Council (CSCC) standard under "Data Residency Challenges." These are slowly beginning to gain momentum, with some early signs of convergence toward the NIST standards. Likewise, interoperability in the edge computing landscape will require standardization for various aspects—visualization, data management, and programming APIs.

However, existing standards from cloud computing cannot be copied over directly to edge computing due to many factors. One primary factor is the sensitivity of information being available at edge devices, such as more personal data being collected at finer time granularity and this increases the risks of exposure of sensitive data. Another distinction is the service delivery requirement. In cloud computing, usually the computing task is more heavy duty with sizable amount of data transfer whereas in edge computing, there is more frequent and lighter communication. Therefore, the standard for edge computing must address the issues of service delivery latency and bandwidth differently. Thus, it appears to us a great deal of effort needs to be expended in standardization—with partnership among industry and non-profits—to create a flourishing edge computing marketplace.

Application Challenges

Here, we illustrate edge computing's resiliency challenges inside of two example applications, the first is a tongue-in-cheek smart toaster and the second a smart home camera. The challenges encompass practical aspects of integrating data-intensive applications in an edge environment with low network bandwidth to the Internet.

Smart toaster. We motivate our discussions with a hypothetical example: a smart toaster (Figure 2). Our smart toaster takes a simple, ordinary function—toasting bread—to an absurd level of IoT edge integration to high-

Natural Failure Case

The mission of a Tsunami Warning Centre (TWC) is to provide early warnings on potentially destructive tsunamis. A TWC uses local and global seismographic networks transmitting seismograms in real time to continuously monitor seismic activity in order to locate and size potential tsunamis. There is a great need for speed here because tsunami waves can travel at the speed of a commercial jet plane, over 500mph, in deep ocean waters. The response time of a TWC when it can rely on local data and processing is 2–5 minutes and this is increased by a factor of 10 when it needs to rely on distant data and processing.⁷ Precisely due to earthquake or pre-earthquake activity, infrastructure may be disrupted, such as, power lines and communication lines, as has happened in multiple past tsunamis such as the 2011 Tohoku tsunami and Fukushima nuclear meltdown. Such disruption to infrastructure can impact the response time when there is reliance on distant data and processing. A resilient edge computing infrastructure can be properly harnessed and coordinated to collect data, do local processing, and aid in localized disaster mitigation efforts. Resilience implies the factors that disrupted the current infrastructure should not affect the edge computing infrastructure. This is possible through localized communication infrastructure and the inherent redundancy in the edge devices. This is a particularly relevant use case because it satisfies the two key characteristics of edge computing, namely, geo-specific data and processing and requirement for low latency.

light the future research issues. The device is a WiFi-connected toaster with video feeds of the toasting area, a motorized bread lift, and electronic heater controls. The toasting can be scheduled for a future time period and the appropriate number of breads can be retrieved from its storage area. All types of breads, from bagels to Texas toast, are accepted, and the device classifies the bread based on its weight and video-derived characteristics. The image processing utilizes a deep learn-

ing approach (for example, CNN) that learns from the raw video feeds. There are decisions made prior to toasting (what breads to schedule at what time based on prior user preferences) and some during toasting.

A typical workflow is that the scheduler schedules the right number of breads for toasting. When the time is reached, the breads are inserted into the toaster. The video feed is processed by the CNN and a classification is created. The toasting

begins, and live video is streamed to the CNN to determine appropriate toast level. Once the proper level is reached, as inferred by the video processing software, the toast is ejected from the oven area and a notification is sent to the user. In the theme of our earlier discussions, the video bandwidth is 24Mbps from four cameras at 6Mbps each (1080p video). Delay tolerance in this domain is on the order of a second, however many edge computing applications will have more demanding delay requirements, such as AR. Such differing delay tolerance requirements should be handled by the edge device, which will typically support multiple client applications.

This workflow parallels many time critical control processes that use complex data-driven algorithms to control real-time systems. Other examples include GE’s Digital Twin where engine controls are supplemented with computationally heavy surrogate models to optimize operating cost and tsunami warning systems (see the sidebar “Natural Failure Case”) where distributed seismograph processing reduces the time to warnings.

Availability. Latency-sensitive applications need graceful degradation support for operation under edge failure. For example, a smart toaster may

Figure 2. Our smart toaster.

A hypothetical example that carries the ordinary function of toasting bread to an absurd level of IoT-edge-integration to highlight the future research issues. It can toast various kinds of bread according to user preferences, using video processing to determine when the appropriate level of toasting has been reached. It can also order the right amounts of the right kind of bread. It relies on edge devices and optionally cloud processing to achieve its functionality.



utilize CNN-based processing for raw video feeds to determine proper toast level. Under ideal circumstances, this video feed is sent to and processed at the local edge computing node (for example, Google Home). Some delay is tolerated, but a baking process cannot be interrupted and still achieve consistency. If the CNN processor at the edge becomes unavailable due to a fault, then the local device must operate without such decision support. Several solutions may exist:

Preprocessed decisions. An acceptable toast time may be generated based on prior toasting events. Whenever the process starts, the toaster is programmed with a fail-safe decision about when to stop. This approach represents classic average-case static control systems. For example, most toast needs five minutes of oven time, so the toasting system is set up for that default.

Supervisory control. Running a full CNN may not be practical at the edge device when it is under contention from other client applications, but alternative low-power algorithms can exist simultaneously with the CNN logic. If the processing time exceeds the delay tolerance level, or is predicted to, then a switch can be made to the alternative, simpler processing block. Mean pixel color, for example, can serve as a

lower-intensity processing alternative to determine adequate toastiness. This represents more robust but still simple control algorithms that can reliably provide improvements even under contention situations.

Degraded operational modes. Trusty toaster controls have existed for quite some time, and often users can accept a temporary lack of feature availability as long as it is not too disruptive. Manual toasting controls as an override can provide degraded mission support, but the user loses the ability to schedule toasting a priori. When high-bandwidth connection to the cloud is available, it may be possible to determine the availability of the bread for ordering from nearby retailers. When such connection is unavailable, a degraded mode of operation could be to simply inform the user the supply in the toaster storage has run out. The general principle is that the edge computing applications must be designed with multiple degraded modes of operation in mind.

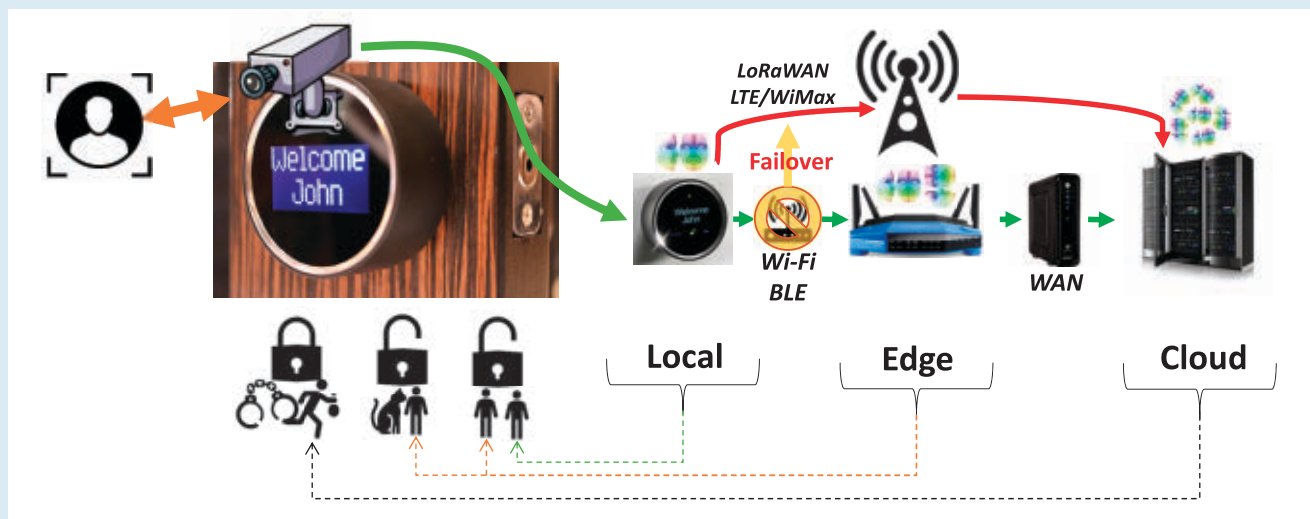
Smart door lock. Our smart door lock, shown in Figure 3, is an example edge system where speed is of essence. Consider that the door is equipped with a camera, which is meant to take pictures of anyone ap-

proaching the door and then depending on the result of an authentication, granting access or not. People that approach the system must be classified very quickly—there is nothing more frustrating than pulling on a locked door. It is not practical, however, to maintain a comprehensive database of authorized and unauthorized faces in the local device, nor is it practical to support very large neural networks or other machine learning algorithms on the resource-constrained embedded device. Reaching out to the upstream edge and cloud servers can provide increasing levels of computation at the expense of latency. Smart edge designers will place the most used profiles in the local domain, such as office staff, and the lesser used profiles in the cloud, such as criminal databases. If the edge devices fail or the connection between the local device and the edge devices fails, perhaps due to malicious activity, then the local links can be bypassed by backup wireless networks such as LTE, LoRaWAN, or WiMax to prevent criminals from going undetected.

Resource contention. Since edge computing enters the domain of real-time control, the edge resources must be properly managed to avoid contention issues. Similar problems arise in

Figure 3. An example smart door lock takes pictures of people and objects that come near it and classifies them as trusted, untrusted, or mischievous.

Deeper layers in the edge computing infrastructure provide larger databases of objects and more comprehensive recognition algorithms. The local device contains only the most common users for low-latency operation. The edge device has several additional users such as frequent visitors and maintenance personnel, and the cloud device includes a criminal database that alarms if known criminals come near the door. In the event of a failure, the door lock can bypass the edge device and go directly to the cloud, albeit through a slower connection.



distributed smart grids—overloading the grid even temporarily can cause issues. Some accepted solutions rely on constraint-managing dispatch systems that classify different loads by their requirements. In Petersen et al.,¹⁴ for example, power loads are classified as batteries, bakeries, and buckets (BBB). A bakery is the kind of load where the process must run in one continuous stretch at constant power consumption. The bakery could be a commercial greenhouse, where plants must receive a specific amount of light each day. This light must, however, be delivered continuously to stimulate the photosynthesis of the plants. Our toaster example is this kind of load. In the edge computing scenario, each client application registers with a resource manager, and the devices can effectively reserve resources prior to execution. In the case of the toaster, this means the CNN process and bandwidth for the video feed are reserved prior to starting the toast process.

This thrust indicates there is research to be done for the appropriate level of reservation and scheduling under time constraints. We can rely on significant prior work in the area of soft real-time systems. However, two domain-specific challenges arise here. First, the delay tolerance can be specific to the context, for example, the specific user using the device. This must be programmed in, and in the longer term, learned by the scheduler for the edge resource. Second, there are several levels of resources available for making the scheduling decision—the client device, the edge device, and resources on the cloud. Each choice has interdependent effect on choices made for other client applications.

Authentication. Authentication will exist in two pieces. The first piece—credentials—will be relatively straight forward to solve. As with SSL, a collection of central authorities (such as Azure, EC2, Rackspace) will provide API’s for registering and generating signed public/private key pairs from their IoT support systems. The second piece, access control, must contain the association between users, their devices, and the edge computers. This piece is complicated by both scale and usability factors.

We propose the best solution will be a distributed management layer for device-to-edge association. Simple trusted interfaces can be established in the local domain, via physical access, and used as a gateway for additional device association. Armed with a trusted root certificate, an edge device can verify the certificates of all peer devices locally. Once trusted, the identity simply needs to be added to the access control list for the local edge system. The list can be managed by a trusted smartphone application with access to add a key, corresponding a new device, to the list. In a simple case, each client device could have a barcode, and the application can be used to identify the device and its public key simply by scanning it with the phone’s camera. The key requirement would be to simplify the user involvement.

Conclusion

Edge computing presents an exciting new computational paradigm that supports growing geographically distributed data integration and data processing for the Internet of Things and augmented reality applications. Edge devices and edge computing interactions reduce network dependence and support low latency, context-aware information processing in environments close to the client devices. However, services built around edge computing are likely to suffer from new failure modes, both hard failures (unavailability of certain resources) and soft failures (degraded availability of certain resources). Low-latency requirements combined with budget constraints will limit the fail-over options available in edge computing compared to a traditional cloud-based environment. Consequently, system developers must develop and deploy applications on the edge with an understanding of such constraints.

Additional issues edge computing will face include authentication at scale, cost amortization, and resource contention management. Further, for a thriving ecosystem, it is essential to have standardization of the device and network APIs, something that has not been seen to date. How these issues are handled will ultimately

determine the success or failure of the paradigm of edge computing. Like many technology inflection points, timely moves on the thrusts outlined in this article can significantly tilt the balance in favor of success. □

References

1. Bagchi, S., Shroff, N.B., Khalil, I.M., Panta, R.K., Krasniewski, M.D., and Krogmeier, J.V. *Protocol for secure and energy-efficient reprogramming of wireless multi-hop sensor networks*. U.S. Patent 8,107,397, 2012.
2. Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. Fog computing and its role in the Internet of things. In *Proceedings of the 1st Edition of the MCC Workshop on Mobile Cloud Computing*. ACM, 2012, 13–16.
3. Butler, R., Welch, V., Engert, D., Foster, I., Tuecke, S., Volmer, J., and Kesselman, C. A national-scale authentication infrastructure. *Computer* 33,12 (2000), 60–66.
4. Capkun, S., Buttyan, L., and Hubaux, J.P. Self-organized public-key management for mobile ad hoc networks. *IEEE Trans. Mobile Computing* 2,1 (2003), 52–64.
5. Felten, W., Ferreira, A., Rajamony, R., and Rubio, J. An updated performance comparison of virtual machines and Linux containers. In *Proceedings of the 2015 IEEE Intern. Symposium on Performance Analysis of Systems and Software*. IEEE, 2015, 1–14.
6. Hu, Y.C., Patel, M., Sabella, D., Sprecher, N., and Young, V. Mobile edge computing—A key technology towards 5G. *ETSI White Paper 11* (2015).
7. Kong, L., McCreery, C., and Yamamoto, M. *Indian Ocean Tsunami Information Center*, 2008; <http://bit.ly/2lQHyhL>.
8. Koo, J., Panta, R.K., Bagchi, S., and Montestrucque, L. A tale of two synchronizing clocks. In *Proceedings of the 7th ACM Conf. Embedded Networked Sensor Systems*. ACM, 2009, 239–252.
9. International Business Machines. *IBM Blockchain*, 2017; <https://www.ibm.com/blockchain/>
10. Maji, A.K., Mitra, S., Zhou, B., Bagchi, S., and Verma, A. Mitigating interference in cloud services by middleware reconfiguration. In *Proceedings of the 15th Intern. Middleware Conf.* ACM, 2014, 277–288.
11. Juniper Networks. *Mobile Edge Computing Use Cases & Deployment Options*, 2015; <https://juni.pr/2kEXy6k>
12. Novakovic, D.M., Vasic, N., Novakovic, S., Kostic, D., and Bianchini, R. DeepDive: Transparently identifying and managing performance interference in virtualized environments. In *Proceedings of USENIX Annual Technical Conf.*, 2013, 219–230.
13. Panta, R.K., Bagchi, S., and Midkiff, S.P. Efficient incremental code update for sensor networks. In *Proceedings of ACM Trans. Sensor Networks*, 2011, 30:1–30:32.
14. Petersen, M.K., Edlund, K., Hansen, L.H., Bendtsen, J., and Stoustrup, J. A taxonomy for modeling flexibility and a computationally efficient algorithm for dispatch in smart grids. In *Proceedings of the American Control Conf.* IEEE, 2013, 1150–1156.
15. Satyanarayanan, M. The emergence of edge computing. *Computer* 50, 1 (2017), 30–39.
16. Ars Technica. *Crypto shocker: Four of every 1,000 public keys provide no security* (2012); <http://bit.ly/2kaCZOY>
17. UNESCO and ITU. *The State of Broadband 2014: Broadband for all*, 2014; <http://bit.ly/2mdHxFb>.

Saurabh Bagchi is a professor in the School of Electrical and Computer Engineering and director of the CRISP Center at Purdue University, West Lafayette, IN, USA.

Muhammad-Bilal Siddiqui is a software engineer at Qualcomm in San Diego, CA, USA.

Paul Wood is a cyber security researcher at The Johns Hopkins University Applied Physics Laboratory in Laurel, MD, USA.

Heng Zhang is a graduate research assistant in the School of Electrical and Computer Engineering at Purdue University, West Lafayette, IN, USA.

ACM Welcomes the Colleges and Universities Participating in ACM's Academic Department Membership Program

ACM offers an Academic Department Membership option, which allows universities and colleges to provide ACM Professional Membership to their faculty at a greatly reduced collective cost.

The following institutions currently participate in ACM's Academic Department Membership program:

- Abilene Christian University
- Afrisol Technical College, Zimbabwe
- Alfred State College
- Amherst College
- Appalachian State University
- Augusta University, School of Computer and Cyber Sciences
- Ball State University
- Bellevue College
- Berea College
- Binghamton University
- Boise State University
- Bridgewater State University
- Bryant University
- California Baptist University
- Calvin College
- Clark University
- Colgate University
- Colorado School of Mines
- Columbus State University
- Cornell University
- Creighton University
- Cuyahoga Community College
- Denison University
- European University (Tbilisi, Georgia)
- Franklin University
- Gallaudet University
- Georgia Institute of Technology
- Georgia State University Perimeter College
- Governors State University
- Harding University
- Harvard University
- Harvey Mudd College
- Hochschule für Technik Stuttgart - University of Applied Sciences
- Hofstra University
- Hope College
- Howard Payne University
- Indiana University Bloomington
- Kent State University
- Klagenfurt University, Austria
- Madinah College of Technology, Saudi Arabia
- Massasoit Community College
- Messiah College
- Metropolitan State University
- Missouri State University
- Modesto Junior College
- Monash University, Australia
- Montclair State University
- Mount Holyoke College
- New Jersey Institute of Technology
- New Mexico State University
- Northeastern University
- Ohio State University
- Old Dominion University
- Pacific Lutheran University
- Pennsylvania State University
- Potomac State College of West Virginia University
- Purdue University Northwest
- Regis University
- Rhodes College
- Rochester Institute of Technology
- Rutgers University
- Saint Louis University
- San José State University
- Shippensburg University
- Simmons University
- Spelman College
- St. John's University
- Stanford University
- State University of New York at Fredonia
- State University of New York at Oswego
- Stetson University
- Trine University
- Trinity University
- Union College
- Union University
- Univ. do Porto, Faculdade de Eng. (FEUP)
- University at Albany, State University of New York
- University of Alabama
- University of Arizona
- University of California, Riverside
- University of California, San Diego
- University of Colorado Boulder
- University of Colorado Denver
- University of Connecticut
- University of Houston
- University of Illinois at Chicago
- University of Jamestown
- University of Liechtenstein
- University of Lynchburg
- University of Maribor, Slovenia
- University of Maryland, Baltimore County
- University of Memphis
- University of Namibia
- University of Nebraska at Kearney
- University of Nebraska Omaha
- University of New Mexico
- University of North Dakota
- University of Pittsburgh
- University of Puget Sound
- University of Southern California
- University of St. Thomas
- University of the Fraser Valley
- University of Victoria, BC Canada
- University of Wisconsin–Parkside
- University of Wyoming
- Virginia Commonwealth University
- Wake Forest University
- Wayne State University
- Wellesley College
- Western New England University
- William Jessup University

Through this program, each faculty member receives all the benefits of individual professional membership, including *Communications of the ACM*, member rates to attend ACM Special Interest Group conferences, member subscription rates to ACM journals, and much more.

Uncovering the mysterious ways machine learning models make decisions.

BY MENGNAN DU, NINGHAO LIU, AND XIA HU

Techniques for Interpretable Machine Learning

MACHINE LEARNING IS progressing at an astounding rate, powered by complex models such as ensemble models and deep neural networks (DNNs). These models have a wide range of real-world applications, such as movie recommendations of Netflix, neural machine translation of Google, and speech recognition of Amazon Alexa. Despite the successes, machine learning has its own limitations and drawbacks. The most significant one is the lack of transparency behind their behaviors, which leaves users with little understanding of how particular decisions are made by these models. Consider, for instance, an advanced self-driving car equipped with various machine learning algorithms does not brake or decelerate when confronting a stopped firetruck. This unexpected

» key insights

- Techniques for interpretable machine learning can be grouped into two categories—intrinsic and post-hoc interpretability—both of which can be further classified into global and local interpretability.
- Model explanation and surprising artifacts are often two sides of the same coin. Good explanation should be meaningful to users and avoid being influenced by artifacts.
- Interpretable machine learning is a multidisciplinary work that needs efforts from computer science, human-computer interaction, and social science, aiming to design real user-oriented and human-friendly explanations.

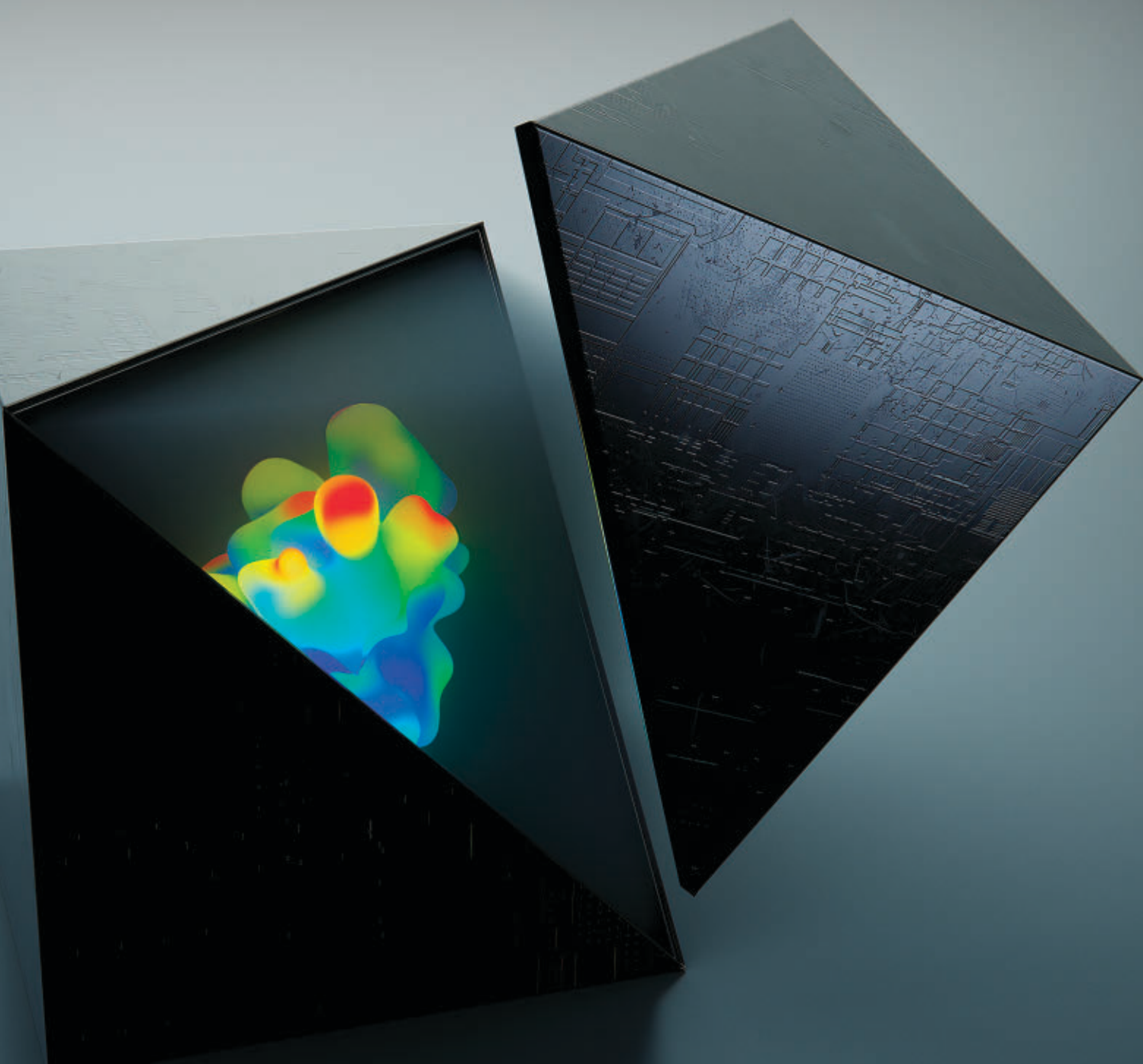


ILLUSTRATION BY PETER GROWTHER ASSOCIATES

behavior may frustrate and confuse users, making them wonder why. Even worse, the wrong decisions could cause severe consequences if the car is driving at highway speeds and might ultimately crash into the firetruck. The concerns about the black-box nature of complex models have hampered their further applications in our society, especially in those critical decision-making domains like self-driving cars.

Interpretable machine learning would be an effective tool to mitigate these problems. It gives machine learning models the ability to explain or to present their behaviors in understandable terms to humans,¹⁰ which is called interpretability or explainability and

we use them interchangeably in this article. Interpretability would be an indispensable part for machine learning models in order to better serve human beings and bring benefits to society. For end users, explanation will increase their trust and encourage them to adopt machine learning systems. From the perspective of machine learning system developers and researchers, the provided explanation can help them better understand the problem, the data and why a model might fail, and eventually increase the system safety. Thus, there is a growing interest among the academic and industrial community in interpreting machine learning models and gaining insights into their

working mechanisms.

Interpretable machine learning techniques can generally be grouped into two categories: intrinsic interpretability and post-hoc interpretability, depending on the time when the interpretability is obtained.²³ Intrinsic interpretability is achieved by constructing self-explanatory models which incorporate interpretability directly to their structures. The family of this category includes decision tree, rule-based model, linear model, attention model, and so on. In contrast, the post-hoc one requires creating a second model to provide explanations for an existing model. The main difference between these two groups lies in the trade-off between

model accuracy and explanation fidelity. Inherently interpretable models could provide accurate and undistorted explanation but may sacrifice prediction performance to some extent. The post-hoc ones are limited in their approximate nature while keeping the underlying model accuracy intact.

Based on categorization noted here, we further differentiate two types of interpretability: global interpretability and local interpretability. Global interpretability means users can understand how the model works globally by inspecting the structures and parameters of a complex model, while local interpretability examines an individual prediction of a model locally, trying to figure out why the model makes the

decision it makes. Using the DNN in Figure 1 as an example, global interpretability is achieved by understanding the representations captured by the neurons at an intermediate layer, while local interpretability is obtained by identifying the contributions of each feature in a specific input to the prediction made by DNN. These two types bring different benefits. Global interpretability could illuminate the inner working mechanisms of machine learning models and thus can increase their transparency. Local interpretability will help uncover the causal relations between a specific input and its corresponding model prediction. Those two help users trust a model and trust a prediction, respectively.

In this article, we first summarize current progress of three lines of research for interpretable machine learning: designing inherently interpretable models (including globally and locally), post-hoc global explanation, and post-hoc local explanation. We proceed by introducing applications and challenges of current techniques. Finally, we present limitations of current explanations and propose directions toward more human-friendly explanations.

Inherently Interpretable Model

Intrinsic interpretability can be achieved by designing self-explanatory models that incorporate interpretability directly into the model structures. These constructed interpretable models either are globally interpretable or could provide explanations when they make individual predictions.

Globally interpretable models can be constructed in two ways: directly trained from data as usual but with interpretability constraints and being extracted from a complex and opaque model.

Adding interpretability constraints. The interpretability of a model could be promoted by incorporating interpretability constraints. Some representative examples include enforcing sparsity terms or imposing semantic monotonicity constraints in classification models.¹⁴ Here, sparsity means a model is encouraged to use relatively fewer features for prediction, while monotonicity enables the features to have monotonic relations with the prediction. Similarly, decision trees are pruned by replacing subtrees with leaves to encourage long and deep trees rather than wide and more balanced trees.²⁹ These constraints make a model simpler and could increase the model’s comprehensibility by users.

Besides, more semantically meaningful constraints could be added to a model to further improve interpretability. For instance, interpretable convolutional neural networks (CNN) add a regularization loss to higher convolutional layers of CNN to learn disentangled representations, resulting in filters that could detect semantically meaningful natural objects.³⁹ Another work combines novel neural units, called capsules, to construct a capsule

Figure 1. An illustration of three lines of interpretable machine learning techniques, taking DNN as an example.

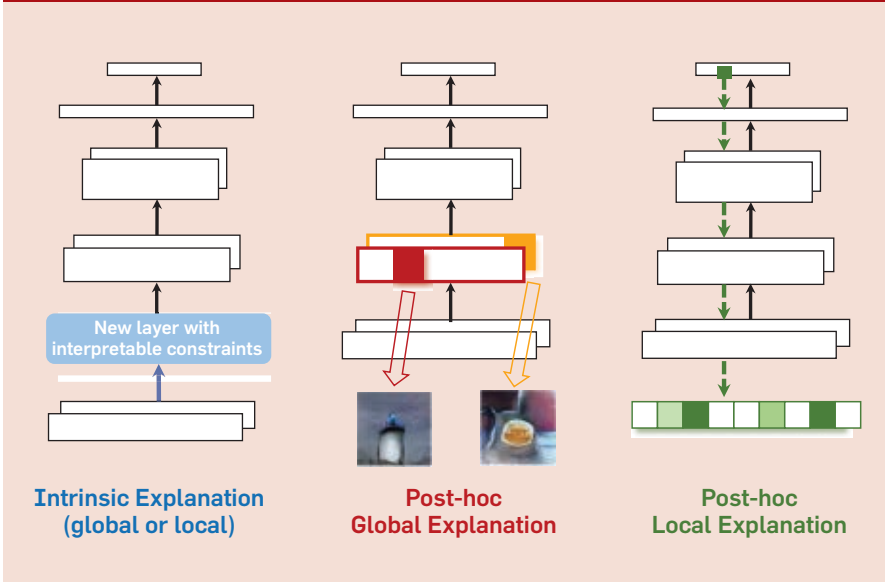
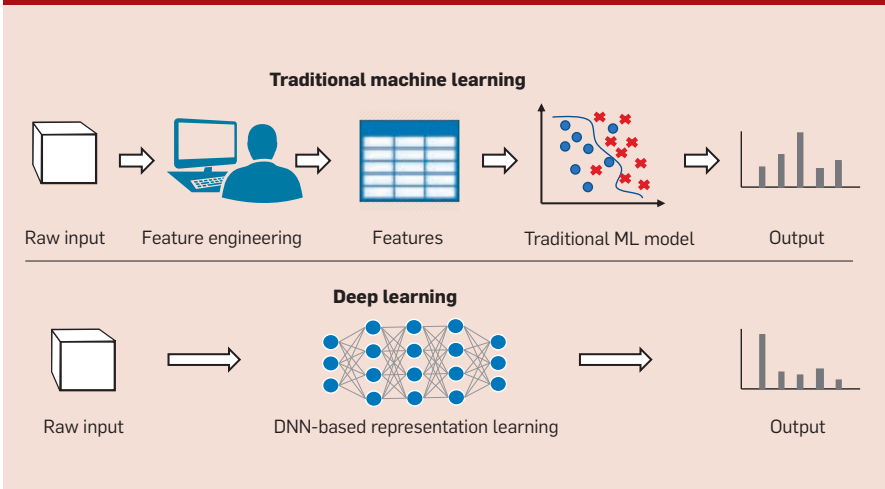


Figure 2. A traditional machine learning pipeline using feature engineering, and a deep learning pipeline using DNN-based representation learning.



network.³² The activation vectors of an active capsule can represent various semantic-aware concepts like position and pose of a particular object. This nice property makes capsule network more comprehensible for humans.

However, there are often trade-offs between prediction accuracy and interpretability when constraints are directly incorporated into models. The more interpretable models may result in reduced prediction accuracy comparing the less interpretable ones.

Interpretable model extraction. An alternative is to apply interpretable model extraction, also referred as mimic learning,³⁶ which may not have to sacrifice the model performance too much. The motivation behind mimic learning is to approximate a complex model using an easily interpretable model such as a decision tree, rule-based model, or linear model. As long as the approximation is sufficiently close, the statistical properties of the complex model will be reflected in the interpretable model. Eventually, we obtain a model with comparable prediction performance, and the behavior of which is much easier to understand. For instance, the tree ensemble model is transformed into a single decision tree.³⁶ Moreover, a DNN is utilized to train a decision tree that mimics the input-output function captured by the neural network so the knowledge encoded in DNN is transferred to the decision tree.⁵ To avoid the overfitting of the decision tree, active learning is applied for training. These techniques convert the original model to a decision tree with better interpretability and maintain comparable predictive performance at the same time.

Locally interpretable models are usually achieved by designing more justified model architectures that could explain why a specific decision is made. Different from the globally interpretable models that offer a certain extent of transparency about what is going on inside a model, locally interpretable models provide users understandable rationale for a specific prediction.

A representative scheme is employing attention mechanism,^{4,38} which is widely utilized to explain predictions made by sequential models, for example, recurrent neural networks (RNNs). Attention mechanism is advantageous

in that it gives users the ability to interpret which parts of the input are attended by the model through visualizing the attention weight matrix for individual predictions. Attention mechanism has been used to solve the problem of generating image caption.³⁸ In this case, a CNN is adopted to encode an input image to a vector, and an RNN with attention mechanisms is utilized to generate descriptions. When generating each word, the model changes its attention to reflect the relevant parts of the image. The final visualization of the attention weights could tell human what the model is looking at when generating a word. Similarly, attention mechanism has been incorporated in machine translation.⁴ At decoding stage, the neural attention module added to neural machine translation (NMT) model assigns different weights to the hidden states of the decoder, which allows the decoder to selectively focus on different parts of the input sentence at each step of the output generation. Through visualizing the attention scores, users could understand how words in one language depend on words in another language for correct translation.

Post-Hoc Global Explanation

Machine learning models automatically learn useful patterns from a huge amount of training data and retain the learned knowledge into model structures and parameters.

Post-hoc global explanation aims to provide a global understanding about what knowledge has been acquired by these pretrained models and illuminate the parameters or learned representations in an intuitive manner to humans. We classify existing models into two categories: traditional machine learning and deep learning pipelines (see Figure 2), since we are capable of extracting some similar explanation paradigms from each category. Here, we introduce how to provide explanation for these two types of pipelines.

Traditional machine learning explanation. Traditional machine learning pipelines mostly rely on feature engineering, which transforms raw data into features that better represent the predictive task, as shown in Figure 2. The features are generally interpretable and the role of machine learning is to map the representation to output. We

consider a simple yet effective explanation measure that is applicable to most of the models belonging to traditional pipeline, called feature importance, which indicates statistical contribution of each feature to the underlying model when making decisions.

Model-agnostic explanation. The model-agnostic feature importance is broadly applicable to various machine learning models. It treats a model as a black-box and does not inspect internal model parameters.

A representative approach is “permutation feature importance.”¹ The key idea is the importance of a specific feature to the overall performance of a model can be determined by calculating how the model prediction accuracy deviates after permuting the values of that feature. More specifically, given a pretrained model with n features and a test set, the average prediction score of the model on the test set is p , which is also the baseline accuracy. We shuffle the values of a feature on the test set and compute the average prediction score of the model on the modified dataset.

This process is iteratively performed for each feature and eventually n prediction scores are obtained for n features respectively. We then rank the importance of the n features according to the reductions of their score comparing to baseline accuracy p . There are several advantages for this approach. First, we do not need to normalize the values of the handcrafted features. Second, it can be generalized to nearly any machine learning models with handcrafted features as input. Third, this strategy has been proved to be robust and efficient in terms of implementation.

Model-specific explanation. There also exists explanation methods specifically designed for different models. Model-specific methods usually derive explanations by examining internal model structures and parameters. Here, we introduce how to provide feature importance for two families of machine learning models.

Generalized linear models (GLM) is constituted of a series of models that are linear combination of input features and model parameters followed by feeding to some transformation function (often nonlinear).²¹ Examples of GLM include linear regression and logistic regression. The weights of a

GLM directly reflect feature importance, so users can understand how the model works by checking their weights and visualizing them. However, the weights may not be reliable when different features are not appropriately normalized and vary in their scale of measurement. Moreover, the interpretability of an explanation will decrease when the feature dimensions become too large, which may be beyond the comprehension ability of humans.

Tree-based ensemble models, such as gradient boosting machines, random forests, and XGBoost,⁷ are typically inscrutable to humans. There are several ways to measure the contribution of each feature. The first approach is to calculate the accuracy gain when a feature is used in tree branches. The rationale behind is that without adding a new split to a branch for a feature, there may be some misclassified elements, while after adding the new branch, there are two branches and each one is more accurate. The second approach measures the feature coverage, that is, calculating the relative quantity of observations related to a feature. The third approach is to count the number of times that a feature is used to split the data.

DNN representation explanation. DNNs, in contrast to traditional models, not only discover the mapping from representation to output, but also learn representations from raw data,¹⁵ as illustrated in Figure 2. The learned deep representations are usually not human interpretable,¹⁹ hence the explanation for DNNs mainly focuses on understanding the representations captured by the neurons at intermediate layers of DNNs. Here, we introduce explanation methods for two major categories of DNN, including CNN and RNN.

Explanation of CNN representation. There has been a growing interest to understand the inscrutable representations at different layers of CNN. Among different strategies to understand CNN representations, the most effective and widely utilized one is through finding the preferred inputs for neurons at a specific layer. This is generally formulated in the activation maximization (AM) framework,³³ which can be formulated as:

$$x^* = \underset{x}{\operatorname{argmax}} f_l(x) - \mathcal{R}(x), \quad (1)$$

where $f_l(x)$ is the activation value of a neuron at layer l for input x , and $\mathcal{R}(x)$ is a regularizer. Starting from random initialization, we optimize an image to maximally activate a neuron. Through iterative optimization, the derivatives of the neuron activation value with respect to the image is utilized to tweak the image. Eventually, the visualization of the generated image could tell what individual neuron is looking for in its receptive field. We can in fact do this for arbitrary neurons, ranging from neurons at the first layer all the way to the output neurons at the last layer, to understand what is encoded as representations at different layers.

While the framework is simple, getting it to work faces some challenges, among which the most significant one is the surprising artifact. The optimization process may produce unrealistic images containing noise and high-frequency patterns. Due to the large searching space for images, if without proper regularization, it is possible to produce images that satisfy the optimization objective to activate the neuron but are still unrecognizable. To tackle this problem, the optimization should be constrained using natural image priors so as to produce synthetic images that resemble natural images. Some researchers heuristically propose handcrafted priors, including total variation norm, α -norm, and Gaussian blur. In addition, the optimization could be regularized using stronger natural image priors produced by a generative model, such as GAN or VAE, which maps codes in the latent space to the image spaces.²⁵ Instead of directly optimizing the image, these methods optimize the latent space codes to find an image that can activate a given neuron. Experimental results have shown the priors produced by generative models lead to significant improvements in visualization.

The visualization results provide several interesting observations about CNN representations. First, the network learns representations at several levels of abstraction, transiting from general to task-specific from the first layer to the last layer. For example, take the CNN trained with the ImageNet dataset. Lower-layer neurons detect small and simple patterns, such as object corners and textures. Mid-layer

neurons detect object parts, such as faces and legs. Higher-layer neurons respond to whole objects or even scenes. Interestingly, the visualization of the last-layer neurons illustrates that CNN exhibits a remarkable property to capture global structure, local details, and contexts of an object. Second, a neuron could respond to different images that are related to a semantic concept, revealing the multifaceted nature of neurons.²⁷ For instance, a face detection neuron can fire in response to both human faces and animal faces. Note that this phenomenon is not confined to high-layer neurons, as all layers of neurons are multifaceted. The neurons at higher layers are more multifaceted than the ones at lower layers, indicating that higher-layer neurons become more invariant to large changes within a class of inputs, such as colors and poses. Third, CNN learns distributed code for objects.⁴⁰ Objects can be described using part-based representations and these parts can be shared across different categories.

Explanation of RNN representation. Following numerous efforts to interpret CNN, uncovering the abstract knowledge encoded by RNN representations (including GRUs and LSTMs) has also attracted increasing interest in recent years. Language modeling, which targets to predict the next token given its previous tokens, is usually utilized to analyze the representations learned by RNN. The studies indicate that RNN indeed learns useful representations.^{17,18,28}

First, some work examines the representations of the last hidden layer of RNN and study the function of different units at that layer, by analyzing the real input tokens that maximally activate a unit. The studies demonstrate that some units of RNN representations are able to capture complex language characteristics, for example, syntax, semantics, and long-term dependencies. For instance, a study analyzes the interpretability of RNN activation patterns using character-level language modeling.¹⁸ This work finds that although most of the neural units are difficult to find particular meanings, there indeed exist certain dimensions in RNN hidden representations that are able to focus on specific language structures such as quotation


marks, brackets, and line lengths in a text. In another work, a word-level language model is utilized to analyze the linguistic features encoded by individual hidden units of RNN.¹⁷ The visualizations illustrate that some units are mostly activated by certain semantic category, while some others could capture a particular syntactic class or dependency function. More interestingly, some hidden units could carry the activation values over to subsequent time steps, which explains why RNN can learn long-term dependencies and complex linguistic features.

Second, the research finds that RNN is able to learn hierarchical representations by inspecting representations at different hidden layers.²⁸ This observation indicates that RNN representations bear some resemblance to their CNN counterpart. For instance, a bidirectional language model is constructed using a multi-layer LSTM.²⁸ The analysis of representations at different layers of this model shows that the lower-layer representation captures context-independent syntactic information. In contrast, higher-layer LSTM representations encode context-dependent semantic information. The deep contextualized representations can disambiguate the meanings of words by utilizing their context, and thus could be employed to perform tasks which require context-aware understanding of words.


Post-Hoc Local Explanation

After understanding the model globally, we zoom in to the local behavior of the model and provide local explanations for individual predictions. Local explanations target to identify the contributions of each feature in the input toward a specific model prediction. As local methods usually attribute a model's decision to its input features, they are also called attribution methods. Here, we first introduce model-agnostic attribution methods and then discuss attribution methods specific to DNN-based predictions.

Model-agnostic explanation. Model-agnostic methods allow explaining predictions of arbitrary machine learning models independent of the implementation. They provide a way to explain predictions by treating the models as black boxes, where explana-



Explanations could help examine whether a machine learning model has employed the true evidences instead of biases that widely exist among training data.



tions could be generated even without access to the internal model parameters. They bring some risks at the same time, since we cannot guarantee the explanation faithfully reflects the decision-making process of a model.

Local approximation-based explanation is based on the assumption the machine learning predictions around the neighborhood of a given input can be approximated by an interpretable white-box model. The interpretable model does not have to work well globally, but it must approximate the black-box model well in a small neighborhood near the original input. Then the contribution score for each feature can be obtained by examining the parameters of the white-box model.

Some studies assume the prediction around the neighborhood of an instance could be formulated as the linearly weighted combination of its input features.³⁰ Attribution methods based on this principle first sample the feature space in the neighborhood of the instance to constitute an additional training set. A sparse linear model, such as Lasso, is then trained using the generated samples and labels. This approximation model works the same as a black-box model locally but is much easier to inspect. Finally, the prediction of the original model can be explained by examining the weights of this sparse linear model instead.

Sometimes, even the local behavior of a model may be extremely non-linear, linear explanations could lead to poor performance. Models which could characterize non-linear relationship are thus utilized as the local approximation. For instance, a local approximation-based explanation framework can be constructed using if-then rules.³¹ Experiments on a series of tasks show that this framework is effective at capturing nonlinear behaviors. More importantly, the produced rules are not confined merely to the instance being explained and often generalize to other instances.

Perturbation-based explanation. This line of work follows the philosophy that the contribution of a feature can be determined by measuring how prediction score changes when the feature is altered. It tries to answer the question: Which parts of the input, if they were not seen by the model, would most change

its prediction? Thus, the results may be called counterfactual explanations. The perturbation is performed across features sequentially to determine their contributions and can be implemented in two ways: omission and occlusion. For omission, a feature is directly removed from the input, but this might be impractical since few models allow setting features as unknown. As for occlusion, the feature is replaced with a reference value, such as zero for word embeddings or specific gray value for image pixels. Nevertheless, occlusion raises a new concern that new evidence may be introduced and that can be used by the model as a side effect.⁸ For instance, if we occlude part of an image using a green color and then we may provide undesirable evidence for the grass class. Thus, we should be particularly cautious when selecting reference values to avoid introducing extra pieces of evidence.

Model-specific explanation. There are also explanation approaches exclusively designed for a specific type of

model. Here, we introduce DNN-specific methods that treat the networks as white boxes and explicitly utilize the interior structure to derive explanations. We divide them into three major categories: back-propagation based methods in a top-down manner; perturbation-based methods in a bottom-up manner; and investigation of deep representations in intermediate layers.

Back-propagation-based methods calculate the gradient, or its variants, of a particular output with respect to the input using back-propagation to derive the contribution of features. In the simplest case, we can back-propagate the gradient.³³ The underlying hypothesis is that larger gradient magnitude represents a more substantial relevance of a feature to a prediction. Other approaches back-propagate different forms of signals to the input, such as discarding negative gradient values at the back-propagation process,³⁴ or back-propagating the relevance of the final prediction score to the input layer.³ These methods

are integrated into a unified framework where all methods are reformulated as a modified gradient function.² This unification enables comprehensive comparison between different methods and facilitates effective implementation under modern deep-learning libraries, such as TensorFlow and PyTorch. Back-propagation-based methods are efficient in terms of implementation, as they usually need a few forward and backward calculations. On the other hand, they are limited in their heuristic nature and may generate explanations of unsatisfactory quality, which are noisy and highlight some irrelevant features, as shown in Figure 3b.

Mask perturbation. The previously mentioned model-agnostic perturbation could be computationally very expensive when handling an instance with high dimensions, since they need to sequentially perturb the input. In contrast, DNN-specific perturbation could be implemented efficiently through mask perturbation and gradient descent optimization. One representative work formulates the perturbation in an optimization framework to learn a perturbation mask, which explicitly preserves the contribution values of each feature.¹³ Note that this framework generally needs to impose various regularizations to the mask to produce meaningful explanation rather than surprising artifacts.¹³ Although the optimization-based framework has drastically boosted the efficiency, generating an explanation still needs hundreds of forward and backward operations. To enable more computationally efficient implementation, a DNN model can be trained to predict the attribution mask.⁸ Once the mask neural network model is obtained, it only requires a single forward pass to yield attribution scores for an input.

Investigation of deep representations. Either perturbation or back-propagation-based explanations ignore the deep representations of the DNN that might contain rich information for interpretation. To bridge the gap, some studies explicitly utilize the deep representations of the input to perform attribution.

Based on the observation that deep CNN representations capture the high-level content of input images as well as their spatial arrangement, a guided

Figure 3. Heatmap of DNN-specific local explanation. (a) original input, (b) back-propagation, (c) mask perturbation, and (d) investigation of representations.

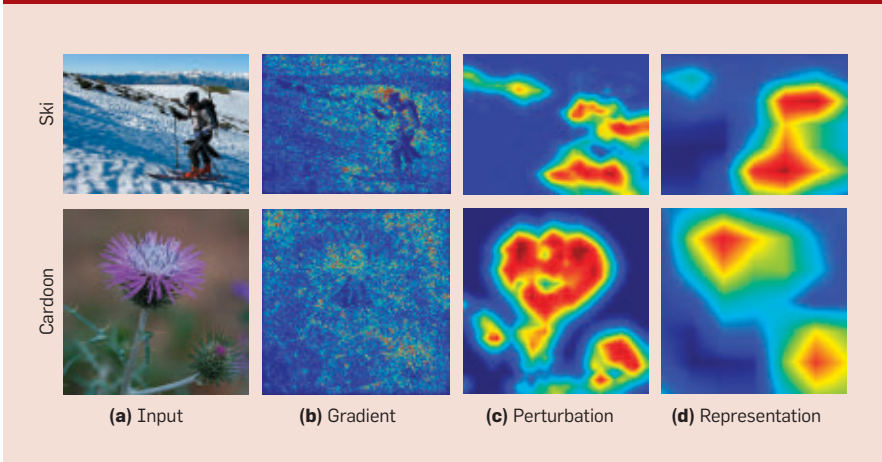
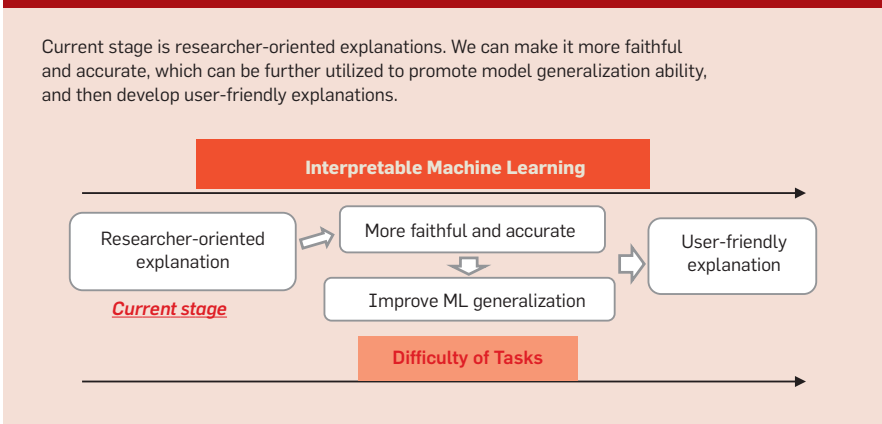


Figure 4. Progress of interpretable ML.



feature inversion framework is proposed to provide local explanations.¹¹ This framework inverts the representations at higher layers of CNN to a synthesized image, while simultaneously encodes the location information of the target object in a mask. Decomposition is another perspective to take advantage of deep DNN representations. For instance, through modeling the information-flowing process of the hidden representation vectors in RNN models, the RNN prediction is decomposed into additive contribution of each word in the input text.¹² The decomposition result could quantify the contribution of each individual word to a RNN prediction. These two explanation paradigms achieve promising results on a variety of DNN architectures, indicating the intermediate information indeed contributes significantly to the attribution. Furthermore, deep representations serve as a strong regularizer, increasing the possibility the explanations faithfully characterize the behaviors of DNN under normal operating conditions. Thus, it reduces the risks of generating surprising artifacts and leads to more meaningful explanations.

Applications

Interpretable machine learning has numerous applications. We introduce three representative ones: model validation, model debugging, and knowledge discovery.

Model validation. Explanations could help examine whether a machine learning model has employed the true evidences instead of biases that widely exist among training data. A post-hoc attribution approach, for instance, analyzes three question answering models.²⁴ The attribution heatmaps show these models often ignore important parts of the questions and rely on irrelevant words to make decisions. They further indicate the weakness of the models is caused by the inadequacies of training data. Possible solutions to fix this problem include modifying training data or introducing inductive bias when training the model. More seriously, machine learning models may rely on gender and ethnic biases to make decisions.⁹ Interpretability could be exploited to identify whether models have utilized these biases to ensure models do not violate ethical and legal requirements.

Model debugging. Explanations also can be employed to debug and analyze the misbehavior of models when models give wrong and unexpected predictions. A representative example is adversarial learning.²⁶ Recent work demonstrated that machine learning models, such as DNNs, can be guided into making erroneous predictions with high confidence, when processing accidentally or deliberately crafted inputs.^{20,26} However, these inputs are quite easy to be recognized by humans. In this case, explanation facilitates humans to identify the possible model deficiencies and analyze why these models may fail. More importantly, we may further take advantage of human knowledge to figure out possible solutions to promote the performances and reasonability of models.

Knowledge discovery. The derived explanations also allow humans to obtain new insights from machine learning model through comprehending their decision-making process. With explanation, the area experts and the end users could provide realistic feedbacks. Eventually, new science and new knowledge, which are originally hidden in the data, could be extracted. For instance, a rule-based interpretable model has been utilized to predict the mortality risk for patients with pneumonia.⁶ One of the rules from the model suggests having asthma could lower a patient's risk of dying from pneumonia. It turns out to be true since patients with asthma were given more aggressive treatments, which led to better outcomes.

Research Challenges

Despite recent progress in interpretable machine learning, there are still some urgent challenges, especially on explanation method design as well as evaluation.

Explanation method design. The first challenge is related to the method design, especially for post-hoc explanation. We argue that an explanation method should be restricted to truly reflect the model behavior under normal operation conditions. This criterion has two meanings. Firstly, the explanations should be faithful to the mechanism of the underlying machine learning model.¹² Post-hoc explanation methods propose to approximate the behavior of models. Sometimes,

the approximation is not sufficiently accurate, and the explanation may fail to precisely reflect the actual operation status of the original model. For instance, an explanation method may give an explanation that makes sense to humans, while the machine learning model works in an entirely different way. Second, even when explanations are of high fidelity to the underlying models, they may fail to represent the model behavior under normal conditions. Model explanation and surprising artifacts are often two sides of the same coin. The explanation process could generate examples that are out of distribution from the statistics in the training dataset, including nonsensical inputs and adversarial examples,¹⁶ which are beyond the capability of current machine learning models. Without careful design, both global and local explanations may trigger the artifacts of machine learning models, rather than produce meaningful explanations.

Explanation method evaluation. The second challenge involves the method evaluation. We introduce below the evaluation challenges for intrinsic explanation and post-hoc explanation.

The challenge for intrinsic explanation mainly lies in how to quantify the interpretability. There are broad sets of interpretable models designed according to distinct principles and have various forms of implementations. Take the recommender system as an example, both interpretable latent topic model and attention mechanism could provide some extent of interpretability. Nevertheless, how can we compare the interpretability between globally interpretable model and locally interpretable model? There is still no consensus on what interpretability means and how to measure the interpretability. Finale and Been propose three types of metrics: application-grounded metrics, human-grounded metrics, and functionally grounded metrics.¹⁰ These metrics are complementary to each other and bring their own pros and cons regarding the degree of validity and the cost to perform evaluations. Adopting what metrics heavily depends on the tasks so as to make more informed evaluations.

For post-hoc explanation, comparing to evaluate its interpretability, it is

equally important to assess the faithfulness of explanation to the original model, which is often omitted by existing literature. As mentioned earlier, generated explanations for a machine learning model are not always reasonable to humans. It is extremely difficult to tell whether the unexpected explanation is caused by misbehavior of the model or limitation of the explanation method. Therefore, better metrics to measure the faithfulness of explanations are needed, in order to complement existing evaluation metrics. The degree of faithfulness can determine how confident we can trust an explanation. Nevertheless, the design of appropriate faithfulness metric remains an open problem and deserves further investigation.

Discussion

We briefly introduce limitations of explanation methods we have surveyed and then present explanation formats that might be more understandable and friendly to users.

Limitations of current explanations.

A major limitation of existing work on interpretable machine learning is that the explanations are designed based on the intuition of researchers rather than focusing on the demands of endusers. Current local explanations are usually given in the format of feature importance vectors, which are a complete causal attribution and a low-level explanation.²³ This format would be satisfactory if the explanation audiences are developers and researchers, since they can utilize the statistical analysis of the feature importance distribution to debug the models. Nevertheless, this format is less friendly if the explanation receivers are lay users of machine learning. It describes the full decision logic of a model, which contains a huge amount of redundant information and will be overwhelming to users. The presentation formats could be further enhanced to better promote user satisfaction.

Toward human-friendly explanations. Based on findings in social sciences and human behavioral studies,²² we provide some directions toward user-oriented explanations, which might be more satisfying to humans as a means of communication.

Contrastive explanations. They are also referred as differential explanations.²² They do not tell why a specific



Model explanation and surprising artifacts are often two sides of the same coin.



prediction was made, but rather explain why this prediction was made instead of another, so as to answer questions like “Why Q rather than R ?” Here Q is the fact that requires explanation, and R is the comparing case, which could be a real one or virtual one. Consider, for instance, a user is declined mortgage. The user may compare with another real case and raise question: “Why didn’t I get a mortgage when my neighborhood did?” On the other hand, the user may ask: “Why was my mortgage rejected?” Here is an implicit contrast case, and actually the user is requesting explanation for a virtual case “How to get my mortgage loan approved?” Since it is compared to an event that has not happened, thus the desirable explanation here can also be called counterfactual explanation.³⁷

To provide contrastive explanations for a model prediction, similar strategy could be used for both comparisons mentioned earlier. We first produce feature importance attribution for two instances: not-accepted case for the user, has-accepted case of a neighbor (or would-be-accepted case of the user), and then compare the two attribution vectors. Note that we could resort to adversarial perturbation to find the would-be-accepted case. Besides, it is recommended to provide a diverse set of reasons, that is, to find multiple contrast cases, to make the explanation more informative. Ultimately, we generate explanations of the form: “Your mortgage is rejected because your income is lower than your neighbor’s, your credit history is not as strong as your neighbor’s ... or “Your mortgage would be accepted if your income is raised from x to y .”

Selective explanations. Usually, users do not expect an explanation can cover the complete cause of a decision. Instead, they wish the explanation could convey the most important information that contributes to the decision.²² A sparse explanation, which includes a minimal set of features that help justify the prediction is preferred, although incompletely. Still use the mortgage case for example. One good explanation could be presenting users the top two reasons contributing to the decision, such as poor credit history or low income to debt ratio.

Credible explanations. Good explanation might be consistent with prior

knowledge of general users.²³ Suppose the generated top reasons for the mortgage case include marital status is single and education status is high school graduate, then it would be less trustworthy than an explanation outputting poor credit history and low income to debt ratio, since the latter two are more reasonable causes leading to rejection. Low credibility could be caused by the poor fidelity of explanation to the original model. On the other hand, the explanations maybe faithful, however, the machine learning model does not adopt correct evidences to make decisions.


Conversational explanations. Explanations might be delivered as a conversation between the explainer and explanation receivers.²² It means we must consider the social context, that is, to whom an explanation is provided,³⁵ in order to determine the content and formats of explanations. For instance, a preferred format is verbal explanation if it is explaining to lay-users.

Note there are many other paths to user-friendly explanations. We refer interested readers to the survey by Miller²² for a comprehensive list of directions. All the aforementioned directions serve an identical purpose that explanation should tell users why a decision was reached in a concise and friendly manner. More importantly, the explanation could inform users what could be possibly changed to receive a desired decision next time. Granted, there is still a long way to go to render explanations that promote user's satisfaction. In the future, researchers from different disciplines, including machine learning, human-computer interaction, and social science, are encouraged to closely cooperate to design really user-oriented and human-friendly explanations.

Conclusion

Interpretable machine learning is an open and active field of research, with numerous approaches continuously emerging every year. We have presented a clear categorization and comprehensive overview of existing techniques for interpretable machine learning, aiming to help the community to better understand the capabilities and weaknesses of different interpretation approaches. Although techniques for interpretable machine learning are advancing quickly, some key challenges remain unsolved,

and future solutions are needed to further promote the progress of this field.

Acknowledgments. We thank the four anonymous reviewers for their helpful suggestions. This work is in part supported by NSF grants IIS-1657196, IIS-1718840, and DARPA grant N66001-17-2-4031. 

References

1. Altmann, A., Tološi, L., Sander, O. and Lengauer, T. Permutation importance: A corrected feature importance measure. *Bioinformatics* 26, 10 (2010), 1340–1347.
2. Ancona, M., Ceolini, E., Oztireli, C. and Gross, M. Towards better understanding of gradient-based attribution methods for deep neural networks. In *Proceedings of the Intern. Conf. Learning Representations*, 2018.
3. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R. and Samek, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS One* 10, 7 (2015), e0130140.
4. Bahdanau, D., Cho, K. and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *Proceedings of the Intern. Conf. Learning Representations*, 2015.
5. Bastani, O., Kim, C., and Bastani, H. Interpretability via model extraction. In *Proceedings of the Fairness, Accountability, and Transparency in Machine Learning Workshop*, 2017.
6. Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M. and Elhadad, N. Interpretable models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the ACM SIGKDD Intern. Conf. Knowledge Discovery and Data Mining*, ACM, 2015.
7. Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the ACM SIGKDD Intern. Conf. Knowledge Discovery and Data Mining*, ACM, 2016.
8. Dabkowski, P. and Gal, Y. Real time image saliency for black box classifiers. *Advances in Neural Information Processing Systems* (2017), 6970–6979.
9. Dix, A. Human issues in the use of pattern recognition techniques. *Neural Networks and Pattern Recognition in Human Computer Interaction* (1992), 429–451.
10. Doshi-Velez, F. and Kim, B. Towards a rigorous science of interpretable machine learning. 2017.
11. Du, M., Liu, N., Song, Q. and Hu, X. Towards explanation of DNN-based prediction with guided feature inversion. In *Proceedings of the ACM SIGKDD Intern. Conf. Knowledge Discovery and Data Mining*, 2018.
12. Du, M., Liu, N., Yang, F. and Hu, X. On attribution of recurrent neural network predictions via additive decomposition. In *Proceedings of the WWW Conf.*, 2019.
13. Fong, R. and Vedaldi, A. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the Intern. Conf. Computer Vision*, 2017.
14. Freitas, A.A. Comprehensive classification models: A position paper. *ACM SIGKDD Explorations Newsletter*, 2014.
15. Goodfellow, I., Bengio, Y. and Courville, A. *Deep Learning*, Vol.1. MIT Press, Cambridge, MA, 2016.
16. Goodfellow, I.J., Shlens, J. and Szegedy, C. Explaining and harnessing adversarial examples. In *Proceedings of the Intern. Conf. Learning Representations*, 2015.
17. Kádár, A., Chrupała, G., and Alishahi, A. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics* 43, 4 (2017), 761–780.
18. Karpathy, A., Johnson, J., and Fei-Fei, L. Visualizing and understanding recurrent networks. In *Proceedings of the ICLR Workshop*, 2016.
19. Liu, N., Du, M., and Hu, X. Representation interpretation with spatial encoding and multimodal analytics. In *Proceedings of the ACM Intern. Conf. Web Search and Data Mining*, 2019.
20. Liu, N., Yang, H., and Hu, X. Adversarial detection with model interpretation. In *Proceedings of the ACM SIGKDD Intern. Conf. Knowledge Discovery and Data Mining*, 2018.
21. McCullagh, P. and Nelder, J.A. *Generalized Linear M*, Vol. 37. CRC Press, 1989.
22. Miller, T. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* (2018).
23. Molnar, C. *Interpretable Machine Learning* (2018);

<https://christophm.github.io/interpretable-ml-book/>.

24. Mudrakarta, P.K., Taly, A., Sundararajan, M. and Dhamdhere, K. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Assoc. Computational Linguistics*, 2018.
25. Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T. and Clune, J. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *Advances in Neural Information Processing Systems*, 2016.
26. Nguyen, A., Yosinski, J. and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conf. Computer Vision and Pattern Recognition*, 2015.
27. Nguyen, A., Yosinski, J. and Clune, J. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. In *Proceedings of the ICLR Workshop*, 2016.
28. Peters, M.E. et al. Deep contextualized word representations. In *Proceedings of the NAACL-HLT*, 2018.
29. Quinlan, J.R. Simplifying decision trees. *Intern. J. Man-Machine Studies* 27, 3 (1987), 221–234.
30. Ribeiro, M.T., Singh, S. and Guestrin, C. Why should I trust you? Explaining the predictions of any classifier. In *Proceedings of the ACM SIGKDD Intern. Conf. Knowledge Discovery and Data Mining*, 2016.
31. Ribeiro, M.T., Singh, S. and Guestrin, C. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conf. Artificial Intelligence*, 2018.
32. Sabour, S., Frosst, N. and Hinton, G.E. Dynamic routing between capsules. *Advances in Neural Information Processing Systems*, 2017.
33. Simonyan, K., Vedaldi, A. and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Proceedings of the ICLR Workshop*, 2014.
34. Springenberg, J.T., Dosovitskiy, A., Brox, T. and Riedmiller, M. Striving for simplicity: The all convolutional net. In *Proceedings of the ICLR workshop*, 2015.
35. Tomsett, R., Braines, D., Harborne, D., Preece, A. and Chakraborty, S. Interpretable to whom? A role-based model for analyzing interpretable machine learning systems. In *Proceedings of the ICML Workshop on Human Interpretability in Machine Learning*, 2018.
36. Vandewiele, G., Janssens, G., Ongenaer, O., and Van Hoecke, F.S. Genesim: Genetic extraction of a single, interpretable model. In *Proceedings of the NIPS Workshop*, 2016.
37. Wachter, S., Mittelstadt, B. and Russell, C. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. 2017.
38. Xu, K. et al. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the Intern. Conf. Machine Learning*, 2015.
39. Zhang, Q., Wu, Y.N. and Zhu, S.-C. Interpretable convolutional neural networks. In *Proceedings of the IEEE Conf. Computer Vision and Pattern Recognition*, 2018.
40. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A. and Torralba, A. Object detectors emerge in deep scene CNNs. In *Proceedings of the Intern. Conf. Learning Representations*, 2015.

Mengnan Du (dumengnan@tamu.edu) is a graduate student in the Department of Computer Science and Engineering at Texas A&M University, College Station, TX, USA.

Ninghao Liu (nhliu43@tamu.edu) is a graduate student in the Department of Computer Science and Engineering at Texas A&M University, College Station, TX, USA.

Xia Hu (xiahu@tamu.edu) is an assistant professor in the Department of Computer Science and Engineering at Texas A&M University, College Station, TX, USA.

© 2020 ACM 0001-0782/20/1



Watch the authors discuss this work in the exclusive *Communications* video. <https://cacm.acm.org/videos/interpretable-machine-learning>

A 50-year history of concurrency.

BY SERGIO RAJSBAUM AND MICHEL RAYNAL

Mastering Concurrent Computing through Sequential Thinking

I must appeal to the patience of the wondering readers, suffering as I am from the sequential nature of human communication.

—E.W. Dijkstra, 1968¹²

ONE OF THE most daunting challenges in information science and technology has always been mastering concurrency. Concurrent programming is enormously difficult because it copes with many possible nondeterministic behaviors of tasks being done at the same time. These come from different sources, including failures, operating systems, shared memory architectures, and asynchrony. Indeed, even today

we do not have good tools to build efficient, scalable, and reliable concurrent systems.

Concurrency was once a specialized discipline for experts, but today the challenge is for the entire information technology community because of two disruptive phenomena: the development of networking communications, and the end of the ability to increase processors speed at an exponential rate. Increases in performance come through concurrency, as in multicore architectures. Concurrency is also critical to achieve fault-tolerant, distributed services, as in global databases, cloud computing, and blockchain applications.

Concurrent computing through sequential thinking. Right from the start in the 1960s, the main way of dealing with concurrency has been by reduction to sequential reasoning. Transforming problems in the concurrent domain into simpler problems in the sequential domain, yields benefits for specifying, implementing, and verifying concurrent programs. It is a two-sided strategy, together with a bridge connecting the two sides.

First, a *sequential specification* of an object (or service) that can be ac-

» key insights

- A main way of dealing with the enormous challenges of building concurrent systems is by reduction to sequential thinking. Over more than 50 years, more sophisticated techniques have been developed to build complex systems in this way.
- The strategy starts by designing sequential specifications, and then mechanisms to associate a concurrent execution to a sequential execution that itself can be tested against the sequential specification.
- The history starts with concrete, physical mutual exclusion techniques, and evolves up to today, with more abstract, scalable, and fault-tolerant ideas including distributed ledgers.
- We are at the limits of the approach, encountering performance limitations by the requirement to satisfy a sequential specification, and because not all concurrent problems have sequential specifications.



cessed concurrently states the desired behavior only in executions where the processes access the object one after the other, sequentially. Thus, familiar paradigms from sequential computing can be used to specify shared objects, such as classical data structures (for example, queues, stacks, and lists), registers that can be read or modified, or database transactions. This makes it easy to understand the object being implemented, as opposed to a truly concurrent specification which would be hard or unnatural. Instead of trying to modify the well-understood notion of say, a queue, we stay with the usual sequential specification, and move the meaning of a concurrent implementation of a queue to another level of the system.

The second part of the strategy is to provide *implementation techniques* for

efficient, scalable, and fault-tolerant concurrent objects. Locks enforce exclusive accesses to shared data, and concurrency control protocols. More abstract and fault-tolerant solutions that include *agreement* protocols that can be used for replicated data to locally execute object operations in the same order. Reliable *communication protocols* such as atomic broadcast and gossiping are used by the processes to communicate with each other. Distributed data structures, such as blockchains. Commit protocols to ensure *atomicity* properties. Several techniques are commonly useful, such as timestamps, quorums, group membership services, and failure detectors. Interesting liveness issues arise, specified by progress conditions to guarantee that operations are actually executed.

The bridge establishes a connection

between the executions of a concurrent program and the sequential specification. It enforces *safety* properties by which concurrent executions appear as if the operations invoked on the object where executed instantaneously, in some sequential interleaving. This is captured by the notion of a consistency condition, which defines the way concurrent invocations to the operations of an object correspond to a sequential interleaving, which can then be tested against the its sequential specification.

A brief history and some examples. The history of concurrency is long and the body of research enormous; a few milestones are in the sidebar “A Few Dates from the History of Synchronization.” The interested reader will find many more results about principles of concurrency in textbooks.^{4,21,35,37} We concentrate here only on a few signifi-

A Few Dates from the History of Synchronization

1965	● Mutual exclusion from atomic read/write registers	Dijkstra ¹¹
1965	● Semaphores	Dijkstra ¹³
1971	● Mutual exclusion from non-atomic read/write registers	Lamport ²³
1974	● Concurrent reading and writing	Lamport, ²⁴ Peterson ³³
1977, 1983	● Distributed state machine (DA 2000)	Lamport ²⁵
1980	● Byzantine failures in synchronous systems (DA 2005)	Pease, Shostak, Lamport ³¹
1981	● Simplicity in mutex algorithms	Peterson ³²
1983	● Asynchronous randomized consensus (DA 2015)	Ben-Or, ⁵ Rabin ³⁴
1985	● Liveness (progress condition) (DA 2018)	Alpern, Schneider ¹
1985	● Impossibility of asynchronous deterministic consensus in the presence of process crashes (DA 2001)	Fischer, Lynch, Paterson ¹⁶
1987	● Fast mutual exclusion	Lamport ²⁷
1991	● Wait-free synchronization (DA 2003)	Herlihy ¹⁹
1993, 1997	● Transactional memory (DA 2012)	Herlihy, Moss, ²⁰ Shavit, Touitou ⁴⁰
1995	● Shared memory on top of asynchronization message-passing systems despite a minority of process crashes (DA 2011)	Attiya, Bar Noy, Dolev ²
1996	● Weakest information on failures to solve consensus in the presence of asynchrony and process crashes (DA 2010)	Chandra, Hadzilacos, Toueg ⁸
2008	● Scalability, accountability	Nakamoto ³⁰

A paper that received the Dijkstra ACM Award in the year X is marked (DA X).

cant examples of sequential reasoning used to master concurrency, providing a sample of fundamental notions of this approach, and we describe several algorithms, both shared memory and message passing, as a concrete illustration of the ideas.

We tell the story through an evolution that starts with mutual exclusion, followed by implementing read/write registers on top of message passing systems, then implementing arbitrary objects through powerful synchronization mechanisms. We discuss the modern distributed ledger trends of doing so in a highly scalable, tamper-proof way. We conclude with a discussion of the limitations of this approach: It may

be that it is expensive to implement, and furthermore, there are inherently concurrent problems with no sequential specifications.

Mutual Exclusion

Concurrent computing began in 1961 with what was called *multiprogramming* in the Atlas computer, where concurrency was simulated—as we do when telling stories where things happen concurrently—interlacing the execution of sequential programs. Concurrency was born in order to make efficient use of a sequential computer, which can execute only one instruction at a time, giving users the illusion that their programs are all running simultaneously, through the operating system. A collection of early

foundational articles on concurrent programming appears in Brinch.⁶

As soon as the programs being run concurrently began to interact with each other, it was realized how difficult it is to think concurrently. By the end of the 1960s a crisis was emerging: programming was done without any conceptual foundation and programs were riddled with subtle errors causing erratic behaviors. In 1965, Dijkstra discovered that *mutual exclusion* of parts of code is a fundamental concept of programming and opened the way for the first books of principles on concurrent programming, which appeared at the beginning of the 1970s.

Locks. A mutual exclusion algorithm consists of the code for two operations—`acquire()` and `release()`—that a process invokes to bracket a section of code called a *critical section*. The usual environment in which it is executed is asynchronous, where process speeds are arbitrary, independent from each other. A mutual exclusion algorithm guarantees two properties.

► **Mutual exclusion.** No two processes are simultaneously executing their critical section.

► **Deadlock-freedom.** If one or several processes invoke `acquire()` operations that are executed concurrently, eventually one of them terminates its invocation, and consequently executes its critical section.

Deadlock-freedom does not prevent specific timing scenarios from occurring in which some processes can never enter their critical section. The stronger starvation-freedom progress condition states that any process that invokes `acquire()` will terminate its invocation (and will consequently execute its critical section).

A mutual exclusion algorithm. The first mutual exclusion algorithms were difficult to understand and prove correct. We describe here an elegant algorithm by Peterson³² based on read/write shared registers. Algorithms for a message-passing system have been described since Lamport's logical clock paper.²⁵

The version presented in Algorithm 1 is for two processes but can be easily generalized to n processes. The two processes p_1 and p_2 share three read/write atomic registers, $FLAG[1]$, $FLAG[2]$, and $LAST$. Initially $FLAG[1]$,

Algorithm 1. Peterson's mutual exclusion algorithm for two processes.

```

operation acquire() is % invoked by  $p_i, i \in \{1, 2\}$ 
   $FLAG[i] \leftarrow \text{up}; LAST \leftarrow i; \text{let } j = 3 - i;$ 
  wait ( $(FLAG[j] = \text{down}) \vee (LAST = i)$ );
  return()
end operation.

operation release() is  $FLAG[i] \leftarrow \text{down}; \text{return}()$  end operation.

```

FLAG[2], are down, while *LAST* does not need to be initialized. Both processes can read all registers. Moreover, while *LAST* can be written by both processes, only p_i , $i \in \{1, 2\}$, writes to *FLAG*[i]. Atomic means the read and write operations on the registers seem to have been executed sequentially (hence, the notion of “last writer” associated with *LAST* is well defined).

When process p_i invokes `acquire()`, it first raises its flag, thereby indicating it is competing, and then writes its name in *LAST* indicating it is the last writer of this register. Next, process p_i repeatedly reads *FLAG*[j] and *LAST* until it sees *FLAG*[j] = down or it is no longer the last writer of *LAST*. When this occurs, p_i terminates its invocation. The operation `release()` consists in a simple lowering of the flag of the invoking process. The read and write operations on *FLAG*[1], *FLAG*[2], and *LAST* are totally ordered (atomicity), which facilitates the proof of the mutual exclusion and starvation-freedom properties.

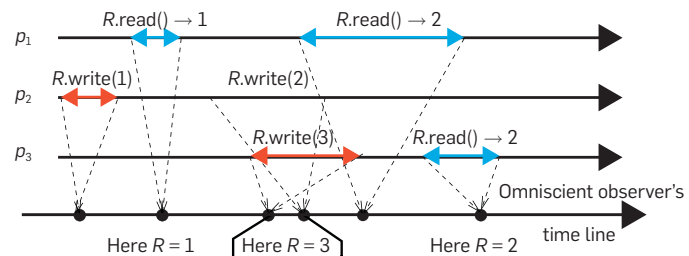
Mutual exclusion was the first mechanism for mastering concurrent programming through sequential thinking, and lead to the identification of notions that began to give a scientific foundation to the approach, such as the concepts of progress condition and atomicity. It is the origin of the important area of concurrency control, by controlling access to data using locks (for example, 2-phase locking).

From Resources to Objects

At the beginning, a critical section was encapsulating the use of a physical resource, which by its own nature, is sequentially specified (for example, disk, printer, processor). Conceptually not very different, locks were then used to protect concurrent accesses of simple data (such as a file). However, when critical sections began to be used to encapsulate more general shared objects, new ideas were needed.

Data is not physical resources. A shared object is different from a physical object, in that it does not a priori require exclusive access; a process can read the data of a file while another process concurrently modifies it. The lock-free approach (introduced by Lamport²⁴), makes possible to envisage implementations of purely digital objects without using mutual exclu-

An Atomic (Linearizable) Execution of Processes p_1, p_2 , and p_3 on xRead/Write Register R .



From an external observer point of view, it appears as if the operations were executed sequentially, at the sequence of linearization points of the read/write operations (indicated by the dotted arrows)

sion, in a way that operations can overlap in time.

Tolerating crash failures. Additionally, mutual exclusion cannot be used to implement an object in the presence of asynchrony and process crashes. If a process crashes inside its critical section, other processes, unable to tell if it crashed or is just slow, are prevented from accessing the object.

Consistency conditions. Wherever concurrent accesses to share data take place, a consistency condition is needed to define which concurrent operation executions are considered correct. Instead of transforming a concurrent execution into a sequential execution (as in mutual exclusion), the idea is to enforce that, from an external observer point of view, everything must appear as if the operations were executed sequentially. This is the *sequential consistency* notion (or *serializability*), which has been used since early 1976 in the database context to guarantee that transactions appear to have executed atomically.³⁸ However, sequential consistency is not composable. The stronger consistency condition of *linearizability* (or *atomicity*) requires that the total order of the operations respects the order on non-overlapping operations.²² Linearizability is illustrated in the sidebar “An Atomic (Linearizable) Execution of Processes,” that describes an execution in which three processes access an atomic read/write register R .

Read/Write Register on Top of Message-Passing Systems

Perhaps the most basic shared object is the read/write register. In the shared memory context, there are implementations from very simple registers (that only one process can write, and another can read), all the way to a multi-writer multireader (MWMR) register, that every process can write and every process can read; for example, see Herlihy.²¹

Distributed message-passing systems often support a shared memory abstraction and have a wide acceptance in both research and commercial computing, because this abstraction provides a more natural transition from uniprocessors and simplifies programming tasks. We describe here a classic fault-tolerant implementation of a read/write register on top of a message passing system.

It is relatively easy to build atomic read/write registers on top of a reliable asynchronous message-passing system, for example, Raynal,³⁶ but if processes may crash, more involved algorithms are needed. Two important results are presented by Attiya, Bar-Noy and Dolev:²

- An algorithm that implements an atomic read/write register on top of a system of n asynchronous message-passing processes, where at most $t < n/2$ of them may crash.

- A proof of the impossibility of

building an atomic read/write register when $t \geq n/2$.

This section presents the algorithm, referred to as the *ABD Algorithm*, which illustrates the importance of the ideas of reducing concurrent thinking to sequential reasoning. A more detailed proof as well as other algorithms can be found.^{2,4,37}

Design principles of ABD. Each written value has an identity. Each process is both a client and a server. Let *REG* be the multi-writer multi-reader (MWMR) register that is built (hence, any process is allowed to read and write the register). On its client side a process p_i can invoke the operations *REG.write*(v) to write a value v in *REG*, and *REG.read*() to obtain its current value. On its server side, a process p_j manages two local variables: *reg_j* which locally implements *REG*, and *timestamp_j*, which contains a timestamp made up of a sequence number (which can be considered as a date) and a process identity j . The timestamp *timestamp_j* constitutes the “identity” of the value v saved in *reg_j* (namely, this value was written by this process at this time). Any two timestamps $\langle sn_i, i \rangle$ and $\langle sn_j, j \rangle$ are totally ordered by their lexicographical order; namely, $\langle sn_i, i \rangle < \langle sn_j, j \rangle$ means $(sn_i < sn_j) \vee (sn_i = sn_j \wedge i < j)$.

Design principles of ABD: intersecting quorums. A process p_i broadcasts a query to all the processes and waits for acknowledgments from a majority of them. Such a majority quorum set, has the following properties. As $t < n/2$, waiting for acknowledgments from a majority of processes can never block forever the invoking process. Moreover, the fact that any two quorums have a non-empty intersection implies the atomicity property of the read/write register *REG*.

The operation *REG.write*(v). This operation is implemented by Algorithm 2. When a process p_i invokes *REG.write*(v), it first creates a tag denoted (*tag*) which will identify the query/response messages generated by this write invocation. Then (phase 1), it executes a first instance of the query/response exchange pattern to learn the highest sequence number saved in the local variables *timestamp_j* of a majority of processes p_j . When this is done, p_i computes the timestamp *ts* which will be associated with the value v it wants to write in *REG*. Finally (phase 2), p_i starts a second query/response pattern in which it broadcasts the pair (v, ts) to all the processes. When it has received the associated acknowledgments from a

quorum, p_i terminates the write operation.

On its server side, a process p_j that receives a *WRITE_REQ* message sent by a process p_i during phase 1 of a write operation, sends it back an acknowledgment carrying the sequence number associated with the latest value it saved in *reg_j*. When it receives *WRITE_REQ* message sent by a process p_j during phase 2 of a write operation, it updates its local data *reg_j* implementing *REG* if the received timestamp is more recent (with respect to the total order on timestamps) than the one saved in *timestamp_j*, and, in all cases, it sends back to p_i and acknowledgment (so p_i terminates its write).

It is easy to see that, due to the intersection property of quorums, the timestamp associated with a value v by the invoking process p_i is greater than the ones of the write operations that terminated before p_i issued its own write operation. Moreover, while concurrent write operations can associate the same sequence number with their values, these values have different (and ordered) timestamps.

The operation REG.read(). Algorithm 3 implements operation *REG.read*(), with a similar structure as the implementation of operation *REG.write*() .

Notice that the following scenario can occur, which involves two read operations read1 and read2 on a register *REG* by the processes p_1 and p_2 , respectively, and a concurrent write operation *REG.write*(v) issued by a process p_3 . Let *ts*(v) be the timestamp associated with v by p_3 . It is possible that the phase 1 majority quorum obtained by p_1 includes the pair ($v, ts(v)$), while the one obtained by p_2 does not. If this occurs, the first read operation read1 obtains a value more recent than the one obtained by the second read2, which violates atomicity. This can be easily solved by directing each read operation to write the value it is about to return as a result. In this way, when read1 terminates and returns v , this value is known by a majority of processes despite asynchrony, concurrency, and a minority of process crashes. This phenomenon (called *new/old inversion*) is prevented by the phase 2 of a read operation (as illustrated in the accompanying figure).

We have seen how the combination of intersecting quorums and

Algorithm 2. ABD's implementation of read/write register: write operation.

operation *REG.write*(v) issued by process p_i is

```

build a new tag tag identifying this write operation;
% Phase 1: acquire information on the system state %
broadcast WRITE_REQ(tag);
wait acknowledgments from a majority of processes,
each carrying tag and a sequence number;
% Phase 2: update system state %
 $ts \leftarrow \langle msn + 1, i \rangle$  where msn is
the greatest sequence number previously received;
broadcast WRITE(tag,  $v$ , ts);
wait acknowledgments carrying tag from a majority of proc.;
return().

```

when *WRITE_REQ*(*tag*) is received from $p_j, j \in \{1, \dots, n\}$ do

```

send to  $p_j$  an acknowledgment carrying tag, and
the sequence number contained in timestampj.

```

when *WRITE*(*tag*) is received from $p_j, j \in \{1, \dots, n\}$ do

```

if (timestampj < ts) then
    timestampj  $\leftarrow ts$ ; regj  $\leftarrow v$  end if;
send to  $p_j$  an acknowledgment carrying tag.

```

timestamps, two ideas useful in other situations, facilitate the implementation of atomic read/write registers in asynchronous message-passing systems where a minority of process may crash. And how sequential thinking for shared registers can be used at the upper abstraction level.

The World of Concurrent Objects

A read/write register is a special case of an object. In general, an *object* is defined by the set of operations that processes can invoke, and by the behavior of the object when these operations are invoked sequentially. These can be represented by an automaton or by a set of sequential traces. In the case of an automaton, for each state, and each possible operation invocation, a transition specifies a response to that invocation, and a new state (the transition is often a deterministic function, but not always). Thus, usual data structures from sequential programming, such as queues and stacks, can be used to define concurrent objects.

Consensus. At the core of many situations where sequential reasoning for concurrent programming is used (including state machine replication) are agreement problems. A common underlying abstraction is the consensus object. Let *CONS* be a consensus object. A process p_i can invoke the operation *CONS.propose*(v) once. The invocation eventually returns a value v' . This sequential specification for *CONS* is defined by the following properties.

- ▶ *Validity.* If an invocation returns v then there is a *CONS.propose*(v).
- ▶ *Agreement.* No two different values are returned.
- ▶ *Termination.* If a process invokes *CONS.propose*(v) and does not crash, the operation returns a value.

All objects are not equal in an asynchronous, crash-prone environment. Consensus objects are the strongest, in the sense that (together with read/write registers), they can be used to implement, despite asynchrony and process crashes, any object defined by a sequential specification. Other important objects, such as a queue or a stack are of intermediate strength: they cannot be implemented by asynchronous processes, which communicate using read/write registers only. Such imple-

mentations, that require that any operation invoked by a process that does not crash must return (independently of the speed or crashes of other processes), are said to be *wait-free*.

One way of measuring the synchronization power of an object in the presence of asynchrony and process crashes is by its consensus number. The consensus number of an object O is the greatest integer n , such that it is possible to wait-free implement a consensus object for n processes from any number of objects O and atomic read/write registers. The consensus number of O is ∞ if there is no such greatest integer. As an example, the consensus number of a Test&Set object or a stack object is 2, while the consensus number of a Compare&Swap or Load/Link&Store/Conditional (LL/SC) object is ∞ . We will discuss a LL/SC object later. These ideas were first discussed by Herlihy.¹⁹

State Machine Replication

A concurrent stack can be implemented by executing the operations *pop*() and *push*() using mutual exclusion. However, as already indicated, this strategy does not work if processes may crash. The *state machine replication mechanism*^{25,39} is a general way of implementing an object by asynchronous processes communicating by message-passing. We will discuss implementations where the processes may fail by crashing; there are also implementations that tolerate arbitrary (Byzantine) failures.⁷ We should point out that non-deterministic automata sometimes appear in applications and pose additional challenges for implementations.

The general idea is for the processes to agree on a sequential order of the concurrent invocations, and then each one to simulate the sequential specification automaton locally. We illustrate here the approach with a *total order*

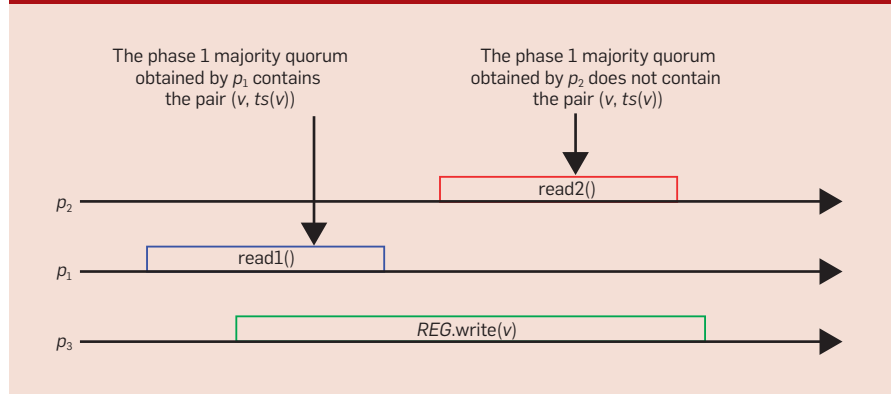
Algorithm 3. ABD's implementation of read/write register: read operation.

```

operation REG.read () is
    build a new tag tag identifying this read operation;
    % Phase 1: acquire information on the system state %
    broadcast READ_REQ (tag);
    wait acknowledgments from a majority of processes,
    each carrying tag and a pair (value,timestamp);
    let ts be the greatest timestamp received,
    and v the value associated with this timestamp;
    % Phase 2: update system state %
    broadcast WRITE (tag, v, ts );
    wait ACK_WRITE (tag) from a majority of proc.;
    return (v).

when READ_REQ (tag) is received from  $p_j, j \in \{1, \dots, n\}$  do
    send to  $p_j$  an ack. carrying tag, reg, and timestampi.
    
```

New/old inversion scenario.



Algorithm 4. TO-broadcast-based universal construction.

when operation $\text{op}_x(\text{param}_x)$ is invoked by process p_i do

```
resulti ← ⊥; let sent_msg = ⟨opx(paramx), i⟩;
TO_broadcast(sent_msg);
wait(resulti ≠ ⊥); return(resulti).
```

background task T is

repeat forever

```
rec_msg ← TO_deliver();
⟨statei, res⟩ ← δ(statei, rec_msg.op);
if(rec_msg.proc = i) then resulti ← res end if
end repeat.
```

Algorithm 5. Implementing TO-broadcast from consensus.

when p_i invokes $\text{TO_broadcast}(m)$ do send(m) to itself.

when m is received for the first time do

```
broadcast( $m$ ); pendingi ← pendingi ∪ { $m$ }.
```

when ($to_deliverable_i$ contains messages not yet to-delivered) do

```
let  $m$  = first message  $\in to\_deliverable_i$  not yet to-delivered;
TO_deliver( $m$ ).
```

background task T is

repeat forever

```
wait(pendingi \ to_deliverablei ≠ ∅);
let seq = (pendingi \ to_deliverablei);
order the messages in seq;
sni ← sni + 1; resi ← CS[sni].propose(seq);
add resi at the end of to_deliverablei;
end repeat.
```

broadcast mechanism for reaching the required agreement.

Total order broadcast. The TO-broadcast abstraction is an important primitive in distributed computing, which ensures that all correct processes receive messages in the same order.^{18,37} It is used through two operations, $\text{TO_broadcast}()$ and $\text{TO_deliver}()$. A process invokes $\text{TO_broadcast}(m)$, to send a message m to all other processes. As a result, processes execute $\text{TO_deliver}()$ when they receive a (totally ordered) message.

TO-broadcast illustrates one more general idea within the theory of mastering concurrent programming through sequential thinking: the identification of communication abstractions that facilitate building concurrent objects defined by a sequential specification.

State machine replication based on TO-broadcast. A concurrent implementation of object O is described in Algorithm 4. It is a universal construction, as it works for any object O defined by a sequential specification. The object has operations $\text{op}_x()$, and a transition function $\delta()$ (assuming δ is deterministic), where $\delta(\text{state}', \text{op}_x(\text{param}_x))$ returns the pair $\langle \text{state}', \text{res} \rangle$, where state' is the new state of the object and res is the result of the operation.

The idea of the construction is simple. Each process p_i has a copy state_i of the object, and the TO-broadcast abstraction is used to ensure that all the processes p_i apply the same sequence of operations to their local representation state_i of the object O .

Implementing TO-broadcast from consensus. Algorithm 5 is a simple construction of TO-broadcast on top of an asynchronous system where consensus objects are assumed to be available.¹⁸

Let $\text{broadcast}(m)$ stand for “**for each** $j \in \{1, \dots, n\}$ **do** send(m) to p_j **end for.**” If the invoking process does not crash during its invocation, all processes receive m ; if it crashes an arbitrary subset of processes receive m .

The core of the algorithm is the background task T . A consensus object $\text{CS}[k]$ is associated with the iteration number k . A process p_i waits until there are messages in the set pending_i and not yet in the queue to_deliverable_i . When this occurs, process p_i computes this set of messages (seq)

Circumventing Consensus Impossibility

Three ways of circumventing the consensus impossibility:

- ▶ The failure detector approach⁸ can abstract away synchrony assumptions sufficient to distinguish between slow processes and dead processes.
- ▶ In eventually synchronous systems¹⁴ there is a time after which the processes run synchronously. The celebrated Paxos algorithm is an example.²⁸
- ▶ By using random coins⁵ consensus is solvable with high probability.
- ▶ Often not all combinations of input values occur.²⁹

and order them. Then it proposes seq to the consensus instance $SC[k]$. This instance returns a sequence saved in res_i , which is added by p_i at the end of its local queue to $deliverable_i$.

When are Universal Constructions Possible?

An impossibility. A fundamental result in distributed computing is the impossibility to design a (deterministic) algorithm that solves consensus in the presence of asynchrony, even if only one process may crash, either in message-passing or read/write shared memory systems.¹⁶ Given that consensus and TO-broadcast are equivalent, the state machine replication algorithm presented above cannot be implemented in asynchronous systems where processes can crash.

Thus, sequential thinking for concurrent computing has studied properties about the underlying system that enable the approach to go through. There are several ways of considering computationally stronger (read/write or message-passing) models,^{35,37} where state machine replication can be implemented. Some ways, mainly suited to message-passing systems, are presented in the sidebar “Circumventing Consensus Impossibility.” Here, we discuss a different way, through powerful communication hardware.

Systems that include powerful objects. Shared memory systems usually include synchronization operations such as Test&Set, Compare&Swap, or the pair of operations Load Link and Store Conditional (LL/SC), in addition to read/write operations. These operations have a consensus number greater than 1. More specifically, the consensus number of Test&Set is 2, while the consensus number of both Compare&Swap and the pair LL/SC, is $+\infty$. Namely, 2-process (but not a 3-process) consensus can be implemented from Test&Set, despite crash failures. Compare&Swap (or LL/SC) can implement consensus for any number of processes. Hence, for any n , any object can be implemented in an asynchronous n -process read/write system enriched with Compare&Swap (or LL/SC), despite up to $n-1$ process crashes. Furthermore, there are implementations that tolerate arbitrary, malicious (Byzantine) failures.^{7,37}

State machine replication based on LL/SC. To give more intuition about state machine replication, and furthermore, about the way that blockchains work, we present an implementation based on LL/SC. (Another option is based on Compare&Swap, but it is not

“self-contained” in the sense it has to deal with the ABA problem.³⁵)

The intuition of how the LL/SC operations work is as follows. Consider a memory location M , initialized to \perp , accessed only by the operations LL/SC. Assume that if a process invokes $M.SC(v)$

Algorithm 6. Implementing a consensus object CONS from the operations LL/SC.

```
Initially:  $M = \perp$ 
operation  $CONS.propose(v)$  is
   $va\ l_i \leftarrow M.LL()$ ;
  if ( $va\ l_i \neq \perp$ ) then return( $va\ l_i$ )
    else  $b_i \leftarrow M.SC(v)$ ;
      if  $b_i$  then return( $v$ )
        else  $va\ l_i \leftarrow M.LL()$ ; return( $va\ l_i$ )
    end if
  end if.
```

Universal Construction based on LL/SC

when the operation op_x ($param_x$) is locally invoked do

```
 $sn_i \leftarrow sn_i + 1$ ;  $BOARD[i] \leftarrow \langle op_x(param_x), sn_i \rangle$ ;
apply();
 $state_i \leftarrow STATE.LL()$ ; return( $state_i.res[i]$ ).
```

internal procedure $apply()$ is

```
 $state_i \leftarrow STATE.LL()$ ;
 $board_i \leftarrow [BOARD[1], BOARD[2], \dots, BOARD[n]]$ ;
for  $\ell \in \{1, \dots, n\}$  do
  if ( $board_i[\ell].sn = state_i.sn[\ell] + 1$ )
    then  $\langle state_i.value, state_i.res[\ell] \rangle \leftarrow$ 
       $\delta(state_i.value, pairs_i[\ell].op)$ ; % line A
       $state_i.sn[\ell] \leftarrow state_i.sn[\ell] + 1$  % line B
    end if
  end for;
 $success \leftarrow STATE.SC(state_i)$ ;
if ( $\neg success$ ) then
   $state_i \leftarrow STATE.LL()$ ;
  if ( $sn_i = state_i.sn[i] + 1$ )
    then same as lines A and B with  $\ell = i$ ;
       $STATE.SC(state_i)$ 
    end if
end if
```

it has previously invoked $M.LL()$. The operation $M.LL()$ is a simple read of M which returns the current value of $M.LL()$. When a process p_i invokes $M.SC(v)$ the value v is written into M if and only if no other process invoked $M.SC()$ since its (p_i) last invocation of $M.LL()$. If the write succeeds $M.SC()$ returns `true`, otherwise it returns `false`.

Algorithm 6 is a simple implementation of consensus from the pair of operations LL/SC , which tolerates any number of process crashes.

In the sidebar “Universal Construction Based on LL/SC ,” there is a shared-memory, LL/SC based universal construction.¹⁵ Looking at the algorithm, one begins to get a feeling for the distributed ledgers discussed next.

Distributed Ledgers

Since ancient times, ledgers have been at the heart of commerce, to represent concurrent transactions by a permanent list of individual records sequentialized by date. Today we are beginning to see algorithms that enable the collaborative creation of digital distributed ledgers with properties and capabilities that go far beyond traditional physical ledgers. All participants within a network can have their own copy of the ledger. Any of them can append a record to the ledger, which is then reflected in all copies in minutes or even seconds. The records stored in the ledger can stay tamper-proof, using cryptographic techniques.

Ledgers as universal constructions. Mostly known because of their use in cryptocurrencies, and due to its *blockchain* implementation,³⁰ from the perspective of this paper a *distributed ledger* is a byzantine fault-tolerant replicated implementation of a specific *ledger* object. The ledger object has two operations, `read()` and `append()`. Its sequential specification is defined by a list of blocks. A block X can be added at the end of the list with the operation `append(X)`, while a `read()` returns the whole list. In the case of a cryptocurrency, X may contain a set of transactions.

Thus, a ledger object, as any other object, can be implemented using a Byzantine failures-tolerant state machine replication algorithm. Conversely, a ledger can be used as a universal construction of an object O defined by a state machine with a transition func-

While resources are physical objects, data is digital objects.

tion δ . To do so, when a process invokes `append(X)`, X consists of a transition to be applied to the state machine. The state of the object is obtained through a `read()` invocation, which returns the sequence of operations which have been sequentially appended to the ledger, and then locally applying them starting from the initial state of the object (see Raynal³⁷ for more details).

Three remarkable properties. The apparently innocent idea of a `read()` operation that returns the list of commands that have been applied to the state machine, opens the discussion of one of the remarkable points of distributed ledgers that has brought them to such wide attention. The possibility of guaranteeing a *tamperproof* list of commands. The blockchain implementation is by using cryptographic hashes that link each record to the previous one (although the idea has been known in the cryptography community for years).

The ledger implementation used in Bitcoin showed it is possible to have a state machine replication tolerating Byzantine failures that scales to hundreds of thousands of processes. The cost is temporarily sacrificing consistency—forks can happen at the end of the blockchain, which implies that the last few records in the blockchain may have to be withdrawn.

The third remarkable property brought to the public attention by distributed ledgers is the issue of who the participants can be. As opposed to classic algorithms for mastering concurrency through sequential thinking, the participants do not have to be a priori-known, can vary with time, and may even be anonymous. Anyone can append a block and `read` the blockchain (although there are also *permissioned* versions where participants have to be registered, and even hybrid models). In a sense, a distributed ledger is an open distributed database, with no central authority, where the data itself is distributed among the participants.

Agreement in dynamic, Byzantine systems. Bitcoin’s distributed ledger implementation is relatively simple to explain in the framework of state machine replication. Conceptually it builds on randomized consensus (something that had already been carefully studied in traditional approaches, as noted in the sidebar “Circumventing Consensus

Impossibility”), through the following ingenious technique to implement it. Whenever several processes (not necessarily known a priori, hence the name of “dynamic system”) want to concurrently append a block, they participate in a lottery. Each process selects a random number (by solving cryptographic puzzles) between 0 and some large integer K , and the one that gets a number smaller than $k \ll K$, wins, and has the right to append its desired block.

The implementation details of the lottery (by a procedure called *proof of work*) are not important for this article; what is important here is that with high probability only one wins (and selected at random). However, from time to time, more than one process wins, and a *fork* happens, with more than one block being appended at the end of the blockchain. Only one branch eventually prevails (in Bitcoin this is achieved by always appending to the longest branch). This introduces a new interesting idea into the paradigm of mastering concurrency through sequential thinking: a trade-off between faster state machine replication, and temporary loss of consistency. In other words, the x operations at the very end of the blockchain, for some constant x (which depends on the assumptions about the environment) cannot yet be considered committed.


The Limits of the Approach

It is intuitively clear, and it has been formally proved, that linearizability or even serializability may be costly. Recent papers in the context of shared memory programming, argue that it is often possible to improve performance of concurrent data structures by relaxing their semantics.⁹ In the context of distributed systems, eventual consistency is widely deployed to achieve high availability by guaranteeing that if no new updates are made to a given data item, eventually all accesses to that item will return the last updated value (despite its name is not technically a consistency condition.³). In the case of distributed ledgers, we have seen the benefit that can be gained by relaxing the sequential approach to mastering concurrency: branches at the end of the blockchain (such as Bitcoin) temporarily violate a consistent view of the ledger. Still, blockchains

suffer from a performance bottleneck due to the requirement of ordering all transactions in a single list, which has prompted the exploration of partially ordered ledgers, based on directed acyclic graphs such as those of Tangle or Hedera Hashgraph.

The CAP Theorem formalizes a fundamental limitation of the approach of mastering concurrency through sequential reasoning—at most, two of the following three properties are achievable: consistency, availability, partition tolerance.¹⁷ This may give an intuition of why distributed ledgers implementations have temporary forks. An alternative is a cost in availability and postpone the property that every non-failing participant returns a response for all operations in a reasonable amount of time. We have already seen in the ABD algorithm that the system continues to function and upholds its consistency guarantees, provided that only a minority of processes may fail.

Finally, another fundamental limitation to the approach of mastering concurrency through sequential reasoning is that not all concurrent problems of interest have sequential specifications. Many examples are discussed in Castañeda et al.,¹⁰ where a generalization of linearizability to arbitrary concurrent specifications is described.

Acknowledgment. The authors acknowledge support of UNAM-PAPIT IN106520, INRIA-Mexico Associate Team, CNRS-Mexico UMI. 

References

- Alpern, B. and Schneider, F.B. Defining liveness. *Information Processing Letters* 21, 4 (1985), 181–18.
- Attiya, H., Bar-Noy, A., and Dolev, D. Sharing memory robustly in message-passing systems. *JACM* 42, 1 (1995), 121–132.
- Attiya, H., Ellen, F. and Morrison, A., Limitations of highly-available eventually-consistent data stores. *IEEE Trans. Parallel Distributed Systems* 28, 1 (2017), 141–155.
- Attiya, H. and Welch, J. *Distributed Computing: Fundamentals, Simulations and Advanced Topics*. (2nd Edition), Wiley, 2004.
- Ben-Or, M. Another advantage of free choice: completely asynchronous agreement protocols. In *Proc. 2nd ACM Symp. on Principles of Distributed Computing*, (1983), 27–30.
- Brinch, H.P. (Ed.). *The Origin of Concurrent Programming*. Springer, (2002).
- Cachin, C. State machine replication with Byzantine faults. *Replication*. Springer LNCS 5959, (2011), 169–184.
- Chandra, T.D., Hadzilacos, V., and Toueg, S. The weakest failure detector for solving consensus. *JACM* 43, 4 (1996), 685–722.
- Calciu, I., Sen, S., Balakrishnan, M., and Aguilera, M. How to implement any concurrent data structure for modern servers. *ACM Operating Systems Rev.* 51, 1 (2017), 24–32.
- Castañeda, A., Rajsbaum, S., and Raynal, M. Unifying concurrent objects and distributed tasks: Interval-linearizability. *JACM* 65, 6 (2018), Article 45.
- Dijkstra, E.W. Solution of a problem in concurrent

- programming control. *Comm. ACM* 8, 8 (Sept. 1965), 569.
- Dijkstra, E.W. Cooperating sequential processes. *Programming Languages*. Academic Press, 1968, 43–112.
- Dijkstra, E.W. Hierarchical ordering of sequential processes. *Acta Informatica* 1, 1 (1971), 115–138.
- Dolev, D., Dwork, C., and Stockmeyer, L. On the minimal synchronism needed for distributed consensus. *JACM* 34, 1 (1987), 77–97.
- Fatourou, P. and Kallimanis, N.D. Highly-efficient wait-free synchronization. *Theory of Computing Systems* 55 (2014), 475–520.
- Fischer, M.J., Lynch, N.A., and Paterson, M.S. Impossibility of distributed consensus with one faulty process. *JACM* 32, 2 (1985), 374–382.
- Gilbert, S. and Lynch, N. Brewer’s conjecture and the feasibility of consistent, available, partition-tolerant web services. *SIGACT News* 33, 2 (2002), 51–59.
- Hadzilacos, V. and Toueg, S. A modular approach to fault-tolerant broadcasts and related problems. TR 94-1425. Cornell Univ. (1994)
- Herlihy, M.P. Wait-free synchronization. *ACM Trans. on Prog. Languages and Systems* 13, 1 (1991), 124–149.
- Herlihy, M.P. and Moss, J.E.B. Transactional memory: Architectural support for lock-free data structures. In *Proc. 20th ACM Int’l Symp. Computer Architecture*. ACM Press, 1993, 289–300.
- Herlihy, M. and Shavit, N. *The Art of Multiprocessor Programming*. Morgan Kaufmann, ISBN 978-0-12-370591-4 (2008).
- Herlihy, M.P. and Wing, J.M. Linearizability: A correctness condition for concurrent objects. *ACM Trans. Programming Languages and Systems* 12, 2 (1990), 463–492.
- Lamport, L. A new solution of Dijkstra’s concurrent programming problem. *Commun. ACM* 17, 8 (1974), 453–455.
- Lamport, L. Concurrent reading and writing. *Commun. ACM* 20, 11 (Nov. 1977), 806–811.
- Lamport, L. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM* 21, 7 (Sept. 1978), 558–565.
- Lamport, L. On interprocess communication, Part I: Basic formalism. *Distributed Computing* 1, 2 (1986), 77–85.
- Lamport, L. A fast mutual exclusion algorithm. *ACM Trans. Computer Systems* 5, 1 (1987), 1–11.
- Lamport, L. The part-time parliament. *ACM Trans. Computer Systems* 16, 2 (1998), 133–169.
- Mostéfaoui, Rajsbaum, S. and Raynal, M. Conditions on input vectors for consensus solvability in asynchronous distributed systems. *JACM* 50, 6 (2003), 922–954.
- Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. *Unpublished manuscript* (2008).
- Pease, M., Shostak, R. and Lamport, L. Reaching agreement in the presence of faults. *JACM* 27 (1980), 228–234.
- Peterson, G.L. Myths about the mutual exclusion problem. *Information Processing Letters* 2, 3 (1981), 115–116.
- Peterson, G.L. Concurrent reading while writing. *ACM Trans. on Prog. Languages and Systems* (1983), 5:46–5:55.
- Rabin, M. Randomized Byzantine generals. *Proc. 24th IEEE Symp. Foundations of Computer Science*. IEEE Computer Society Press, 1983, 116–124.
- Raynal, M. *Concurrent Programming: Algorithms, Principles and Foundations*. Springer, 2013, ISBN 978-3-642-32026-2.
- Raynal, M. *Distributed Algorithms for Message-Passing Systems*. Springer, 2013, ISBN 978-3-642-38122-5.
- Raynal, M. *Fault-Tolerant Message-Passing Distributed Systems: An Algorithmic Approach*. Springer, 2018, ISBN 978-3-319-94140-0.
- Stearns, R.C., Lewis, P.M. and Rosenkrantz, D.J. Concurrency control for database systems. In *Proc. 16th Conf. Found. Comp. Sci.*, (1976), 19–32.
- Schneider, F.B. Implementing fault-tolerant services using the state machine approach. *ACM Computing Surveys* 22, 4 (1990), 299–319.
- Shavit, N. and Touitou, D. Software transactional memory. *Distributed Computing* 10, 2 (1997), 99–116.

Sergio Rajsbaum (rajsbaum@im.unam.mx) is a professor at the Instituto de Matemáticas at the Universidad Nacional Autónoma de México inn Mexico City, México.

Michel Raynal (raynal@irisa.fr) is a professor at IRISA, University of Rennes, France, and Polytechnic University in Hong Kong.

ACM Transactions on Evolutionary Learning and Optimization (TELO)

Open for Submissions

Publishes papers at the intersection of optimization and machine learning, making solid contributions to theory, method and applications in the field.



ACM Transactions on Evolutionary Learning and Optimization (TELO) publishes high-quality, original papers in all areas of evolutionary computation and related areas such as population-based methods, Bayesian optimization, or swarm intelligence.

We welcome papers that make solid contributions to theory, method and applications. Relevant domains include continuous, combinatorial or multi-objective optimization. Applications of interest include but are not limited to logistics, scheduling, healthcare, games, robotics, software engineering, feature selection, clustering as well as the open-ended evolution of complex systems.

We are particularly interested in papers at the intersection of optimization and machine learning, such as the use of evolutionary optimization for tuning and configuring machine learning algorithms, machine learning to support and configure evolutionary optimization, and hybrids of evolutionary algorithms with other optimization and machine learning techniques.

For more information and to submit your work, please visit:

telo.acm.org



Association for Computing Machinery

research highlights

P. 90

Technical Perspective Is There a Geek Gene?

By Mark Guzdial

P. 91

Evidence That Computer Science Grades Are Not Bimodal

By Elizabeth Patitsas, Jesse Berlin, Michelle Craig, and Steve Easterbrook

Technical Perspective

Is There a Geek Gene?

By Mark Guzdial

MANY COMPUTER SCIENCE teachers have told me that some students just *get* computer science—and others do not. We certainly have a lot of evidence that students enter introductory computer science courses with big differences in skills. Some students have already had years of programming experience, while others have never programmed at all. The question is whether those gaps close or diverge further.

Are the differences between students in CS classes explained by experience and background, or are the differences innate? Innate difference among CS students has been dubbed the *Geek Gene*. Many CS teachers believe a Geek Gene (or something similar) is necessary to succeed in CS, and not everyone has it. A 2007 study found 77% of surveyed CS faculty strongly disagreed with the statement: “Nearly everyone is capable of succeeding in computer science if they work at it.” CS teachers point to a bimodal distribution of grades in their CS classes as evidence for its existence. Some students “get it” and do well, while others do not, which appears as two peaks in a grade distribution. Is it real? Are some students born to be computer scientists, and are others unlikely to succeed because they

Are some students born to be computer scientists, and are others unlikely to succeed because they do not have the right stuff?

do not have the right stuff?

There is a long history of researchers trying to discover the variables that predict student success in computer science class. Probably the most famous of these had the odd title “The Camel has Two Humps.” It was never published in a peer-reviewed venue, was not replicated in multiple attempts, and was later retracted—but its power persists because it rings true to many. The underlying research questions are important: What skills and knowledge predict success in CS? How can we measure them? Can we teach any missing but necessary skills and knowledge explicitly?


The following paper “Evidence that Computer Science Grades Are Not Bimodal” by Elizabeth Patitsas, Jesse Berlin, Michelle Craig, and Steve Easterbrook takes aim at belief in the Geek Gene. If there is a Geek Gene, one would expect bimodal grade distributions in CS classes. If grades are not bimodal, perhaps the Geek Gene is just a figment of teachers’ biases. The authors explicitly check a large corpus of grade data for bimodality, and then run a study with CS teachers as participants to determine if belief in innate differences may itself explain why teachers see bimodality in grades. This paper is important for showing student performance may not be bimodal and for offering evidence of an alternative, plausible hypothesis.

► First, they review all final grades in every undergraduate class from 1996 to 2013 at the University of British Columbia (UBC). This dataset included over 700 sections and over 30,000 grades. 85% of the grade distributions were normally distributed.

► Then, they run a deception study. They recruit 60 CS instructors. Half were asked to agree or disagree with statements like in the 2007 study: “Some students are innately predisposed to do better at CS than others,” and then shown a set of histograms representing grade distributions. The

instructors were asked to label which were bimodal and which were not. The other half of the instructors saw the histograms first, and then were asked to respond to the statements. The deception was that *all* the histograms were generated from normal distributions, yet participants who agreed with the Geek Gene statements were more likely to identify the distributions as bimodal.

As in all empirical studies involving humans, we can disagree about the details. How UBC counts withdrawing from a class or failing a class in the grade distribution is probably different than many institutions. There is some possibility that participants might have seen the histograms, then gone back to change their answers on the statements. There can and should be more studies on these questions.

This paper does not prove there is no Geek Gene. There may actually be bimodality in CS grades at some (or even many) institutions. What this paper does admirably is to use empirical methods to question some of our long-held (but possibly mistaken) beliefs about CS education. Through papers like these, we will learn to measure and improve computing education, by moving it from folk wisdom to evidence-based decision-making. 

Mark Guzdial (mjguz@umich.edu) is a professor of electrical engineering and computer science in the College of Engineering and a professor of information in the School of Information at the University of Michigan, Ann Arbor, MI, USA.

Evidence That Computer Science Grades Are Not Bimodal

By Elizabeth Patitsas, Jesse Berlin, Michelle Craig, and Steve Easterbrook

Abstract

Although it has never been rigorously demonstrated, there is a common belief that grades in computer science courses are bimodal. We statistically analyzed 778 distributions of final course grades from a large research university and found that only 5.8% of the distributions passed tests of multimodality. We then devised a psychology experiment to understand why CS educators believe their grades to be bimodal. We showed 53 CS professors a series of histograms displaying ambiguous distributions that we asked them to categorize. A random half of participants were primed to think about the fact that CS grades are commonly thought to be bimodal; these participants were more likely to label ambiguous distributions as “bimodal.” Participants were also more likely to label distributions as bimodal if they believed that some students are innately predisposed to do better at CS. These results suggest that bimodal grades are instructional folklore in CS, caused by confirmation bias and instructor beliefs about their students.

1. INTRODUCTION

It is a prevailing belief in the computer science education community that CS grades are bimodal, and much time has been spent speculating and exploring why that could be (For a review, see Ahadi and Lister¹.) These discussions generally do not include statistical testing of whether the CS grades are bimodal in the first place. From what we have seen, people take a quick visual look at their grade distribution, and if they see two peaks, they conclude that it is bimodal. But eyeballing a distribution is unreliable; for example, if you expect the data to have a certain distribution, you are more likely to see it.

Anecdotally, we have seen new instructors and TAs (and students) who have shown histograms of grades and told the grades were “bimodal.” The bimodality perception hence becomes an organizational belief, and those who enter the community of practice of CS educators are taught this belief.

1.1. Explanations for bimodal grades

A number of explanations have been presented for why CS grades are bimodal, all of which begin with the assumption that this is the case.

Prior experience. A bimodal distribution generally indicates that two distinct populations have been sampled together.⁵ One explanation for bimodal grades is that CS1 classes have two populations of students: those with experience, and those without.¹

In many places, high school CS is not common or standardized, and so students enter university CS with differing amounts of prior experience. However, this explanation fits

students into only two bins. Prior experience is not as simple as “have it” vs. not-there is a wide range of how much prior programming experience students may have, and practice with nonprogramming languages such as HTML/CSS could also be beneficial.¹⁸

Learning edge momentum, stumbling points, and threshold concepts. One family of explanations posits that some CS concepts are more difficult for students to learn, and if they miss these concepts, they fall behind, whereas their peers advance ahead of them. As it is typically taught, CS1 builds on itself heavily. So once a student falls behind, they continue to fall further and further behind.¹ This may be exacerbated by the fact that some concepts may be key to understanding (“threshold concepts”). One might think of this explanation as a variant of the prior-experience explanation, where the students who have better study skills succeed, and those with weaker skills fall behind.

The Geek Gene Hypothesis. Some would instead argue that the two populations in CS1 classes are those who have some “natural talent,” giftedness, or predisposition to succeed at computing. Guzdial has referred to this belief as the “Geek Gene Hypothesis”.⁶ This belief appears to be quite prevalent. In a survey of CS faculty, Lewis found that 77% of them strongly disagree with the statement “Nearly everyone is capable of succeeding in the computer science curriculum if they work at it.”¹⁴ However, there seems to be little evidence that there is indeed a “Geek Gene”, and plenty of evidence that effective pedagogy allows for all students to succeed.⁸

Coarse assessment. Another line of explanation implicates instructors’ assessment tools as the source of bimodally distributed grades.^{28,20} A common trend on CS exams is to ask a series of long-answer coding questions. Zingaro et al. found that these questions offer only coarse assessment information to instructors: students either put all the pieces together, or fail to. Instructors do not adequately identify when a student has partial understanding nor quantify how much understanding a student has of a concept.

As an alternative, Zingaro et al. experimentally compared using short-answer questions that build upon each other to have one isomorphic long-answer question. When the different conceptual parts of the question were broken up, the resulting grades were normally distributed. The all-or-nothing nature of long-answer questions could lead to grades more likely to be (or appear) bimodal.²⁸

Or perhaps CS grades are not bimodal? A competing view

The original version of this paper was published in the *Proceedings of the 2016 ACM Conference on International Computing Education Research (ICER)*.

of CS grades put forth by Lister is that the grades are not, in fact, bimodal.¹⁵ Lister observed that CS grade distributions are generally noisy, and in line with what statisticians would accept as normally distributed. Lister argued that the perception of bimodal grades results from instructors' beliefs in the Geek Gene Hypothesis, and hence, instructors see bimodality where there is none.¹⁵ Lister's argument was theoretical, and based on statistical theory; in this paper, we test his argument by statistically analyzing actual grade distributions.

2. WHAT IS A BIMODAL DISTRIBUTION?

To properly tackle the question of "are CS grades bimodal?", we should first clearly establish what bimodality means. For a comprehensive discussion of this, we suggest the reader consult²⁵; we summarize some major points of that article in this section.

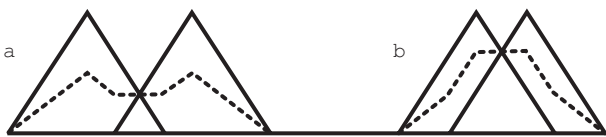
Most standard continuous probability distributions have a mean, a median, a mode, and some measure of the distribution's width (variance). Standard distributions include the normal (Gaussian), Pareto, Poisson, Cauchy, Student's t, and logistic distributions. When we plot them (or likely, a sample thereof) with a histogram, we see their probability density. All of these distributions have a single mode, and have a probability density that can be modeled with a function that has a single term. For example, the normal distribution's PDF is

$$f(x) = ae^{-\frac{(x-b)^2}{2c^2}}$$

In this function, a represents the height of the curve's peak, b is the position of the center of the peak, and c represents the width of the curve.²⁷

In contrast, a bimodal distribution has two *distinct* modes. A 'multimodal' distribution is any distribution with multiple distinct modes (two or more). For example, consider these examples from.²⁵ Both are created by the equal mixture of two triangular distributions (solid lines). The sums are shown with dashed lines:

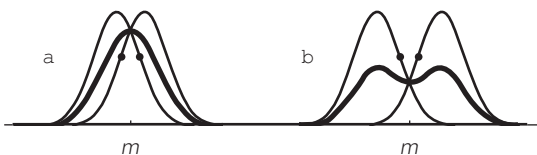
As we can see, when the two subdistributions are far away



(example a), we get a distribution with two peaks. But when the two subdistributions are close together (example b), they add together to form a plateau, with a single peak. Example a is considered bimodal; example b is not.

The same is true for normal distributions (also from Schilling et al.²⁵):

For a distribution to be bimodal, the subdistributions



cannot overlap too much. As shown in Schilling et al.²⁵, for the two distributions to be sufficiently far apart, the distance

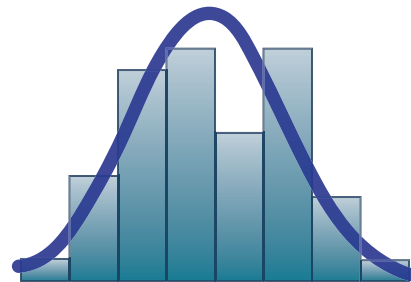
between the means of the two distributions needs to exceed 2σ . This, however, assumes that the two distributions have the same variance. More formally, if the two subdistributions do not have the same variance, then for their sum to be bimodal, the following must hold²⁶:

$$2^{\frac{1}{2}} \frac{|\mu_1 - \mu_2|}{\sqrt{(\sigma_1^2 + \sigma_2^2)}} > 2$$

2.1. Histograms can deceive

Consider this histogram of sepal widths for the Iris species *versicolor*, taken from the Wikipedia page on "normal distribution"²⁷:

The data has two peaks, but the data is considered to be



sampled from a normal distribution. If we were to try and model this data as the mixture of two normal distributions, the two subdistributions would be too close together to produce two distinct peaks. The simplest way to model this data is as a normal distribution, especially as this is consistent with biological theory.

Remember that what we see in a histogram is a result of how we select the bins. It is possible to bin this data in a way that does not have two 'peaks' (for example, by using larger bin intervals, or shifting the bin boundaries). With grade distributions, ceiling effects are common: if you take normally distributed data, and then lower the values above 100% down to 100%, you may wind up seeing a second "peak" in your histogram's top bin. For an illustration, see distribution 6 in Figure 1.

3. STUDY 1: GRADES ANALYSIS

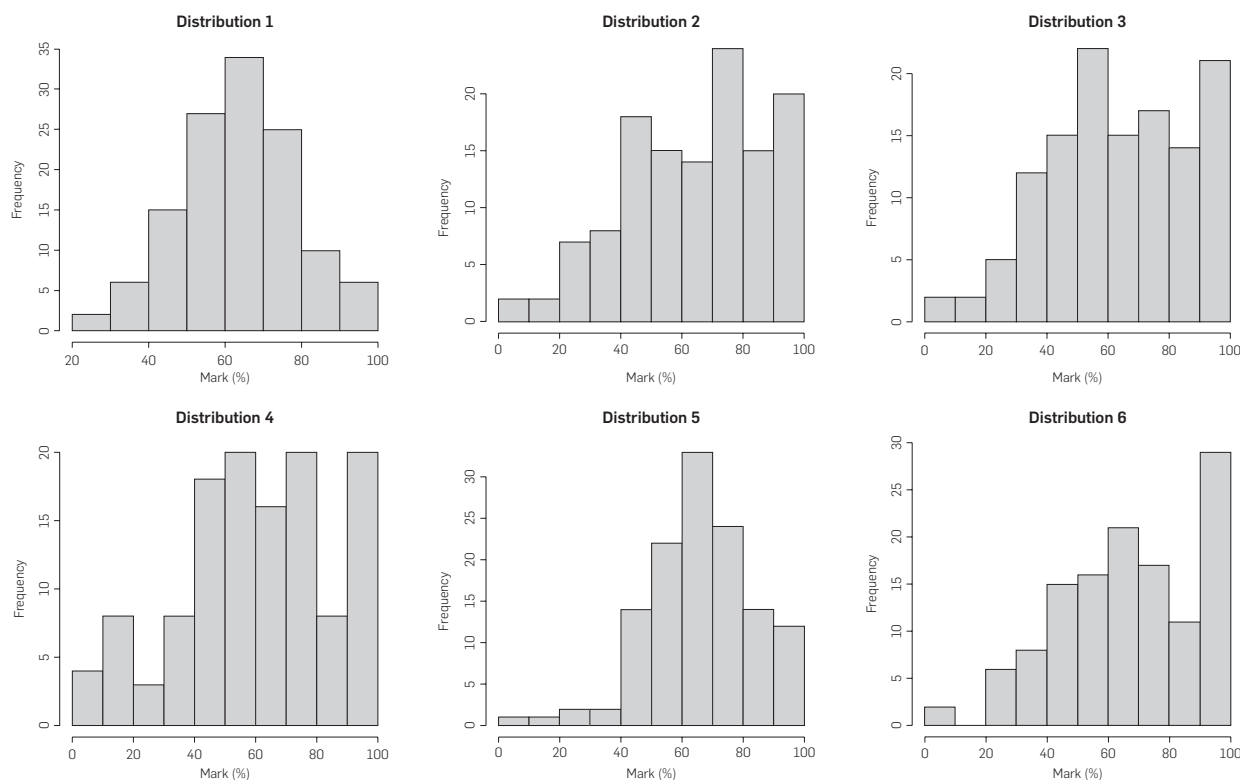
Are CS grades bimodal, or unimodal? To test this, we acquired the final grade distributions for every undergraduate CS class at the University of British Columbia (UBC), from 1996 to 2013. This represents 778 different lecture sections, containing 30,214 final grades (average class size: 75). We analyzed this data to see what distribution(s) it may have most likely come from. Frequentist null-hypothesis testing is the standard in computer science education research; for readers who are unfamiliar interpreting p values from null-hypothesis tests, we recommend consulting Goodman.⁴

3.1. Testing for multimodality

We began by computing the kurtosis for each class. *Kurtosis* is a measure of how 'tailed' the data is: high kurtosis means a distribution has a sharp peak and short tails, whereas low kurtosis implies low peak(s) and long tails.

If you look back at the illustration of adding two normal

Figure 1. The six histograms shown to participants, all of which were generated using GNU R's `rnorm` function. A ceiling of 100% was used, which is most evident in Distribution 6. Each generated distribution had 100 points, and was generated with an average of 60 and standard deviation of 5 and displayed as a histogram with bins of size 10.



distributions together, for the bimodal example, the distribution winds up being rather spread out horizontally. That distribution has low kurtosis. Indeed, for a distribution to be spread out far enough horizontally to allow for multimodality, it necessarily will have low kurtosis.

The normal distribution has a kurtosis of three. A distribution with a kurtosis greater than three cannot be bimodal.²⁶ We found that 323 of the 778 classes had a kurtosis less than 3. This means that 455 (58%) of the classes were not bimodal, and that *at most* 323 (42%) classes could be bimodal.

Hartigan's Dip Test. Hartigan's Dip Test is a test for testing whether data is multimodal (bimodal, trimodal, etc.). It looks at whether there is a "dip" in between the possible means and how deep the "dip" is (essentially: whether there is a concave-up section in the distribution). We applied Hartigan's Dip Test to the 323 classes that had a kurtosis less than 3. We chose to apply the test only to these 323 classes rather than the full 778 set in order to reduce the likelihood of false positives. For Hartigan's Dip Test, the null hypothesis is that the population is unimodal. As such, **our null hypothesis for each of the 323 tests was that a given class is unimodal.**

Results of the Dip Test. Of the 323 classes that had a kurtosis below three, 45 classes yielded a p value from Hartigan's Dip Test that was below our α value of 0.05. This is 13.9% of all the classes on which we ran Hartigan's Dip Test, or 5.8% of all the classes in our data set.

We chose the standard α value of 0.05. This means that *if* the null hypothesis is true (unimodal), the chance of a false null hypothesis rejection is 5%.⁴ If the null hypothesis is false (multimodal), the chance of a false null hypothesis rejection is 0%, because we cannot falsely reject a false null hypothesis.⁴ Until it is known whether the null hypothesis is true or not, the chance of a false positive lies between 0% and 5%.^{a4}

It could be the case that all 45 classes where we can reject the null hypothesis are indeed multimodal. But also given the noisiness of grading, ceiling effects, and small sample sizes, it could still be the case that all of these 45 classes are indeed unimodal.⁴

Although we cannot give conclusive determinations on a given null hypothesis test, the results here do provide information. Even in the unlikely case that all 45 of these classes are indeed multimodal, we see that multimodal distributions are far from being typical.^b

^a To give the reader a sense of the reliability of Hartigan's Dip Test, we generated 100,000 distributions with R's `rnorm` with $n=100$, $\mu=60$, and $\sigma=5$. A total of 133 distributions (1.3%) were tested as multimodal per Dip Test. This gives us some indication that false positives will occur with the test, but likely less than 5.8%.

^b Many people have asked whether first-year classes are more likely to be multimodal than upper-level classes. Given how few classes passed the test of multimodality, we do not have sufficient data one way or the other to properly evaluate this. More data and replication at other universities would be needed to properly test if multimodal distributions occur more often in lower-level courses.

3.2. Testing for normality

A variety of null hypothesis tests, such as Anderson-Darling, Shapiro-Wilk, and Pearson's chi-squared test determine whether a dataset is normal. We chose Shapiro-Wilk, because it has been found to have the highest statistical power.²¹

Shapiro-Wilk test. For the Shapiro-Wilk test, **the null hypothesis is that the population is normally distributed.** So, if $p < \alpha$, we can reject the null hypothesis and have evidence that the population is not normally distributed. We could reject the null hypothesis for 106 classes. This indicates that 13.6% of the classes in the data set are *not* normally distributed. As with the results of Hartigan's Dip Test, this does not mean that the null hypothesis is necessarily false in these cases. There are many reasons a distribution could not be normal: for example, it could be too skewed, it could be the wrong shape (e.g., triangular and uniform), or it could be multimodal.

It is worth noting that of the 45 classes where we rejected the null hypothesis that they were unimodal, for 44 of these classes we also rejected the null hypothesis that they were not-normal. As such, 44 of the 106 (41.5%) of the classes that were tested as being not-normal were also tested as being multimodal.

For the 86.4% of classes where we failed to reject the null hypothesis, we cannot guarantee that they are actually normal (type II error). To give an estimate of how many are actually normal, we bootstrapped a likely beta value. This yielded an estimated false negative rate of 1.48%.

From our data, we estimate that 85.1% of the final grades in UBC's CS classes are normally distributed. This indicates that grades from a computer science class are typically normal—not bimodal.

Skewness. Although most of the distributions appear to be normally distributed, it is worth noting that the average skewness of all the distributions is -0.33 , whereas a normal distribution should have a skewness of zero. If we only consider the distributions whose test results indicated normality, the average skewness is -0.13 . This provides some sanity checking on our normality testing: the “normal” distributions are not particularly skewed. For the classes where we rejected the null hypothesis of normality (i.e., probably not normal), the average skewness was higher. Likely, this is why many of these classes were indicated by Shapiro-Wilk as not normal. Higher skewness could also be a result of the ceiling effect in grade distributions.

3.3. Discussion

We only examined final grades: our analysis did not include term grades. And as grades only came from one institution, one may wonder about generalizability. We tried to acquire grade distributions from other institutions, but generally found it difficult to gather the same scale of data. What stood out for us is that our colleagues (both at UBC and elsewhere) would routinely assert that their CS grades are bimodal, and our analysis gives evidence to the contrary. Although we cannot assert from this analysis that every university has the same distributions as UBC, the large scale of data both in numbers and time-span is compelling. Our

interpretation is also not alone: our results support Lister's argument that CS grades are generally not bimodal.

We invite readers to replicate our findings at other institutions.^c The code to replicate the analysis is available online at <https://github.com/patitsas/bimodality>.

4. STUDY 2: HUMAN INTERPRETATION OF DISTRIBUTIONS

So if CS grades are rarely bimodal, why does the belief in bimodality persist? An insight came one day when generating some random normal distributions in R: with only 100 data points, the resulting histogram often had more than one peak and could be easily erroneously perceived as “bimodal”. A typical “large class” does not have a large enough sample size to consistently provide a smooth curve. Indeed, many of the distributions produced by R's `rnorm` looked very much like the grade distributions we had seen in our own classes and called “bimodal.”

Interested in whether instructor perceptions affect the interpretation of noisy distributions, we designed an experiment wherein participants are presented with histograms of distributions produced by R's `rnorm` function, and asked to categorize the distribution (normal, bimodal, uniform, etc.). We initially had two research questions:

1. Do CS instructors who believe in innate ability categorize more noisy distributions as bimodal?
2. If we prime participants that CS distributions are commonly thought to be bimodal, are they then more likely to see bimodal distributions in the noise?

Once we analyzed our data for those two research questions, a third research question arose:

3. If instructors label noisy distributions as bimodal, are they more likely to agree with the idea of innate CS ability? (i.e., is there a possible feedback loop between looking at distributions and instructors' beliefs?)

4.1. Experimental design

A difficulty in studies looking at priming effects is that you cannot state the purpose of the study in the consent form. If you do, then you are priming participants, even the participants you want in your control group. To disguise our study, we presented it as one asking people how often they saw various distribution shapes in their own classes.

We presented each participant with the six histograms as shown in Figure 1, all of which we generated using R's `rnorm` function. We generated a few dozen histograms and selected the six histograms from that pool: one to be clearly normal (distribution 1), one that was mildly skewed as though students who were failing were pushed up to 50% (distribution 5), one where the ceiling effect was visible (distribution 6), and three noisy distributions which had multiple peaks (distributions 2–4).

We asked each participant whether they saw this shape of

^c Since the original ICER publication, our findings have been replicated at a university in the United States.²

distribution in their own classes (“very often” to “never” on a Likert scale), and how they would categorize the distribution (normal, bimodal, multimodal, uniform, and others). We randomly assigned participants to one of two treatments:

Treatment 0: participants were asked whether they agreed that CS ability is innate, then asked to categorize the distributions, and were not being primed to think about bimodality.

Treatment 1: participants were primed to think about the common-held belief about CS grade distributions, before they saw the distributions; after that, we asked whether they agreed that CS ability is innate.

The survey’s five pages are described in Table 1. For each question, we created a shorthand label, shown in sans-serif, for use in our analysis.

Because so many of the potential participants were our colleagues, we deliberately did not collect names and identify information about participants. We did not want to know who was or was not a participant, nor how they responded to the survey.

As a courtesy, we offered participants the option of having their email recorded on a separate platform if they wanted us to follow up with them about the results of the study.

We did not look at this email list until after our analysis was complete.

4.2. Participants

We recruited 60 CS instructors, mostly from the SIGCSE members’ list. Some participants were recruited from other online CS education communities, and some were recruited at ICER 2015. Fifty-three participants completed every question on the survey; twenty-eight were in Treatment 0 (the nonprimed group), and twenty-five were in Treatment 1 (the

primed group). The participants who had provided their emails for follow-up purposes were debriefed. As fewer than half of the participants had provided their email, we posted open debriefing statements to the online communities where we had recruited participants.

4.3. Results

For each participant, we computed a value we call “seeing-bimodality,” which is how many of the six distributions the participant had categorized as bimodal or multimodal. In our data, seeing-bimodality ranged from 0 to 5.

Regression on seeing-bimodality. We wanted to see if seeing-bimodality could be predicted by participants’ responses to our questions. The regression we performed was to model seeing-bimodality as a function of innately-predisposed, all-succeed, look-histo, and look-letter (shorthand names from Table 1).

When visualizing the results, we noticed that the relationship between seeing-bimodality and the Likert questions varied between the two treatments. As a nonparametric equivalent of ANCOVA, we performed an ordinal logistic regression on the two treatments separately using the `polr` function from R’s `MASS` library, and then used the `Anova` function from the `car` package to compare the two. This allowed us not only to test whether there were relationships between seeing-bimodality and the Likert questions, but to see if these relationships were different for the two treatments. This approach required computing 28 p values. To reduce the chance of false positives from using multiple statistical tests, we applied a Šidák correction, which reduced our α level to 0.002 for this section of our analysis.

In both our regressions on Treatment 0 and on Treatment 1, we found a significant relationship between seeing-bimodality and participants’ responses to the questions related to innate ability (all-succeed and innately predisposed).^d

We then looked to see if this relationship was stronger in one treatment than the other. In both questions about innate ability, the effect was significantly stronger in the treatment where subjects were primed to think about CS grades being bimodal, as shown in Table 2.

Both regressions also revealed a statistically significant relationship between seeing-bimodality and how often participants reported looking at histograms of their grades (look-histo). This relationship was not statistically significantly different between the two treatment groups.

Perhaps unsurprisingly, there was a strong negative

^d Regression tables are provided in the original ICER publication, and are omitted due to page limitations.

Table 1. The pages of the survey.

1. Questions about how large their typical class was (“class-size”) and how long they had been teaching (“years-experience”).
2. A priming question: ‘It is a commonly held belief that CS grade distributions are bimodal. Do you find this to be the case in your teaching?’ (“have-bimodal”)
3. Questions on how often they look at their grade distributions:
 - ‘When teaching, how often do you look at histograms of your students’ grades? (This applies both to term work and final grades.)’ (“look-histo”)
 - ‘How often do you look at how many students fall into each letter category (A, B, etc.)? (This applies both to term work and final grades.)’ (“look-letter”)
4. Six histograms, all generated with GNU R’s `rnorm`, shown in Figure 1. For each histogram, we asked two questions:
 - ‘How often do you see the shape of [this distribution] in your classes?’
 - ‘What sort of distribution would you describe [this distribution] as?’
5. Likert-style questions on innate ability (5 points, Strongly Agree to Strongly Disagree):
 - Nearly everyone is capable of succeeding in computer science if they work at it. (“all-succeed”)
 - Some students are innately predisposed to do better at CS than others. (“innately-predisposed”)

Pages 2 and 5 were swapped for a random half of the participants. We chose the all-succeed question because it had been used previously in the literature.

Table 2. Results of the Anova of the regressions on the two treatments; that is, does the relationship between a given factor and seeing-bimodality differ between the two treatments?

	LR Chisq	Df	Signif?
innately-predisposed	11.0	2	yes
all-succeed	14.8	3	yes
look-histo	4.1	4	no
look-letter	6.1	4	no

correlation between all-succeed and innately predisposed. Those who felt there was an innate predisposition to do well in CS also felt that not everyone could succeed in the field.

Regression on all-succeed. After finding a one-way relationship between grade perceptions and the innateness belief, we wanted to see if there was any evidence of a feedback loop between the two. Because all-succeed and innately predisposed correlated so highly, we found they were interchangeable as measures of belief in innate ability. As logistic regression involves only one dependent variable, we had to pick one of the two to use. We chose to do this analysis with all-succeed because the question item had been used in another study.¹⁴

Recall that our study was set up so that a random half of the participants categorized distributions and then were asked about innate ability (Treatment 1), whereas the other half were asked about innate ability and then categorized the distributions (Treatment 0). If there is a feedback loop here, we would expect that seeing-bimodality would predict all-succeed in Treatment 1, but not in Treatment 0.

Guidelines for statistical power in logistic regression suggest that an α level of 0.05 requires 10–20 data points per independent variable in your model.¹⁶ Because this part of the analysis requires the statistical power to reject a null hypothesis, we modeled all-succeed as only a function of seeing-bimodality, and set $\alpha = 0.05$.

For Treatment 1, we found that seeing-bimodality was a statistically significant predictor of all-succeed. In Treatment 0, it was not. This indicates that there is a feedback loop between categorizing distributions as bimodal and agreement with the idea of innate ability. We hence have observed evidence for the relationships illustrated in **Figures 2 and 3**.

4.4. Discussion

With regard to the feedback loop between seeing-bimodality and all-succeed, we have some weak evidence that categorizing distributions as bimodal increases belief in the

Figure 2. Individual-level feedback loop leading individuals to categorize ambiguous distributions as bimodal.

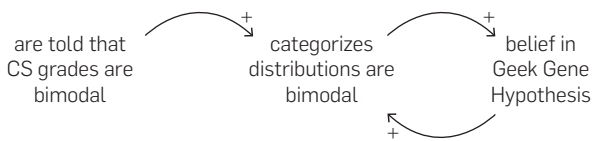
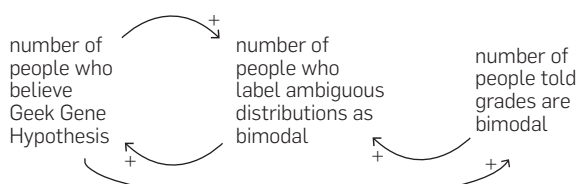


Figure 3. Social-level feedback loops leading individuals to categorize ambiguous distributions as bimodal.



Geek Gene Hypothesis. We consider our evidence weak because our study was underpowered, and caution should be taken in interpreting the lack of significance in the second treatment.

We were initially surprised that regularly looking at histograms of grades was associated with a higher score for seeing-bimodality. This led us to add our third research question, based on the idea that it could be that the more often you look at your grades, the more it solidifies your conception of what your grades are like. This supports our observation that categorizing distributions as bimodal increases belief in innate ability. System justification theory explains how once you are forced to take a position on a subject, you are more likely to believe and defend it.¹¹ Our approach to priming—stating that it is a commonly held belief that CS grade distributions are bimodal—may have strengthened our participants’ beliefs about the bimodality of CS grades. Because the survey presents us, the researchers, as authority figures, and we imply that grades may be bimodal, some participants could assume it to be true because of our endorsement.

When we piloted our survey, some participants opined that they believed that some students were predisposed because of prior experience, rather than inherent brilliance.

We did not have a representative sample of CS educators. The educators who participate in CS education communities are generally much more invested in their teaching than their peers who do not. Furthermore, some of our participants may be familiar with Ahadi and Lister¹, which could have influenced their responses. But we would expect the SIGCSE community to be less inclined to believe in innate ability than their non-SIGCSE peers. We still had enough participants who agreed with the hypothesis for us to conduct our analysis. Future work is needed to replicate our findings with a more representative sample of CS educators.

Supporting literature. Our findings agree with the psychology literature: people’s biases affect their decision-making more when they are judging more ambiguous information.¹⁰ For example, Heilman et al. found that resumes of extremely qualified candidates were likely to be judged worthy of a salary increase regardless of the gender listed on the resume—but for resumes of ambiguously qualified candidates, resumes with male names were more likely to be viewed positively than those with female names.¹⁰ Eyesnck et al. studied the interpretation of written sentences as either threatening or nonthreatening by people who have anxiety and by a control group.³ They found that unambiguously threatening or nonthreatening sentences were interpreted similarly between groups, but participants with anxiety were more likely than controls to label ambiguous sentences as threatening. Visual information is also subject to this phenomenon: Payne et al. showed participants a series of photos of people holding either guns or ambiguous objects, and participants were more likely to identify the ambiguous object as a gun if it was held by a black person.¹⁹

Furthermore, belief can affect judgment regardless of ambiguity. For example, Kahan et al. found that participants were more likely to get a math problem incorrect if the correct result would disagree with their political beliefs.¹² It is hence plausible that a computer scientist who believes in the Geek Gene Hypothesis could look at an unambiguously unimodal distribution and still view it as bimodal.

5. THE GEEK GENE HYPOTHESIS AS A SOCIAL DEFENSE

Once again, our findings support Lister's hypothesis that CS grades are generally not bimodal and this perception stems from instructors expecting to find bimodal grades due to a belief in the Geek Gene Hypothesis. We now go a step further and argue that the perception of bimodality is a *social defense* in the CS education community.

In sociology and social psychology, a "social defense is a set of organizational arrangements, including structures, work routines, and narratives, that functions to protect members from having to confront disturbing emotions stemming from internal psychological conflicts produced by the nature of the work".¹⁷

5.1. Social defenses in teaching

Guzdial reports that teachers generally have a high level of self-efficacy (great confidence in their teaching ability) at the start of their career. This then plummets as they face the realities of classroom teaching but slowly returns with time.⁹ Teacher self-efficacy is not necessarily tied to teaching ability: university educators often get little meaningful feedback on how their students are learning, given their large class sizes and lecture-based pedagogies.⁹

Guzdial notes that if an individual university-level CS educator has high self-efficacy, and sees evidence of students not learning, then it is rational for them to believe that the problem lies with the students and that the problem is innate to them—that is, beyond the ability of the teacher to influence.⁹ Compounding this, Sahami and Piech have observed that CS educators are more aware of their top and bottom students than they are of their average students, giving educators a biased perception of their students' abilities.²⁴ Guzdial argues that CS educators have poor results, because we so frequently use ineffective teaching methods.⁷ Zingaro et al. suggest that not only do CS educators frequently use ineffective pedagogies, they also frequently use ineffective assessment tools.^{28, 20}

We theorize that the Geek Gene Hypothesis is a social defense: it is easier for computer science educators to blame innate qualities of their students for a lack of learning than it is for the educators to come to terms with the ineffectiveness of their teaching.

A social defense is a phenomenon on a *social* scale, in contrast to Guzdial's observation about individual teachers. When numerous educators bond over how their students just "do not have it," it allows for the Geek Gene hypothesis to go from one individual's suspicion to a social narrative. And as bimodal grade distributions sometimes do occur, those cases are used to argue that

this is a common and inherent phenomenon in CS classes. The perception of bimodal grades provides evidence to the Geek Gene narrative that some students "have it" and some do not. And when new educators who have been primed to see bimodality then begin teaching and do not see all their students learning, these new educators can then see this as evidence of the Geek Gene. The reproduction of the Geek Gene Hypothesis is hence social in nature.

5.2. The "Geek Gene" is an equity issue

Debunking the "Geek Gene" is also important for equity reasons. Recent studies have found that academic disciplines in which "brilliance" is seen as necessary for success have less gender diversity.¹³ Looking at the history of science, women and people of color were long denied entry and acknowledgment in science because they were seen as lacking the "brilliance" needed to do science.²³ If computing ability is viewed as being the result of a "Geek Gene," then educators may use this as a reason not to teach students who they perceive as lacking this "gene." Similarly, they could lower expectations of these groups and encourage them less—which is troubling given evidence that teacher expectations have an effect on student performance.²²

6. CONCLUSION

Our analysis of one institution's CS grades indicates that although bimodal grade distributions can be found, they are far from typical. Much more commonly, grade distributions are normal (85.1% of cases) or highly skewed unimodal distributions. Our psychology experiment found that participants who were more likely to label ambiguous distributions as bimodal were also more likely to report a belief in an innate ability to succeed in CS. This suggests that instructor beliefs play a role in the perception of bimodality.

Priming participants to think about the common perception of bimodal grades also led to participants being more likely to label ambiguous distributions as bimodal. This suggests that confirmation bias plays a role in the belief that bimodal grades are typical.

Given that the belief that CS ability is innate is widespread among CS educators, there is likely a social element to the confirmation bias. This belief in bimodality appears related to the belief in innate ability, which in turn has been implicated in the under-representation of women and minorities in computing. We encourage educators reading this paper to take time to analyze the grades in their own classes, and bring the same level of rigor and skepticism we would use in our research to understand our own teaching.

Acknowledgments

The first author received funding from the Social Science and Humanities Research Council of Canada. We would also like to thank our anonymous reviews, Aditya Bhargava, Jinghui Cheng, Jeff Forbes, Jin Guo, Mark Guzdial, Ray Lister, Andrew Petersen, Greg Wilson, and Dan Zingaro for their feedback and suggestions on this line of investigation. 

References

1. Ahadi, A., Lister, R. Geek genes, prior knowledge, stumbling points and learning edge momentum: parts of the one elephant? In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, ACM, 2013, 123–128.
2. Basnet, R.B., Payne, L.K., Doleck, T., Lemay, D.J., Bazalais, P. Exploring bimodality in introductory computer science performance distributions. *EURASIA J. Math. Sci. Technol.*, 14 (2018), 10.
3. Eysenck, M.W., Mogg, K., May, J., Richards, A., Mathews, A. Bias in interpretation of ambiguous sentences related to threat in anxiety. *J. Abnorm. Psychol.* 2, 100 (1991), 144.
4. Goodman, S. A dirty dozen: twelve p-value misconceptions. In *Seminars in Hematology*, Volume 45. Elsevier, 2008, 135–140.
5. Gould, S.J. *The Mismeasure of Man*. WW Norton & Company, 1996.
6. Guzdial, M. Anyone can learn programming: Teaching > genetics, 2014.
7. Guzdial, M. Teaching computer science better to get better results, 2014.
8. Guzdial, M. Learner-centered design of computing education: Research on computing for everyone. *Synth. Lect. Hum. Cent. Inform.* 6, 8 (2015), 1–165.
9. Guzdial, M. Source of the "geek gene"? *Teacher beliefs: Reading on Lijun Ni, learning from Helenrose Fives on teacher self-efficacy*, 2015.
10. Heilman, M.E., Black, C.J., Stathatos, P. The affirmative action stigma of incompetence: Effects of performance information ambiguity. *Acad. Mgmt. J.* 3, 40 (1997), 603–625.
11. Jost, J.T., Banaji, M.R., Nosek, B.A. A decade of system justification theory: Accumulated evidence of conscious and unconscious bolstering of the status quo. *Polit. Psychol.* 6, 25 (2004), 881–919.
12. Kahan, D.M., Peters, E., Dawson, E.C., Slovic, P. Motivated numeracy and enlightened self-government. *Yale Law School, Public Law Working Paper*, (307), 2013.
13. Leslie, S.-J., Cimpian, A., Meyer, M., Freeland, E. Expectations of brilliance underlie gender distributions across academic disciplines. *Science* 6219, 347 (2015), 262–265.
14. Lewis, C. Attitudes and beliefs about computer science among students and faculty. *SIGCSE Bull.* 2, 39 (2007), 37–41.
15. Lister, R. Computing education research geek genes and bimodal grades. *ACM Inroads* 3, 1 (2010), 16–17.
16. McDonald, J.H. *Handbook of Biological Statistics*, Volume 2. Sparky House Publishing, Baltimore, MD, 2009.
17. Padavic, I., Ely, R.J. The work-family narrative as a social defense, 2013.
18. Park, T.H., Saxena, A., Jagannath, S., Wiedenbeck, S., Forte, A. Towards a taxonomy of errors in HTML and CSS. In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, ACM, 2013, 75–82.
19. Payne, B.K., Shimizu, Y., Jacoby, L.L. Mental control and visual illusions: Toward explaining race-biased weapon misidentifications. *J. Exp. Soc. Psychol.* 1, 41 (2005), 36–47.
20. Petersen, A., Craig, M., Zingaro, D. Reviewing CS1 exam question content. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, SIGCSE'11 (New York, NY, USA, 2011). ACM, 631–636.
21. Razali, N.M., Wah, Y.B. Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. *J. Stat. Model. Anal.* 1, 2 (2011), 21–33.
22. Rosenthal, R. Teacher expectancy effects: A brief update 25 years after the pygmalion experiment. *J. Res. Educ.* 1, 1 (1991), 3–12.
23. Rossiter, M.W. *Women Scientists in America: Struggles and Strategies to 1940*, Volume 1. JHU Press, 1982.
24. Sahami, M., Piech, C. As CS enrollments grow, are we attracting weaker students? In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE'16 (New York, NY, USA), 2016. ACM, 54–59.
25. Schilling, M.F., Watkins, A.E., Watkins, W. Is human height bimodal? *Am. Stat.* 3, 56 (2002), 223–229.
26. Wikipedia. Multimodal distribution—wikipedia, the free encyclopedia, 2016 [online; accessed 6-April-2016].
27. Wikipedia. Normal distribution—wikipedia, the free encyclopedia, 2016 [online; accessed 6-April-2016].
28. Zingaro, D., Petersen, A., Craig, M. Stepping up to integrative questions on cs1 exams. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*. ACM, 2012, 253–258.

Elizabeth Patitsas (elizabeth.patitsas@mcgill.ca), McGill University Montreal, Québec, Canada.

Michelle Craig and Steve Easterbrook ([mcraig.sme]@cs.toronto.edu), University of Toronto Toronto, Ontario, Canada.


Jesse Berlin (jesse.berlin1@gmail.com), Toronto, Ontario, Canada.

© 2020 ACM 0001-0782/20/1 \$15.00

ACM Transactions on Computing for Healthcare (HEALTH)


Open for Submissions

A multidisciplinary journal for high-quality original work
on how computing is improving healthcare



ACM Transactions on Computing for Healthcare (HEALTH) is a multidisciplinary journal for the publication of high-quality original research papers, survey papers, and challenge papers that have scientific and technological results pertaining to how computing is improving healthcare.

For further information and to submit
your manuscript, visit health.acm.org



Boston College

Non Tenure-Track Positions in Computer Science

The Computer Science Department of Boston College seeks to fill one or more non-tenure-track teaching positions, as well as shorter-term visiting teaching positions. All applicants should be committed to excellence in undergraduate education and be able to teach a broad variety of undergraduate computer science courses. Faculty in longer-term positions will also participate in the development of new courses that reflect the evolving landscape of the discipline.

Minimum requirements for the title of Assistant Professor of the Practice, and for the title of Visiting Assistant Professor, include a Ph.D. in Computer Science or closely related discipline.

Candidates without a Ph.D. will be eligible for the title of Lecturer or Visiting Lecturer.

We will begin reviewing applications on October 15, 2019 and will continue considering applications until the positions are filled. Applicants should submit a cover letter, CV, and a separate teaching statement and arrange for three confidential letters of recommendation that comment on their teaching performance to be uploaded directly to Interfolio. To apply go to <https://apply.interfolio.com/68339>. Boston College conducts background checks as part of the hiring process. Information about the university and our department is available at <https://www.bc.edu> and <http://cs.bc.edu>.

Boston College is a Jesuit, Catholic university that strives to integrate research excellence with a foundational commitment to formative liberal arts education. We encourage applications from candidates who are committed to fostering a diverse and inclusive academic community. Boston College is an affirmative action/equal opportunity employer.

Boston College

Tenure-Track Assistant Professor of Computer Science

The Computer Science Department of Boston College seeks a tenure-track Assistant Professor beginning in the 2020-2021 academic year. Successful candidates for the position will be expected to develop strong research programs that can attract external funding in an environment that also values high-quality undergraduate teaching. Outstanding candidates in all areas of Computer Science will be considered, with a preference for those who demonstrate a potential to contribute to cross-disciplinary teaching and research in conjunction with the planned Schiller Institute for Integrated Science and Society at Boston College.

A Ph.D. in Computer Science or a closely related discipline is required. See <http://cs.bc.edu> and <https://www.bc.edu/bc-web/schools/mcas/sites/schiller-institute.html> for more informa-

tion. Application review is ongoing. Boston College conducts background checks as part of the hiring process.

Submit a cover letter, a detailed CV and teaching and research statements. Arrange for three confidential letters of recommendation to be uploaded directly to Interfolio. To apply go to <https://apply.interfolio.com/68273>.

Boston College is a Jesuit, Catholic university that strives to integrate research excellence with a foundational commitment to formative liberal arts education. We encourage applications from candidates who are committed to fostering a diverse and inclusive academic community. Boston College is an affirmative action/equal opportunity employer.

Bowling Green State University

Assistant Professor – Computer Science

Department of Computer Science: Assistant Professor in Data Science, Bowling Green State University. Tenure-track faculty position available August 2020.

Responsibilities: The successful candidate will develop a productive research program supported by external funding, teach effectively at the undergraduate and graduate levels, advise undergraduate and graduate students, and provide service to the department, university, and profession.

Minimum Qualifications: A Ph.D. or equivalent in computer science or related discipline (e.g., Data Science, Information Science, Engineering). ABDs will be considered as long as the requirements for the degree are completed by August 2020; research experience in a data science related area, including but not limited to machine learning, deep learning, natural language processing, data mining, data quality, and data analysis; demonstrated record of teaching effectiveness, such as teaching records, student evaluations, and/or use of instructional tools and technologies; demonstrated record of research productivity, with at least one or more peer reviewed journal/conference publications that have been accepted Preferred Qualifications: established background in data science as evidenced by peer-reviewed publications, reports, and/or scientific presentations; teaching experience at the college level (course instructor, teaching assistant, teaching practicum), particularly in the areas of data science

Credentials Required for Application: Cover letter, detailing data science related teaching and research experience; curriculum vitae; statement of teaching philosophy and evidence of teaching effectiveness; statement of research interests and agenda; statement of contributions or commitment to diversity in higher education; one or more pdf files of in-press manuscripts or publications within the last 3 years; contact information for three professional references, letter of refer-

ence must be dated within the past year; and copies of transcripts indicating all graduate course work.

For a complete job description & instructions on how to apply for this position visit <https://bgsu.hiretouch.com/> or contact the Office of Human Resources, BGSU. Application deadline is January 20, 2020. Background check and official transcripts indicating highest degree earned are required for employment. BGSU is an AA/EEO/Vet employer. We encourage applications from women, minorities, veterans, and persons with disabilities regardless of age, gender identity, genetic information, religion, or sexual orientation.

Rutgers University

Tenure-Track Position in Computer Science

The Computer Science Department at Rutgers University invites applications for a tenure-track position. We are especially interested in hiring at the Assistant Professor level in the area of Cybersecurity, but invite applications in all areas of Computer Science at all levels. We take a broad view of security within Computer Science, including formal and algorithmic approaches to secure computation and communication; empirical approaches to the security of deployed software and systems; and research on the security of data, such as privacy, authenticity and digital forensics; as well as interdisciplinary security research linking innovative Computer Science to its implications for regulation, public policy or business strategy.

The appointment will start September 1, 2020. Responsibilities include research, supervision of PhD students, and teaching undergraduate and graduate level courses in Computer Science. Pursuit of external research funding is expected.

Qualifications

Successful completion of a PhD or equivalent in Computer Science or a closely related field is required by the start date.

Application Instructions

Applicants should submit their CV, a research statement addressing both past and future work, a teaching statement, and contact information for three references at <http://jobs.rutgers.edu/postings/104678>. Applications received by January 15, 2020 will be given priority.

For questions, contact: santosh.nagarakatte@cs.rutgers.edu.

The CS Department is strongly committed to increasing the diversity of our faculty and welcomes applications from women, dual-career couples, historically underrepresented populations and candidates with disabilities. Offer is contingent upon successful completion of all pre-employment screenings. Rutgers is an affirmative action/equal opportunity employer.

Southern University of Science and Technology (SUSTech)

Professor/Associate Professor/Assistant Professor

The Department of Computer Science and Engineering (CSE, <http://cse.sustc.edu.cn/en/>), Southern University of Science and Technology (SUSTech) has multiple Tenure-track faculty openings at all ranks, including Professor/Associate Professor/Assistant Professor. We are looking for outstanding candidates with demonstrated research achievements and keen interest in teaching, in the following areas (but are not restricted to):

- ▶ Data Science
- ▶ Artificial Intelligence
- ▶ Computer Systems (including Networks, Cloud Computing, IoT, Software Engineering, etc.)
- ▶ Cognitive Robotics and Autonomous Systems
- ▶ Cybersecurity (including Cryptography)

Applicants should have an earned Ph.D. degree and demonstrated achievements in both research and teaching. The teaching language at SUSTech is bilingual, either English or Putonghua. It is perfectly acceptable to use English in all lectures, assignments, exams. In fact, our existing faculty members include several non-Chinese speaking professors.

As a State-level innovative city, Shenzhen has identified innovation as the key strategy for its development. It is home to some of China's most successful high-tech companies, such as Huawei and Tencent. SUSTech considers entre-

preneurship as one of the main directions of the university. Strong supports will be provided to possible new initiatives. SUSTech encourages candidates with experience in entrepreneurship to apply.

SUSTech is a pioneer in higher education reform in China. The mission of the University is to become a globally recognized research university which emphasizes academic excellence and promotes innovation, creativity and entrepreneurship. Set on five hundred acres of wooded landscape in the picturesque Nanshan (South Mountain) area, the campus offers an ideal environment for learning and research.

SUSTech is committed to increase the diversity of its faculty, and has a range of family-friendly policies in place. The university offers competitive salaries and fringe benefits including medical insurance, retirement and housing subsidy, which are among the best in China. Salary and rank will commensurate with qualifications and experience. More information can be found at <http://talent.sustc.edu.cn/en>.

We provide some of the best start-up packages in the sector to our faculty members, including one PhD studentship per year, in addition to a significant amount of start-up funding (which can be used to fund additional PhD students and postdocs, research travels, and research equipments).

To apply, please provide a cover letter identifying the primary area of research, curriculum vitae, and research and teaching statements, and forward them to eshire@sustc.edu.cn.

University of Macau

Assistant/Associate Professor in Computer and Information Science

Ref. No.: FST/CIS/AAP/11/2019

The Department of Computer and Information Science (CIS) of the University of Macau (UM) invites applications for the position of Assistant / Associate Professor in Cloud Computing and Distributed and Parallel Computing. We are seeking candidate with a proven track record in research and education.

CIS offers Bachelor's, Master's and Doctoral degrees programmes. Currently, it has 25 academic staff, around 220 undergraduate students and 215 postgraduate students. UM is the only public comprehensive university in Macao, English is its working language. UM is among the top 1% in ESI rankings in both Engineering and Computer Science. In the THE World University Rankings, the Computer Science programme is ranked among the top 200.

The candidates must have an earned PhD degree in related areas. Preference will be given to candidates with specialties in cloud computing and parallel and distributed system support for big data and artificial intelligence.

Applicants should visit <https://career.admo.um.edu.mo/> for more details and apply **ONLINE**.

University of Maryland, Baltimore County (UMBC)

Multiple Open Rank Tenure-Track Faculty Positions in the Department of Information Systems

The Department of Information Systems (IS) at UMBC invites applications for three open rank tenure-track faculty positions starting August 2020. Successful candidates will complement and extend our current strengths in AI, Data Science, Human-Centered Computing, Health IT, and Software Engineering. Candidates with research interests cross-cutting multiple areas are particularly encouraged to apply. Strong candidates with a research emphasis in other areas may also be considered. Candidates must have earned a PhD in Information Systems, Computer Science, or a related field by August 2020.

All candidates are expected to establish a collaborative, externally funded, and nationally recognized research program and to contribute to teaching both graduate and undergraduate courses effectively. Innovation is expected in pedagogical methods, course content, and curriculum development as well as in advising and mentoring of a diverse student body. Candidates for Associate Professor rank should also have a strong record of research, teaching, service, and a sustained externally-funded research program. Candidates for Full Professor rank should also demonstrate leadership in their field, hold an excellent academic record, and show a history of securing external funds for multiple sizable research projects.

The department offers undergraduate degrees in Information Systems and Business Technology Administration and MS and PhD degrees in both Information Systems and Human-Centered Computing, including an innovative online MS program in IS and a professional masters' program in Health IT. Our faculty are actively en-



Australian National University

OPEN RANK TENURE TRACK – MULTIPLE COMPUTER SCIENCE FACULTY POSITIONS

Australian National University | Research School of Computer Science

The ANU College of Engineering and Computer Science (CECS) is undergoing significant change and expansion through the Reimagine investment. This substantial 15-year, commitment will transform traditional engineering and computer science disciplines for the 21st century.

This is an exciting time to join our Faculty and be part of a community that prides itself on delivering cutting-edge research and research-led education to develop future leaders, who will find solutions to some of the world's greatest technological and social challenges.

To enquire about these positions please contact Computer Science Director, Professor Tony Hosking, via email to director.rscs@anu.edu.au

Come and enjoy the fantastic Australian lifestyle, while working for a world leading University with outstanding staff benefits, including;

- 17% superannuation
- 26 weeks paid parental leave
- 4 weeks paid annual leave
- Exceptional Professional Development opportunities

www.anu.edu.au/jobs





INDIANA UNIVERSITY

Luddy School of Informatics, Computing, and Engineering

Chair of Computer Science

The Luddy School of Informatics, Computing, and Engineering at Indiana University Bloomington invites applications for chair of computer science, a tenured full professor position to begin in Fall 2020. This position will be appointed as Luddy endowed faculty. The chair will enter the School at a transformative period of substantial new resources. A recent \$60 million gift to the School provides funding for significant faculty and student endowments as well as a state-of-the-art artificial intelligence facility. The Luddy School aspires to be the leading school of computing in the world.

The Luddy School seeks a dynamic individual with an international reputation and a thirst for excellence to lead the school's computing efforts. The chair will have significant ability to shape the growth of computer science in the School, including through several new tenure-track faculty lines.

Candidates from all areas of computer science are encouraged to apply. Applicants should have a world-class research record, a strategic vision for excellence in computer science research and education, and the academic and research leadership skills to advance that vision. They should also have an established record for excellence and a Ph.D. in computer science or another relevant area.

The School seeks candidates prepared to contribute to our commitment to diversity and inclusion in higher education. The strongest candidates can demonstrate their experience in working with diverse student populations. Women and minorities are encouraged to apply. Duties will include research, teaching, and service.

The Luddy School was the first of its kind and is among the largest in the country. Its mission is to excel and lead in education, research, and outreach spanning and integrating the full breadth of computing and information technology.

It includes computer science, informatics, library and information science, intelligent systems engineering, and data science, with 140 faculty, 1,100 graduate students, and 2,000 undergraduate majors. It offers Ph.D.s in computer science, informatics, information science, and intelligent systems engineering, and actively supports entrepreneurship.

Bloomington is a culturally thriving college town with a moderate cost of living and the amenities for an active lifestyle. IU is renowned for its top-ranked music school, high-performance computing and networking facilities, and performing and fine arts.

Candidates should review the application requirements and apply online at: indiana.peopleadmin.com/postings/8907

For full consideration, submit your application by January 10, 2020. Applications will be accepted until the position is filled. Questions may be sent to: luddyjob+cs-chair@indiana.edu

Indiana University is an equal employment and affirmative action employer, and a provider of ADA services. All qualified applicants will receive consideration for employment without regard to age, ethnicity, color, race, religion, sex, sexual orientation, gender identity or expression, genetic information, marital status, national origin, disability status or protected veteran status.



LUDDY
SCHOOL OF INFORMATICS,
COMPUTING, AND ENGINEERING

gaged in collaborative interdisciplinary research and three of our current faculty have received NSF CAREER awards.

UMBC's strategic location in the Baltimore-Washington corridor puts us close to many government agencies and high-tech companies. The 2018 Chronicle of Higher Education also named UMBC as one of the best colleges to work for, for the ninth year in a row. UMBC is a national model for diversity and inclusive excellence in STEM. We especially welcome applications from candidates who are willing to contribute to the diversity mission of the university and the department. Members of groups that historically have been underrepresented in the professoriate are especially encouraged to apply.

For full consideration, applications for the positions must be submitted as PDF files including a cover letter, CV, teaching statement (1 page), research statement (2 pages), statement of demonstrated commitment to diversity and inclusive excellence (1 page), and names of 3 references via Interfolio at <http://apply.interfolio.com/69677> by December 15, 2019.

Candidates' experience will be evaluated commensurate to the rank to which they apply. For inquiries, please email is_faculty_search_2019@umbc.edu. Review of applications will begin in December, 2019 and will continue until the positions are filled.

UMBC is an Affirmative Action/Equal Opportunity Employer and welcomes applications from minorities, women, veterans, and individuals with disabilities.

University of Tennessee

Three (3) Assistant or Associate Professor Positions in Computer Science and Computer Engineering

The Department of Electrical Engineering and Computer Science (EECS) at The University of Tennessee, Knoxville (UTK) is seeking candidates for three (3) tenure track faculty members at the assistant or associate professor level. Applicants should have an earned Ph.D. in Computer Science, Computer Engineering, or a related field by time of appointment.

The department has sustained an ambitious growth period over the past six years. This year, we are strategically targeting areas of (1) **cybersecurity**; (2) **data analytics, machine learning, and artificial intelligence**; and (3) **Internet of Things (IoT), embedded systems, edge computing, and mobile computing systems**. In all three cases, the area of interest is defined broadly. Apply at <https://academicjobsonline.org/ajo/jobs/15198> and submit a cover letter, a curriculum vitae, a statement of research and teaching interests, and contact information for a minimum of three professional references. Applicants should have a demonstrated commitment to and knowledge of equal employment opportunity and affirmative action. Application deadline: December 15, 2019. Applications received after the deadline may be considered until the position is filled.

UTK is the state's flagship campus and leading research institution with a strong partnership with the nearby Oak Ridge National Laboratory (ORNL), where many UTK faculty have ongoing joint positions and/or joint research projects.

EECS at UTK has 49 full-time faculty mem-

bers, 4 members of the National Academy of Engineering, 17 IEEE Fellows, and 12 NSF/DOE CAREER awardees. The department is housed in a new \$37.5 million building completed in 2012 and has an annual research expenditure exceeding \$25 million. EECS has a growing enrollment of over 825 undergraduate and 275 graduate students across the three majors of Electrical Engineering, Computer Engineering, and Computer Science. In addition, the department is offering an undergraduate minor in cybersecurity, a minor in datacenter technology and management, and developing a minor in data science. Successful faculty candidates will be expected to contribute to the continued growth and excellence of EECS.

UTK is located in Knoxville TN, within an easy driving distance to Asheville, Nashville, Atlanta and the Great Smoky Mountains. The City of Knoxville is a hidden gem with an elegant and walkable downtown, rich and varied nightlife cultures, vibrant neighborhoods, eclectic restaurants, and amazing access to outdoor activities of all kinds as well as exciting cultural events throughout the year. From Knoxville's TYS Airport, Knoxville has nonstop flights to 22 major airports in the US, including direct flights to cities such as DC, NYC, Atlanta, Chicago, Denver, and Miami. Furthermore, the City of Knoxville and the surrounding areas boast great K-12 schools and one of the most highly educated populations in the entire US.

From 2007 to 2017, Tennessee's overall economy growth ranked #7 among all 50 US states. In 2019, US News ranks the State of Tennessee in the US as the #1 in fiscal stability, #12 in economy, and #13 in infrastructure.

For any additional questions, please contact Search Committee Chair: Prof. Jian Huang, EECS, University of Tennessee, Knoxville, at huangj@utk.edu.

The University of Tennessee is an EEO/AA/Title VI/Title IX/Section 504/ADA/ADEA institution in the provision of its education and employment programs and services. All qualified applicants will receive equal consideration for employment and admission without regard to race, color, national origin, religion, sex, pregnancy, marital status, sexual orientation, gender identity, age, physical or mental disability, genetic information, veteran status, and parental status.

Western Michigan University Department of Computer Science College of Engineering and Applied Sciences Faculty Positions in Computer Science

Applications are invited for two tenure-track positions in the Department of Computer Science at Western Michigan University (Kalamazoo, MI) starting August 2020. One position is at the Assistant/Associate Professor level and the other is a teaching faculty position at the Faculty Specialist I/II level.

Applicants for the Assistant/Associate Professor position must have earned a Ph.D. in computer science or a closely related field by August 15, 2020. Candidates with expertise in any area of computer science are welcome to apply. Of particular interest are candidates with expertise in artificial intelligence, computational medicine, or computer security. The successful candidate

will be capable of establishing an active research program leading to funding, supervising graduate students, and teaching courses at both the undergraduate and graduate levels. Other duties include development of undergraduate and graduate courses, advising, and service at the university, college, department and professional society levels.

Applicants for the Faculty Specialist I/II position must have earned a M.S. in computer science or a closely related field by August 15, 2020. This position is in support of a new online B.S. in cybersecurity and our existing M.S. in cybersecurity. Candidates should have a background in cybersecurity either through significant work experience or education. The successful candidate will be capable of teaching and developing a broad range of courses in cybersecurity at the undergraduate level. Other duties include service at the university, college, department and professional society levels.

Application screening for both positions will start on January 15, 2020; however, the positions will remain open until filled.

The Department has 290 undergraduates, 45 M.S. students and 25 Ph.D. students. We offer B.S. and M.S. degrees in computer science and cybersecurity, and a Ph.D. in computer science. Current active research areas include artificial intelligence, compiler optimization, data mining, embedded systems/internet of things, formal verification, networks, parallel computing, privacy, scientific computing, and security. More information regarding Western Michigan University, the College of Engineering and Applied Sciences and the Department of Computer Science are available at <http://www.wmich.edu>, <http://www.wmich.edu/engineer>, and <http://www.wmich.edu/cs>, respectively.

The Carnegie Foundation for the Advancement of Teaching has placed WMU among the 76 public institutions in the nation designated as research universities with high research activity.

To apply for this position, please visit: <http://www.wmich.edu/hr/jobs> and provide a cover letter, curriculum vitae, research statement (Assistant/Associate Professor Only), teaching statement, and names and contact information of at least three references.

WMU is an Equal Employment Opportunity/Affirmative Action Employer. Minorities, women, protected veterans, individuals with disabilities and all other qualified individuals are encouraged to apply.



DOI:10.1145/3372389

Dennis Shasha

Upstart Puzzles

Feedback for Foxes

Searching for the best strategy for shifty maneuvers.

THE EARLY HISTORY of differential games gave us the parable of the homicidal chauffeur. The chauffeur, it seems, wants to use his car to collide with a person running on an infinite plane. The car is faster but less maneuverable than the runner. The general question is: What is a good strategy for each?

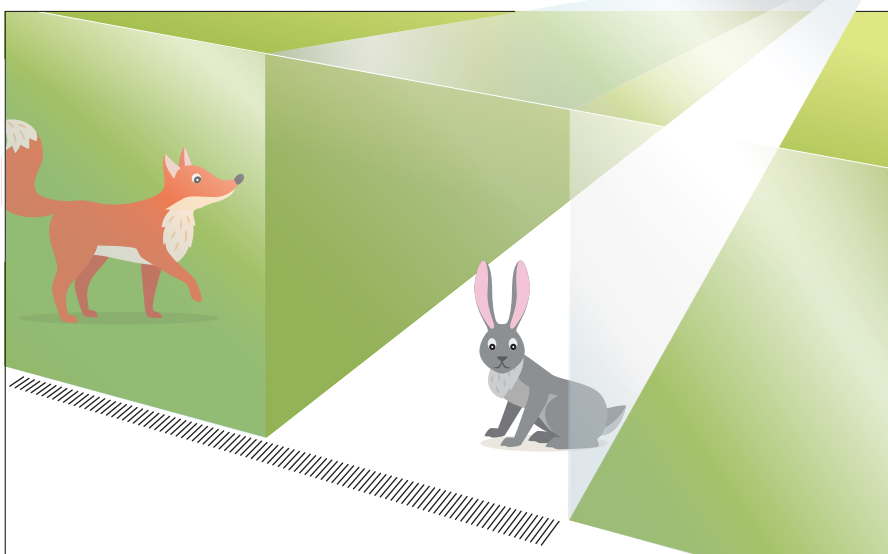
In this Upstart Puzzle, I will simplify and discretize the problem in order to discuss the interrelated questions of feedback, impossibility, and time complexity. To begin, consider the scenario of the figure in this column. There is a scared rabbit inside a straight corridor C that is glass-lined so the position of the rabbit is known to any observer. A fox starts outside the corridor at a point such that the line segment of length D from the fox to the rabbit is perpendicular to C (hereafter, the perpendicularity condition). The fox's goal is to be able to catch the rabbit, which it can do from at most one unit away.

To start, assume the fox is initially 100 units away and moves twice as fast (two distance units per second) as the rabbit.

Warm-Up: How long would it take the fox to catch the rabbit if the fox goes straight to the corridor and then turns in the direction of the rabbit? Assume the rabbit does everything it can to get away.

Solution to Warm-Up: The fox will reach the corridor in 50 seconds at two units per second, the rabbit will be at most 50 units away. In the worst case, the fox will catch the rabbit in another 50 seconds. This gives a total of 100 seconds.

Question: Suppose the fox is allowed to change direction just once



Suppose a rabbit can move one unit per second inside a corridor and a fox two units per second whether inside or outside. Further suppose the fox can change direction once outside the corridor and can take either direction once inside the corridor. When and how should the fox change direction outside the corridor to minimize the time needed until the fox is at most one unit away from the rabbit inside the corridor?

while outside the corridor. Can the fox catch the rabbit in 80 seconds or less?

Solution: Yes. The fox can turn after 30 seconds to point in the direction of the rabbit at that time. The rabbit will have left its original resting place and traveled at most 30 units away. At that time the fox will be only 40 units away from the corridor. The diagonal between the fox and the rabbit is therefore at most of length 50, so in another 25 seconds, the fox will be at the corridor. At that point the rabbit may be 25 units away. The fox will then capture the rabbit in another 25 seconds. This gives a total time of 80 seconds.

Question: Suppose the fox starts 100 units away from the rabbit but could change direction at any time. How slowly could the fox move but

still catch the rabbit in 100 time units?

Solution: The fox could move as slowly as $\sqrt{2}$ distance units per time unit. The approach is as follows: The fox continually maintains itself at a point that satisfies the perpendicularity condition. If the rabbit moves one unit in a time period, this will require the fox to move $\sqrt{2}$ units to be one unit closer to the corridor and still be perpendicular. So in 100 units it will be right on top of the rabbit in the corridor.

Question: Suppose the rabbit can go anywhere on an infinite floor and the fox is initially also on the floor but 100 units away. The fox can turn seven times and goes twice as fast as the rabbit. Can the fox guarantee to catch the rabbit in 100 seconds? Is it possible to do better? [CONTINUED ON P. 103]

IoTDI'20

5th ACM/IEEE International Conference on Internet of Things Design and Implementation



April 21-24, 2020 – CPS-IoT Week – Sydney, Australia

<https://conferences.computer.org/iotDI/2020/>

ACM/IEEE IoTDI is the premier venue for all topics related to the Internet of Things: an interdisciplinary forum to discuss challenges, technologies, and emerging directions in system design and implementation that pertain to the IoT.

Topics covered include:

- Analytic foundations and theory of IoT
- Reliability, security, timeliness, and robustness in IoT systems
- Novel protocols and network abstractions
- Data streaming architectures and data analytics for IoT
- AI/ML for IoT & embedded systems
- IoT-motivated cyber-physical and Industrial IoT (IIoT) systems
- Novel quality requirements and their enforcement mechanisms
- Cloud back-ends and resource management for IoT applications
- Edge and fog computing
- Personal, wearable, and other embedded networked front-ends
- Social computing and human-in-the-loop issues
- Applications for specific domains (smart cities, health, ITS, ...)
- Deployment experiences, case studies & lessons learned
- Evaluation and testbeds
- Energy/power management & harvesting for IoT platforms

General Chairs

Gian Pietro Picco (Univ. of Trento, Italy)
Prashant Shenoy (UMASS, Amherst, USA)

Program Chairs

Valerie Issarny (INRIA, France)
Archan Misra (SMU, Singapore)

Steering Committee Chairs

Tarek Abdelzaher (UIUC, USA)
Hui Lei (IBM Watson Health Cloud, USA)

Poster & Demo Chairs

Christopher Stewart (Ohio State, USA)
Rui Tan (Nanyang Tech. Univ., Singapore)

Publicity Chairs

Josiah Hester (Northwestern Univ., USA)
Oana Iova (INSA Lyon, France)
Feng Lin (Zhejiang Univ., China)

Social Media Chair

Tian Guo (Worcester Polytechnic Inst., USA)

Web Chair

Stephen Lee (University of Pittsburgh, USA)

The 34th Edition of ECOOP

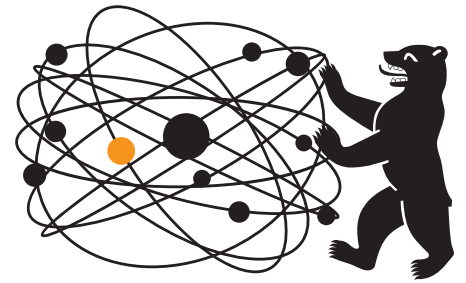
ECOOP is Europe's longest-standing annual Programming Languages conference, bringing together researchers, practitioners, and students to share their ideas and experiences in all topics related to programming languages, software development, object-oriented technologies, systems and applications. It welcomes high quality research papers relating to these fields in a broad sense; solicits both innovative and creative solutions to real problems as well as evaluations of existing solutions—evaluations that provide new insights; encourages the submission of reproduction studies.

Areas of interest

Design, implementation, optimization, analysis, and theory of programs, programming languages, and programming environments.

Publication

Affordable CC-BY [open access](#) in Dagstuhl LIPIcs or as [journal first](#) in ACM TOPLAS or Elsevier SCP.




ECOOP

Berlin 2020

July 13–17



 Based on a photo by Volker Davids
<https://creativecommons.org/licenses/by/2.0/>

Program Committee

Karim Ali, Davide Ancona, Carl Friedrich Bolz-Tereick, John Boyland, Shigeru Chiba, Theo D'Hondt, Wolfgang De Meuter, Sebastian Erdweg, Tim Felgentreff, Olivier Flückiger, Lidia Fuentes, Richard P. Gabriel, Anitha Gollamudi, Elisa Gonzalez Boix, Philipp Haller, Christian Hammer, Felienne Hermans, Atsushi Igarashi, Stephen Kell, Raffi Khatchadourian, Yu David Liu, Hidehiko Masuhara, James Noble, Klaus Ostermann, Patrick Rein, Guido Salvaneschi, Manuel Serrano, Jeremy G. Siek, Friedrich Steimann, Emma Söderberg, Peter Thiemann, Eli Tilevich, Frank Tip, Jan Vitek, Tobias Wrigstad

General Chair

Christian Hammer, U. Potsdam

Program Chair

Robert Hirschfeld, HPI, U. Potsdam

Diversity Chair

Julia Belyakova, Northeastern U.

External Reviewers

Erik Ernst, Matthew Flatt, Jeremy Gibbons, Doug Lea, Crista Lopes, Toni Mattis, Todd Millstein, Jens Palsberg, Tomas Petricek, Benjamin C. Pierce, Joe Gibbs Politz, Tiark Rumpf, Laurence Tratt

AiO

 **In-Cooperation**

 **SIGPLAN**



<https://2020.ecoop.org>