

COMMUNICATIONS OF THE ACM

CACM.ACM.ORG

05/2019 VOL.62 NO.05

Countering the Negative Image of Women in Computing

Algorithmic Randomness

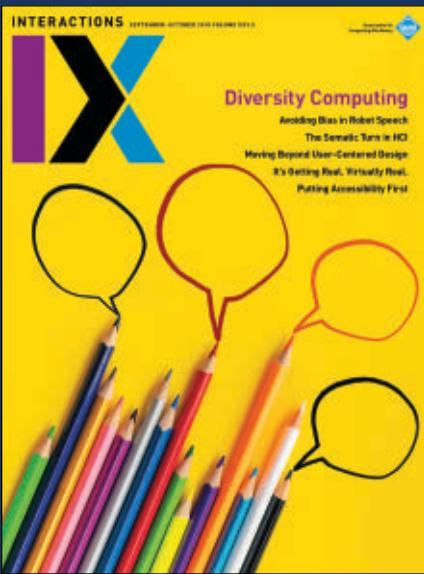
Questioning Quantum

Enterprise Wi-Fi

Continuity and Change
in Internet Law

Association for
Computing Machinery





Association for
Computing Machinery



ACM Seeks New Editor(s)-in-Chief for *ACM Interactions*

The ACM Publications Board is seeking a volunteer editor-in-chief or co-editors-in-chief for its bimonthly magazine *ACM Interactions*.

ACM Interactions is a publication of great influence in the fields that envelop the study of people and computers. Every issue presents an array of thought-provoking commentaries from luminaries in the field together with a diverse collection of articles that examine current research and practices under the HCI umbrella.

For more about *ACM Interactions*, see <http://interactions.acm.org>

Job Description The editor-in-chief is a volunteer position responsible for organizing all editorial content for every issue. These responsibilities include: proposing articles to prospective authors; overseeing the magazine's editorial board and contributors; creating new editorial features, columns, and much more.

An annual stipend will be available for the hiring of an editorial assistant.

Financial support will also be provided for any travel expenses related to this position.

Eligibility Requirements The EiC search is open to applicants worldwide. Experience in and knowledge about the issues, challenges, and advances in human-computer interaction is a must.

The ACM Publications Board has convened a special search committee to review all candidates for this position.

Please send your CV and vision statement of 1,000 words or less expressing the reasons for your interest in the position and your goals for *Interactions* to the search committee at eicsearch@interactions.acm.org, Subject line: RE: Interactions.

**The deadline for submissions is June 1, 2019 or until position is filled.
The editorship will commence on October 1, 2019.
You must be willing and able to make a three-year commitment to this post.**



THE ACM A. M. TURING AWARD

by the community◆ from the community◆ for the community



"Deep neural networks are responsible for some of the greatest advances in modern computer science, helping make substantial progress on long-standing problems in computer vision, speech recognition, and natural language understanding. At the heart of this progress are fundamental techniques developed starting more than 30 years ago by this year's Turing Award winners, Yoshua Bengio, Geoffrey Hinton, and Yann LeCun. By dramatically improving the ability of computers to make sense of the world, deep neural networks are changing not just the field of computing, but nearly every field of science and human endeavor."

Jeff Dean
Google Senior Fellow and SVP of Google AI
Google Inc.

ACM and Google congratulate
**YOSHUA BENGIO
GEOFFREY HINTON and
YANN LeCUN**

For conceptual and engineering breakthroughs that have made deep neural networks a critical component of computing.



Google™

For more information see <http://research.google.com/>

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

COMMUNICATIONS OF THE ACM

Departments

- 5 **Cerf's Up**
APIs, Standards,
and Enabling Infrastructure
By Vinton G. Cerf

- 7 **Vardi's Insights**
Quantum Hype and
Quantum Skepticism
By Moshe Y. Vardi

- 10 **Letters to the Editor**
Don't Ignore the Cost
of 'Embedded Energy'

- 12 **BLOG@CACM**
Implementing Guidelines
for Governance, Oversight of AI,
and Automation

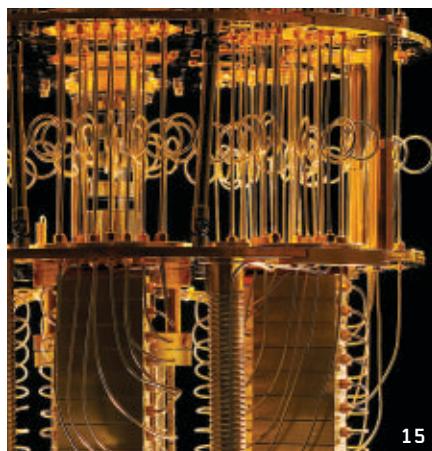
- 31 **Calendar**

- 92 **Careers**

Last Byte

- 96 **Future Tense**
Like Old Times
The Furby singularity
promises eternal conversation
with the untimely departed.
By Ken MacLeod

News



Viewpoints



- 15 **Questioning Quantum**
Researchers hunt for ways to keep
quantum computing honest.
By Chris Edwards

- 18 **Code Talkers**
Using voice input to write programs.
By Neil Savage

- 20 **Deep Insecurities:**
The Internet of Things Shifts
Technology Risk
A more connected world sounds
alluring, but without better
protections, the Internet of Things
could lead to disaster.
By Samuel Greengard

- 24 **Law and Technology**
Continuity and Change
in Internet Law
The fundamentals of the field
of Internet law have remained
consistent, but details have
evolved in response to
technological innovation.
By James Grimmelmann

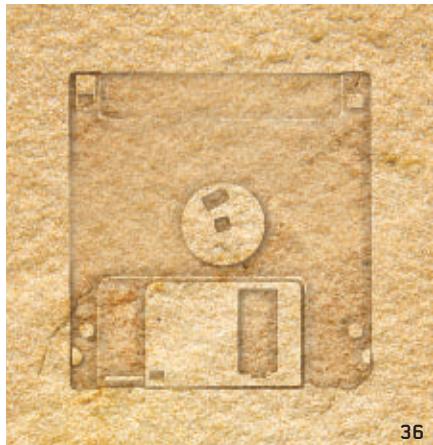
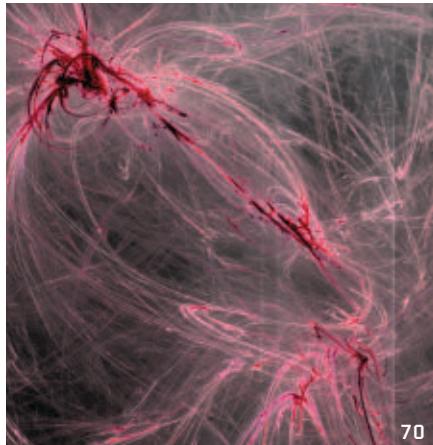
- 27 **Privacy and Security**
Encryption and Surveillance
Why the law-enforcement access
question will not just go away.
By Joan Feigenbaum

- 30 **Education**
What Does It Mean for a Computing
Curriculum to Succeed?
Examining the expansion,
proliferation, and integration
of computing education everywhere.
By Emmanuel Schanzer,
Shriram Krishnamurthi,
and Kathi Fisler

- 33 **Viewpoint**
Enterprise Wi-Fi: We Need Devices
That Are Secure by Default
Seeking to increase awareness
of WPA2 Enterprise network
security technology flaws
and reduce risk to users.
By Alberto Bartoli, Eric Medvet,
Andrea De Lorenzo, and Fabiano Tarlao



Association for Computing Machinery
Advancing Computing as a Science & Profession

Practice**Contributed Articles****Review Articles****36 Achieving Digital Permanence**

The many challenges to maintaining stored information and ways to overcome them.

By Raymond Blum and Betsy Beyer

43 Online Event Processing

Achieving consistency where distributed transactions have failed.

*By Martin Kleppmann,
Alastair R. Beresford,
and Boerge Svingen*

50 Net Neutrality: Unexpected Solution to Blockchain Scaling

Cloud-delivery networks could dramatically improve blockchains' scalability, but clouds must be provably neutral first.

By Aleksandar Kuzmanovic

56 Countering the Negative Image of Women in Computing

A positive image would inspire the capable but underrepresented who might otherwise give up on computing.

By Fay Cobb Payton and Eleni Berki



Watch the authors discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/stereotypical-gendered-media-images>

64 The Limit of Blockchains: Infeasibility of a Smart Obama-Trump Contract

Although smart contracts are Turing complete, it is a misconception that they can fulfill all routine contracts.

By Yongge Wang and Qutaibah M. Malluhi

70 Algorithmic Randomness

Tracing some of the latest advancements in algorithmic randomness.

By Rod Downey and Denis R. Hirschfeldt

Research Highlights**82 Technical Perspective****Compressing Matrices for Large-Scale Machine Learning**

By Zachary G. Ives

83 Compressed Linear Algebra for Declarative Large-Scale Machine Learning

By Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, and Berthold Reinwald



Watch the authors discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/compressed-linear-algebra>



About the Cover:
Does the stereotypical image of women in computing, as portrayed by many media outlets, impact the growth of women in the field? Yes, indeed, say Faith Cobb Payton and Eleni Berki, authors of this month's cover story (p. 56). Cover by Andrij Borys Associates, using photo by Anastasiya Tsiasevnikava.





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.

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 Randin
Director, Office of SIG Services
Donna Cappo
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 Anderst-Kotis; Susan Dumais; Renée McCauley; Claudia Bauzer 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
Abhiram Ranade
ACM China Council
Wenguang Chen

PUBLICATIONS BOARD

Co-Chairs
Jack Davidson and Joseph Konstan
Board Members
Phoebe Ayers; Edward A. Fox; Chris Hankin; Xiang-Yang Li; Nenad Medvidovic; Tulika Mitra; Sue Moon; 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; acmopo@acm.org

Computer Science Teachers Association
Jake Baskin
Executive Director

STAFF

DIRECTOR OF PUBLICATIONS

Scott E. Delman
caca.publisher@acm.acm.org

Executive Editor

Diane Crawford

Managing Editor

Thomas E. Lambert

Senior Editor

Andrew Rosenblom

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 Balaquit

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

acmhelp@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

2 Penn Plaza, Suite 701, New York, NY 10121-0701
T (212) 626-0686
F (212) 869-0481

Advertising Sales Account Manager

Ilia Rodriguez

ilia.rodriguez@hq.acm.org

Media Kit

acmmediasales@acm.org

Association for Computing Machinery (ACM)

2 Penn Plaza, Suite 701
New York, NY 10121-0701 USA
T (212) 869-7440; F (212) 869-0481

EDITORIAL BOARD

EDITOR-IN-CHIEF

Andrew A. Chien

eic@cacm.acm.org

Deputy to the Editor-in-Chief

Lihan Chen

caca.deputy.to.eic@gmail.com

SENIOR EDITOR

Moshe Y. Vardi

NEWS

Co-Chairs

Marc Snir and Alain Chesnais

Board Members

Monica Divitini; Mei Kobayashi;

Rajeev Rastogi; François Sillion

VIEWPOINTS

Co-Chairs

Tim Finin; Susanne E. Hambrusch;

John Leslie King; Paul Rosenbloom

Board Members

Michael L. Best; Judith Bishop;

James Grimmelmann; 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 Musick; Erik Meijer;

George Neville-Neil; Jim Waldo;

Meredith Whittaker

CONTRIBUTED ARTICLES

Co-Chairs

James Larus and Gail Murphy

Board Members

William Aiello; Robert Austin; Kim Bruce;

Alan Bundy; Peter Buneman; Jeff Chase;

Andrew W. Cross; Yannis Ioannidis;

Gal A. Kaminka; Igor Markov;

Lionel M. Ni; Adrian Perrig; Doina Precup;

Marie-Christine Rousset; Krishnan Sabnani;

m.c. schraefel; Ron Shamir; Alex Smola;

Sebastian Uchitel; Hannes Werthner;

Reinhard Wilhelm

RESEARCH HIGHLIGHTS

Co-Chairs

Azer Bestavros and Shriram Krishnamurthi

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; Sven Koenig;

Greg Morrisett; Tim Roughgarden;

Guy Steele, Jr.; Robert Williamson;

Margaret H. Wright; Nicholai 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 © 2019 by Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to publish from permissions@hq.acm.org or fax (212) 869-0481.

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

Subscriptions

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

ACM Media Advertising Policy

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

Single Copies

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

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

POSTMASTER

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

Printed in the USA.



Association for
Computing Machinery





DOI:10.1145/3322094

Vinton G. Cerf

APIs, Standards, and Enabling Infrastructure

APPLICATION PROGRAMMING INTERFACES (APIs) are handy programming tools for describing how one program can access the functionality of another. In many cases, an API might manifest as the definition of a subroutine call, describing how to reference the subroutine (for example, its *name*) and what *parameters* the subroutine is expecting and in what format, and the nature and format of any returned parameters resulting from invoking the subroutine call. It is common for *libraries* of subroutines to be created and APIs to them standardized so that programmers using the library can exercise the functionality of the subroutines at need. Interestingly, the standardization of the APIs can lead to the property that a collection of software that relies on the library subroutines might run successfully on more than one, independently programmed library, as long as the APIs used are the same (that is, refer to the same names and use the same parametric specifications).

This aspect of APIs leads to the same kind of interoperability that the Internet Protocol has provided in the Internet. Many protocols that lie *above* the IP *layer* (for example, transmission control protocol, user datagram protocol, real-time protocol) can make use of a substantial array of alternative underlying network technologies (for example, Ethernet, multiprotocol label switching, asynchronous transfer mode switches, software-defined networks, frame relay, point-to-point radio, laser links, and so on) without knowing anything about how these underlying layers work. They “see” the Internet through the lens of the standard Internet Pro-

tocol with its standard packet format and its addressing structure.

The Internet Protocol is sometimes referred to as the *narrow waist* of the Internet Protocol architecture. Think of an hourglass with a narrow waist. Above are all the protocols that are encapsulated in standard Internet Protocol packets and below are all the possible carriers of the IP packets. An API has a similar property. Any number of programs can be written that call upon the functionality of the libraries or operating system functions referenced by the APIs. More than one library or operating system can be programmed to behave in accordance with the standardized APIs. The commonality of the API *enables* the alternative implementation of operating systems and libraries, permitting “mixing and matching” of the application programs and the independently produced libraries or operating systems supporting them.

The widespread sharing of common or standardized APIs confers rich opportunities for choices of operating system or library implementations for the programming of applications. Underlying hardware, library, and operating system implementations can host the same application software on a wide range of platforms. The investment in platform, operating system, and library software can yield widespread benefits to the application programs that can run on any of them. Applications don’t have to be rewritten and users benefit from enhanced choices.

Widely used programming languages often define common subroutines to make programming more efficient. Programmers do not have to create all new subroutines for com-

mon functions, rather, they can refer to the common functions using common APIs. For this benefit to be widely realized and to avoid having to reprogram applications to refer to the same functions by different names and formats, adoption of standardized APIs induces portability of applications to run on multiple platforms. Of course, these same programming languages typically also allow programmers to define their own, idiosyncratic subroutines for functions that may be unique to a particular application.

The availability of common libraries referenced by a common set of APIs also has the benefit that the common libraries may receive additional scrutiny for reliability and security owing to widespread use. But the standard APIs also allow for distinctly programmed libraries or operating system functions that are matched to the underlying hardware platform and its peculiarities and capabilities that are invoked by a common set of APIs or system calls.

A powerful example of standardization is the so-called Portable Operating System Interface (POSIX)^a standardized by the IEEE. Innumerable applications intended to run in the UNIX operating system environment can be made to run on many variations of UNIX that implement the POSIX API. This is but one of many examples of the powerful, *enabling* effect of adopting common APIs with standardized reference names and parametric format. □

a <https://en.wikipedia.org/wiki/POSIX>

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

Copyright held by author.

ACM-IMS Data Science Summit

June 15, 2019 | Palace Hotel, San Francisco

An interdisciplinary event bringing together researchers and practitioners to address deep learning, reinforcement learning, robustness, fairness, ethics, and the future of data science.

Computing and statistics underpin the rapid emergence of data science as a pivotal academic discipline. ACM and IMS—the Institute of Mathematical Statistics—the two key academic organizations in these areas, have launched a new joint venture to propel data science and to engage and energize our communities to work together.

ACM and IMS will hold an all-day launch event to address topics such as deep learning, reinforcement learning, fairness, and ethics, in addition to discussions about the future of data science and the role of ACM and IMS.

Panels and Panelists

Deep Learning, Reinforcement Learning, and Role of Methods in Data Science

- Shirley Ho, Flatiron Institute
- Sham Kakade, University of Washington
- Suchi Saria, Johns Hopkins University
- Manuela Veloso, J.P. Morgan, Carnegie Mellon University

Robustness and Stability in Data Science

- Aleksander Madry, Massachusetts Institute of Technology
- Xiao-Li Meng, Harvard University
- Richard J. Samworth, University of Cambridge, Alan Turing Institute
- Bin Yu, University of California, Berkeley

Fairness and Ethics in Data Science

- Alexandra Chouldechova, Carnegie Mellon University
- Andrew Gelman, Columbia University
- Kristian Lum, HRDAG (Human Rights Data Analysis Group)

Future of Data Science

- Michael I. Jordan, University of California, Berkeley
- Adrian Smith, Alan Turing Institute

Keynote Speakers



Jeffrey Dean
Google



David Donoho
Stanford University



Daphne Koller
insitro
Stanford University

Seating is limited, so register early!
<https://www.acm.org/data-science-summit>





Moshe Y. Vardi

DOI:10.1145/3322092

Quantum Hype and Quantum Skepticism

THE FIRST THIRD of the 20th century saw the collapse of many absolutes. Albert Einstein's 1905 special relativity theory eliminated the notion of absolute time, while Kurt Gödel's 1931 incompleteness theorem questioned the notion of absolute mathematical truth. Most profoundly, however, quantum mechanics raised doubts on the notion of absolute objective reality. Is Schrödinger's cat dead or alive? Nearly 100 years after quantum mechanics was introduced, scientists still are not in full agreement on what it means.

The problem with objective reality stems from the *superposition principle*. In a nutshell, quantum systems can exist in a superposition of their possible observable states before measurement. While a classical bit has a unique value, 0 or 1, a quantum bit, or *qubit*, exists as a superposition of two classical bits. The physicist Richard Feynman hinted at the possibility of using quantum effects for computation in his 1959 lecture, "There's Plenty of Room at the Bottom." But this possibility seemed somewhat remote, until Peter Shor's seminal 1994 paper, where he showed it is theoretically possible to use quantum computation to factor numbers in polynomial time, which would break current public-key cryptographic schemes. This made the realization of quantum computing one of the holy grails of computing in the 21st century. In fact, quantum computing is an important part of one of the U.S. National Science Foundation's "Ten Big Ideas."

The popular media regularly reports breathlessly on quantum computing: "Quantum computing will break your encryption in a few years"; "Why quantum's computing time is now"; and "The computer that could rule the world." Yet the physical realization of quantum comput-

ing has been a hard slog. A Canadian company, D-Wave Systems, has claimed to be the world's first company to sell computers that exploit quantum effects in their operation. But the D-Wave machine is far from being a general quantum computer, and several researchers disagree with D-Wave's claims.

In fact, several quantum-computing researchers have expressed skepticism regarding the physical realizability of the quantum-computing dream.^a Quantum skeptics agree that quantum computation does offer an exponential advantage of classical computation in theory, but they argue it is not physically possible to build scalable quantum computers. Gil Kalai is one of the most prominent quantum skeptics. All physical systems are noisy, he argues,^b and qubits kept in highly sensitive superpositions will inevitably be corrupted by any interaction with the outside world. In contrast, quantum-skepticism skeptics, such as Scott Aaronson, view the realizability of quantum computing as an outstanding question in physics,^c and regard the skeptical view as representing an implausible revolution in physics.

A recent U.S. National Academies report^d reviewed current progress and prospects of quantum computing, taking a sober view of the field. Given the current state of quantum computing and the significant challenges that still must be overcome, the report argues it is highly unlikely a quantum computer that can compromise public-key cryptography—a basis for the security of most of today's computers and networks—will be built within the next decade. Yet, because re-

placing an established Internet protocol generally takes over a decade, work to develop and deploy algorithms that are resilient against an attack by a quantum computer is critical now, the report advised.

The report identified major challenges that lie ahead for quantum computing. Agreeing somewhat with Kalai, the report stated the need to correct the errors in a quantum system, without which it is unlikely that a highly complex quantum program would ever run correctly on the system. Error-correcting algorithms incur significant costs, so in the near term quantum computers are likely to be error-prone, the report concluded.

An important part of the report is an analysis of why and how computing technology scaled exponentially in performance for over half a century. This scaling was mostly the result of a virtuous cycle, where products using the new technology allowed the industry to make more money, which it then used to create newer technology. For quantum computing to be similarly successful, it must create a virtuous cycle to fund the development of increasingly useful quantum computers. But the beauty of classical computing is that developing algorithms is incredibly easy. Every teenager writing a program is developing an algorithm. In contrast, in more than 25 years of intense research on quantum computing, only a few dozen algorithms have been developed.^e It is conceivable that governments will pour major investments into a small number of critical quantum-computing applications, but this will not give rise to the thriving marketplace that is needed to sustain the virtuous cycle. Count me a quantum skeptic! □

e <https://bit.ly/2uq1rof>

See p. 15 for more on the quantum argument.

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.

ACM ON A MISSION TO SOLVE TOMORROW.

Dear Colleague,

Without computing professionals like you, the world might not know the modern operating system, digital cryptography, or smartphone technology to name an obvious few.

For over 70 years, ACM has helped computing professionals be their most creative, connect to peers, and see what's next, and inspired them to advance the profession and make a positive impact.

We believe in constantly redefining what computing can and should do.

ACM offers the resources, access and tools to invent the future. No one has a larger global network of professional peers. No one has more exclusive content. No one presents more forward-looking events. Or confers more prestigious awards. Or provides a more comprehensive learning center.

Here are just some of the ways ACM Membership will support your professional growth and keep you informed of emerging trends and technologies:

- Subscription to ACM's flagship publication ***Communications of the ACM***
- Online books, courses, and videos through the **ACM Learning Center**
- Discounts on registration fees to ACM Special Interest Group conferences
- Subscription savings on specialty magazines and research journals
- The opportunity to subscribe to the **ACM Digital Library**, the world's largest and most respected computing resource

Joining ACM means you dare to be the best computing professional you can be. It means you believe in advancing the computing profession as a force for good. And it means joining your peers in your commitment to solving tomorrow's challenges.

Sincerely,



Cherri M. Pancake
President
Association for Computing Machinery



Association for
Computing Machinery

Advancing Computing as a Science & Profession

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.



Association for
Computing Machinery

**ACM Journal of
Data and
Information Quality**



*Providing Research and Tools
for Better Data*

ACM JDIQ is a multi-disciplinary journal that attracts papers ranging from theoretical research to algorithmic solutions to empirical research to experiential evaluations. Its mission is to publish high impact articles contributing to the field of data and information quality (IQ).



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

DOI:10.1145/3322209

Don't Ignore the Cost of 'Embedded Energy'

I WAS HAPPY TO see Andrew A. Chien's Editor's Letter "Owning Computing's Environmental Impact" (Mar. 2019) addressing the topic of our common responsibility for the ecological cost of our industry but surprised it said nothing about so-called "embedded energy" when exploring the growth of electronic waste. Even if all that waste could be miraculously disposed of properly and recycled, the resulting reduction in the ecological cost of the devices would be small compared to the unrecoverable part of the energy needed to manufacture them in the first place, which for many devices is, according to a number of studies, is greater than all the energy they will consume during their functional lifetimes.

A few pages later, Logan Kugler's news story "Building a Better Battery" about building bigger and more energy-efficient batteries likewise completely overlooked the ecological impact of the batteries themselves.

For good measure, it was followed several pages later by Keith Kirkpatrick's news story "Electronics Need Rare Earths" that focused on the geopolitical and economic issues surrounding those "exotic" elements, which ironically are omnipresent in modern technology, but made no mention of the ecological cost incurred in the extraction of the materials that car-

ry over to the ecological cost of producing the gadgets we all throw away at a staggering and increasing rate.

Maybe rather than invent new devices to manufacture, we should focus on figuring out how to prolong the lives of those we have.

Stefan Monnier, Montréal, Canada

Response from the Editor-in-Chief:

The notion of embedded energy is indeed an important consideration, and one that the computing community and industry must eventually confront. However, one challenge involving environmental responsibility for computing is the historic reluctance of the computing community to face our own direct impact. That must come first, and hence the focus of my editorial on direct effect—emissions and waste. I do agree that beyond that, system lifetime, embedded energy, lifecycle impact, and more must be addressed.

Andrew A. Chien, Chicago, IL, USA

For More Women in Computing, Give to Alma Mater

It was good to read about ongoing efforts to increase the participation of women in the computing field, as in Carol Frieze's and Jeria L. Quesenberry's Viewpoint "How Computer Science at CMU Is Attracting and Retaining Women" (Feb. 2019). I would thus like to point out that there is something all computer scientists can do toward that end, not just rely on institutions to do it for us. Donations to one's alma mater earmarked for scholarships for women are within the reach of any of us. Such donations are also of course tax deductible. Moreover, it is important not to let donations jeopardize the retirement future of the donor, but at the same time for many of us, money, fortunately, is available for such a worthwhile end. I myself donated scholarship money for women and underrepresented minorities studying in the computing

Donations to one's alma mater earmarked for scholarships for women are within the reach of any of us.

field to my own alma mater—Culver-Stockton College in Canton, MO. At age 87, it was easy for me to recognize the tax benefits of doing it, the societal benefits to computer science, and choose an amount that allowed me to continue to ensure for my alma mater and myself a solid financial future.

Robert L. Glass, Toowong, Australia

Broaden Focus, Even in the Library

Vinton G. Cerf's Cerf's Up column "Libraries Considered Hazardous" (Feb. 2019) reminded me of a prescient passage from Norbert Wiener's book *Cybernetics*¹ published more than 70 years ago: "There is a well-known tendency of libraries to become clogged by their own volume; of the sciences to develop such a degree of specialization that the expert is often illiterate outside his own minute specialty. Vannevar Bush has suggested the use of mechanical aids for the searching through vast bodies of material. These probably have their uses, but they are limited by the impossibility of classifying a book under an unfamiliar heading unless some particular person has already recognized the relevance of the heading for that particular book. In the case where two subjects have the same technique and intellectual content but belong to widely separated fields, this still requires some individual with an almost Leibnizian catholicity of interest."

Wiener may have conflated classifying with indexing but made several points as pertinent today as they were in 1948.

Reference

1. Wiener, N., *Cybernetics*. John Wiley & Sons, Inc., New York, 1948, 158.

Charles H. Davis, Bloomington, IN, USA

Misguided CS Education

In his article "Four Ways to Make CS and IT More Immersive" (Oct. 2017), Thomas A. Limoncelli referred to "best practices" and "best-of-breed DevOps practices" when working in IT organizations. The notion that such "best practices" exist at all and can be clearly identified to cover any software engineering circumstance is naive, and the examples of the practices he men-

tioned include specifics (such as Git, Jenkins, IDEs, Web applications, and HTML) are likely to eventually fade away. Indeed, newer tools will claim the role of existing tools, but relying too heavily on regularly changing dependencies is a disadvantage.

The method Limoncelli suggested for teaching computer science students these alleged best practices is somewhat like brainwashing; that is, pretend that only such ideal practices exist, at least for a while, so later, when confronted with a less-than-ideal situation, it will "disgust them."

I was disturbed by how Limoncelli's proposed method attempts to hide certain aspects of a situation from the students' view. Presumably, the method means teachers should resist answering certain kinds of questions. I recently met a software engineer who, apparently as the result of being exposed to such a method, was limited to considering only software structured according to the idea of object orientation. And that even discussing certain other parts of the software was strongly (and counterproductively) rejected.

Teachers must face the fact that different students differ in how they learn. Limoncelli's method will definitely not work for everybody, even if it helped educate him. More likely it will leave significant gaps in the education of future computer scientists.

Thorkil Naur, Odense, Denmark

Author Responds:

Naur and I agree that "best practice" is a moving target and students should learn a variety of tools that embody it. I disagree that the solution is to give up and not update the curriculum. I would like to see professors incentivized to stay current and incorporate their learning into the curriculum. Consider that 10 years ago teaching DevOps principles would have been radical and risky. Now the biggest threat is depriving students of such knowledge.

Thomas A. Limoncelli, Bloomfield, NJ, USA

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

© 2019 ACM 0001-0782/19/5

Coming Next Month in COMMUNICATIONS

ACM A.M. Turing Award Recipients:

**Geoffrey Hinton,
Yoshua Bengio,
Yann LeCun**

The Challenge of Crafting Intelligible Intelligence

Engineering Trustworthy Systems

Personal Data and the Internet of Things

A New Era of Programmable Solid-State Storage in Cloud Datacenters

Garbage Collection as a Joint Venture

Troubling Trends in Machine Learning Scholarship

How to Create a Great Team Culture (and Why It Matters)

Heterogeneous Von Neumann/Dataflow Microprocessors

Plus the latest news about continuous AI learning, ethics in technology jobs, and silicon foundries.



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:10.1145/3317669

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

Implementing Guidelines for Governance, Oversight of AI, and Automation



Ryan Carrier
Governance and Oversight Coming to AI and Automation: Independent Audit of AI Systems

<http://bit.ly/2tPlSk>
February 12, 2019

Governance and independent oversight on the design and implementation of all forms of artificial intelligence (AI) and automation is a cresting wave about to break comprehensively on the field of information technology and computing.

If this is a surprise to you, then you may have missed the forest for the trees on a myriad of news stories over the past three to five years. Privacy failures, cybersecurity breaches, unethical choices in decision engines, and biased datasets have repeatedly sprung up as corporations around the world deploy increasing numbers of AIs throughout their organizations.

The world, broadly speaking, combined with legislative bodies, regulators, and a dedicated body of academics operating in the field of AI Safety, have

been pressing the issue. Now guidelines are taking hold in a practical format.

IEEE's Ethically Aligned Design (<https://ethicsinaction.ieee.org/>) is the Gold Standard for drawing together a global voice, using open source crowdsourcing techniques to assert some core ethical guidelines. Additionally, the standards body is deeply into the process of creating 13 different sets of standards covering areas from child and student data governance to algorithmic bias.

Others have joined the call. The EU recently created an ethical guidelines working group (<https://ec.europa.eu/digital-single-market/en/news/draft-ethics-guidelines-trustworthy-ai>), and one of the original efforts includes: The Future of Life's AI principles (<https://futureoflife.org/ai-principles/>) created in conjunction with its Asilomar conference back in 2017. Even some specific companies, like Google, have gotten into the act, creating their own set of public ethical guidelines (<https://www.blog.google/technology/ai/ai-principles/>). This collection of work demonstrates the importance

and a considerable amount of effort to govern and oversee AI and automation from all corners of the globe.

Independent Audit of AI Systems is the next evolution of that governance: genuine accountability. It will build upon these global community guidelines to give audit and assurance companies the ability to assess compliance amongst companies employing AI and Automation. Let me explain how this all works.

ForHumanity, a non-profit organization, will sit at the center of key constituencies ranging from the world's leading audit/assurance firms to global academia and back to the companies themselves. ForHumanity's "client-base," however, is ONLY humanity (thus the name). Revenue to operate ForHumanity comes from donations, and companies who wish to license the audit process and the SAFEAI brand once compliance is achieved. Unlike the Credit Agency business model, where the rated entity "pays" for its rating, creating an inherent conflict of interest, ForHumanity does not exist to profit from audit compliance. This allows ForHumanity to act purely in the best interest of soci-

ety-at-large, seeking out “best practices” in the following areas (silos):

1. Ethics
2. Bias
3. Privacy
4. Trust
5. Cybersecurity

The ForHumanity team will operate global office hours and have dedicated staff for each of these audit silos seeking, sourcing, collating, moderating, and facilitating the search for “auditable best practices.” “Auditable” means binary, there is either compliance or non-compliance with the audit rule. Where we are unable to craft rules that are auditable, they will not become part of the audit. Gray areas are not the domain for compliance or non-compliance. Where gray areas are found (and there will be many), it will be the goal of the ForHumanity team, in conjunction with the global community, to distill these issues into binary parts and/or simply introduce transparency and disclosure (which is a compliance/non-compliance concept) into areas that in the past have been opaque, even if they remain gray. With transparency and disclosure, at least the public can choose which shade of gray they prefer.

Audit silo heads will hold two-hour office hours each day of the week. The times will be scheduled to accommodate the work day all around the world. Additionally, those who register may participate in an online, permanent chat designed to allow people to track the discussion over time at their convenience.

The creation and maintenance of the Independent Audit of AI Systems will be an ongoing and dynamic process. It will be fully transparent to all who would choose to participate, provided they join the discussion and participate with decorum. Each audit silo head will engage the community and seek points of consensus on auditable best practices. Once they believe they have found one, that audit rule will be proposed to the community at-large for consent or dissent. Dissent will also be tracked and shown to the Board of Directors for consideration. It is the role of the audit silo head to manage dissent and work toward reducing and eliminating dissent over time. If consensus is achieved, then that audit rule will be proposed to the ForHumanity Board of Directors. The Board will have

the final say, quarterly, on the current set of audit best practices rules. ForHumanity is dedicated to ensuring the Board of Directors is diversified across ethnic, gender and geography.

ForHumanity exists to achieve the best possible results for all. It does not get paid for the work it provides; instead, it is operated on a non-profit basis, licensing the SAFEAI logo and SAFEAI audit brand to those entities who submit to and pass the SAFEAI audit. In essence, we are asking those who benefit from the audit to pay it forward, so that we may continue and expand our work. Once an audit is passed, the company may *choose* to license the brand/logo in order to demonstrate to the world their compliance with the SAFEAI audit. The brand/logo may also be used by companies that wish to sell specific products that are SAFEAI-compliant as well. The brand/logo may be used on their packaging to enhance their ability to market and sell their product, versus competitors who may not have achieved SAFEAI audit compliance.

The rules are 100% transparent, so when an audit is conducted, compliance is expected. However, there may be areas of the audit that require remediation. Companies will be given a window of time in which to remedy their shortfall. Failure to comply will result in a public “failure” and transparency with regard to the noncompliance. This element is crucial to protect the SAFEAI brand, as well as to protect humanity from unsafe, dangerous, or irresponsible AIs. Over time, we expect the SAFEAI seal of approval to be an important part of consumers’ decision-making process for products and services. The theory is simple:

If we can make good, safe, and responsible AI profitable, whilst making dangerous and irresponsible AIs costly, then we achieve the best possible result for humanity.

In 1973, the major accounting firms came together and formed FASB (The Financial Accounting Standards Board), and the result of that work was GAAP (Generally Accepted Accounting Principles) which still govern financial accounting today. That work eventually became mandated by the SEC (and other jurisdictions around the world) for all publicly listed companies. The investing world was significantly improved through this clarity and uni-

formity. Third-party oversight gives great confidence to those who examine financial accounts to inform their decisions. It is a cornerstone of a robust market economy.

ForHumanity is working with major players to bring Independent Audit of AI Systems to fruition with the same robust and comprehensive oversight and accountability for artificial intelligence, algorithms, and automation. An effort like this will not eliminate fraud and irresponsible behavior. The world still suffered through the Enron and WorldCom financial accounting scandals, but by and large, accountability and universal rules will go a long way toward mitigating dangerous, irresponsible, and unfair behavior that has already challenged the world of technology. Microsoft and Google just recently informed their investors that ethics, bias, privacy, and other “risk factors” may occur, putting shareholders of those companies at risk (https://www.wired.com/story/google-microsoft-warn-ai-may-do-dumb-things/?mbid=social_twitter_onsiteshare).

Independent Audit is the best mechanism for companies to examine their compliance with best-practice rules and to make changes, mitigating downside risk. We look forward to working with them. We also ask each of you to participate as well. There are many ways to do so:

1. Track the progress of the SAFEAI audits and when the seal of approval begins to be used by compliant companies, buy those products.
2. Use services from companies that are SAFEAI-compliant.
3. Participating in the process for setting the auditing rules; it is open and all may join. You may not be a technical expert or have ideas to put forward, but your votes will count just as much as everyone else’s.
4. Donate to ForHumanity; we are a non-profit and you can find us at <http://forhumanity.center>
5. Tell others about the SAFEAI brand and help us spread the word.

Ryan Carrier is executive director of ForHumanity, a non-profit organization created to examine and mitigate the downside risks associated with Artificial Intelligence and Automation. Independent Audit of AI Systems is one such risk mitigation tool.

Introducing ACM Digital Threats: Research and Practice

A new journal in digital security from
ACM bridging the gap between academic
research and industry practice

Now Accepting Submissions

ACM 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.

The journal welcomes the submission of scientifically rigorous 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.

DTRAP invites researchers and practitioners to submit manuscripts that present scientific observations about the identification, prevention, mitigation, and elimination of digital threats in all areas, including computer hardware, software, networks, robots, industrial automation, firmware, digital devices, etc.



For more information and to submit your work,
please visit <https://dtrap.acm.org>.



Association for
Computing Machinery

Advancing Computing as a Science & Profession

Science | DOI:10.1145/3317673

Chris Edwards

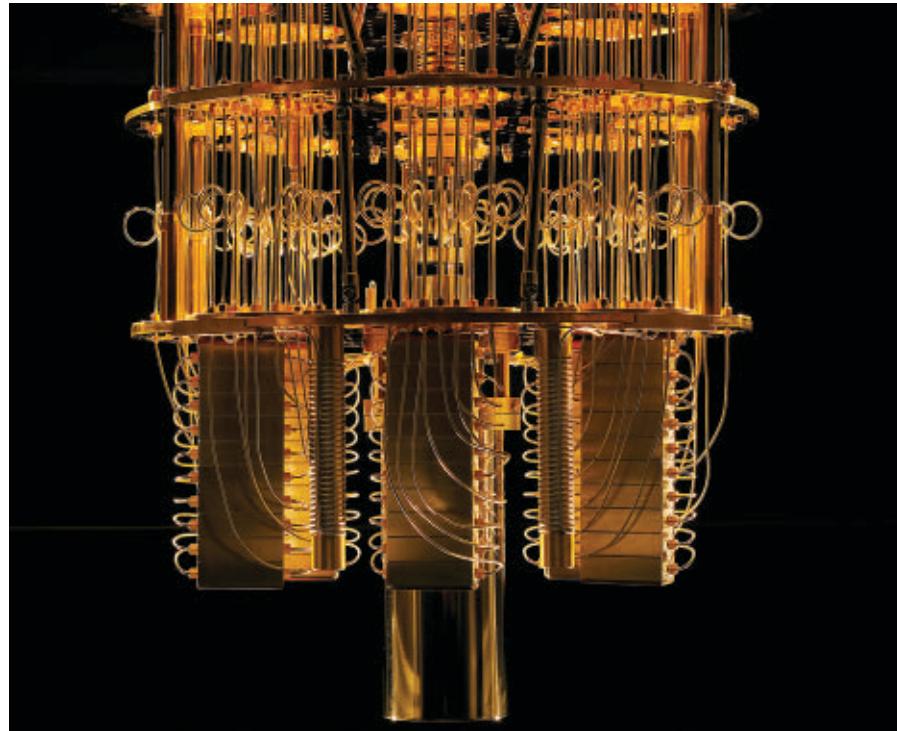
Questioning Quantum

Researchers hunt for ways to keep quantum computing honest.

AT THE BEGINNING OF December last year, a committee set up by the U.S. National Academies of Sciences, Engineering, and Medicine said it had come to the conclusion a viable quantum computer with the ability to break ciphers based on today's encryption algorithms is a decade or more away, but they are coming. Committee chair Mark Horowitz said he and his colleagues could see no fundamental reason, in principle, why a functional quantum computer could not ever be built.

When they do finally arrive, quantum computers pose a number of problems for computer scientists when it comes to determining whether they work as expected. Quantum computers can make use of the property of superposition: where the bits in a register in the machine do not exist in a single known state, but in a combination of states. Each state has a finite probability of being the one recorded when the register is read and the superposition collapses.

Quantum computers can use quantum entanglement and interference between states in superposition to slash the number of compute operations needed for certain problems. Interference is a major part of the algorithm developed by Massachusetts Institute of Technology professor of applied mathematics Peter Shor to fac-



The IBM Q dilution refrigerator cools the quantum processor to 15 millikelvin (-459 degrees Fahrenheit).

tor the large primes that form the basis of many current encryption systems. Although it is possible to simulate the behaviors of a quantum computer with a conventional computer, the programs are extremely slow because the complexity of the state space grows exponentially as more quantum bits—or qubits—are added.

Without a radical improvement in technology, production machines are likely to be large, expensive, and only suitable for use as coprocessors for specific problems. The most likely usage model is as a server accessed using the conventional Internet, or a network able to support communication based on quantum states.

California Institute of Technology researcher Alexandru Gheorghiu points out a number of potential problems a remote user will face: "If this communication is performed over a network, one can imagine man-in-the-middle attacks in which malicious parties are trying to deceive us. Alternatively, it could be that the party claiming to have a quantum computer is lying," Gheorghiu says.

Verification of a result can be simple in some cases. Shor's algorithm for factoring large primes provides in its results a convenient method for testing whether a quantum computer behaved correctly. If the quantum computer outputs the wrong primes, it can be judged on whether it produced two primes that, when multiplied together, generate the expected value.

In the long term, few problems for which quantum computation offers a dramatic speedup over their classical counterparts will be as easy to verify as Shor's algorithm. Problems for which there is no obvious solution cannot be readily checked, unless the user is willing to wait days or months for a classical computer to produce a result.

In order to address this problem, researchers, starting with Dorit

Aharonov and colleagues at the Hebrew University of Jerusalem, began working on the use of interactive proofs for the problem a little over a decade ago. With interactive proofs, someone who wants to verify whether the system they are using is providing them with correct answers tests it time and again with different requests. If the answers are consistently believable, they can be confident the application server is functioning properly.

To provide reliable tests, interactive proofs assume the verifier can call on assistance from an oracle: a system that can compute anything immediately. In practice, there is no technology that can deliver such an all-powerful oracle, but for the purposes of checking quantum computing, another quantum computer acting as a 'prover' can fulfill the role of oracle. But why would anyone trust that quantum prover, either?

If a prover has been compromised, it might use knowledge of the algorithms it is expected to test to provide believable but false answers. The answer to this is for the verifier to disguise its intent, generally by encrypt-

ing the tasks and the data. "It's harder for a prover to misbehave when performing a certain computation if they don't know what the computation is," Gheorghiu says. However, further checks are needed to be certain.

Around the same time Aharonov's work was published, Gheorghiu's Ph.D. advisor at the University of Edinburgh, Elham Kashefi, developed with Anne Broadbent and Joseph Fitzsimons of the University of Waterloo, Ontario, a protocol they call Universal Blind Quantum Computation (UBQC). As well as using encryption to hide the intent of computational requests supplied to the prover and the target quantum computer, traps are buried in the encrypted. These trigger if the server fails to compute correctly, alerting the verifier client to the problem. Because of the many iterations needed to complete a proof, there is a potentially heavy computational cost.

Broadbent says she had reservations about the complexity of the protocols developed in the earlier work, and set about creating a simpler system. She opted to work with a technique based on quantum-circuit mod-

ACM News

Bees With Backpacks: The Next Army of Data Collectors?

Researchers are outfitting insects with all sorts of sensors to collect a wide array of data in the wild.

At the University of Washington (UW), researchers attached sensors that can gather farm data (temperature, humidity, and light) for up to seven hours to the back of a bee. Says UW's Sawyer B. Fuller, "Once you have flying sensor nodes, you can imagine the ability to collect a lot more information about the health of a farm, such as distribution of pollutants, moisture levels, or being able to detect diseases early."

Meanwhile, researchers at the University of California, Berkeley have developed a miniature radio system that is able to communicate with a backpack mounted on a beetle, in order to control some aspects of the beetle's flight.

Nanyang Technological University in Singapore researchers are also working

to control the movement of beetles with backpacks, while researchers at the University of Connecticut are having some success influencing insect movement using a prototype "system on a chip" that is wired into an insect's brain.

Scientists say one of the main attractions of combining living insects with technology is that insects are marvels of engineering that roboticists have been unable to duplicate.

"The more I work in robotics, the more I am impressed by the locomotion abilities of animals," says Auke Jan Ijspeert, a professor in the Biorobotics Laboratory of the Ecole polytechnique fédérale de Lausanne (EPFL, the Swiss Federal Institute of Technology in Lausanne, Switzerland). "They require only a few drops of sugar to fly for hours, and they can handle many situations that are tricky for current robots, such as

wind, collisions, passing through small gaps, and switching between different modes of locomotion; for example, walking and flying."

Researchers working in the cyborg insect space acknowledge that fusing live insects with tiny electronic components has its limitations.

For one, bees "cannot carry a heavy backpack equipped with a large array of sensors," says Barani Raman, Dennis & Barbara Kessler Career Development Associate Professor of biomedical engineering at Washington University in St. Louis, MO, USA.

Another concern is the ethical implication of dragooning insects and other animals into scientific experiments. "Several projects in this field—for example, to create remote-controlled insects—are quite harmful to animals, with different types of lesions and surgical interventions," Ijspeert

says. "I personally feel uneasy about them, and think that any type of research involving animal-robot interfaces—even insects—should involve careful societal and ethical discussions before moving forward."

Yet researchers working with electronically enhanced bugs see a bright future ahead.

Says NCSU's Bozkurt, "We are living in the Internet of Things era and we have been observing an exponential growth in miniaturized low-power electronics and micro-scale wireless connections. This is leading to several new ways for us engineers working with biologists to collaborate with insects in novel ways in order to solve several real-life problems, from environmental pollution to search-and-rescue after natural disasters."

—Joe Dysart is an Internet speaker and business consultant based in Manhattan, NY, USA.

els, which provide a way of expressing quantum-computing operations using notation similar to classical logic.

As with previous methods, the computations in Broadbent's technique are encrypted. However, in place of the traps are what she calls 'gadgets'. These are computational elements made from standard quantum gates placed in the circuit. The gadgets support testing, but do not alter computations if a test is not active.

Used in combination with encryption, quantum computers cannot determine whether the additional gadgets are part of the computation or the test. In one form of Broadbent's protocol, the verifier can take advantage of quantum behavior to decide whether a run was a test or actual computation after the event, instead of specifying this upfront.

"Some of the circuits are larger than what you would want for efficiency, but the overhead is very reasonable in this model: performing a series of tests is pretty cheap," Broadbent says.

A common factor in almost all the protocols developed so far is that they require the verifier's own machine be able to prepare or measure quantum states—or do both. But this raises the possibility of a hacker corrupting the device used to prepare quantum states before the protocol even starts. That might be exploited to trick the verifier into accepting incorrect results.

Gheorghiu and Kashefi worked with Petros Wallden, also based at the University of Edinburgh, to propose a device-independent verification protocol that makes it possible to operate without the need to trust any of the quantum devices that are needed to run the tests. In doing so, they called on ideas developed by Ben Reichardt of the University of Southern California working with Falk Unger and Umesh Vazirani, who were based at the University of California at Berkeley. They found it is possible to avoid the issue of preparing and measuring quantum states if the quantum computers being used for the application and proving functions are far apart. The distance is important, because it introduces a communications barrier enforced by the speed of light. The protocol oper-

"Experiments can tell us what kind of noise we can expect for different kinds of implementations, and that, in turn, can tell us how to design our protocols to cope with it."

ates in such a way that the machines cannot collude with each other effectively to fake results. With the help of this requirement, Gheorghiu and colleagues found this property makes it possible for a verifier to avoid handling quantum states at all.

"In most situations we would, ideally, like the verifier to be classical," Gheorghiu says, because most users will only have access to remote quantum computers over the conventional Internet and not a network that supports quantum communications.

Last year, UC Berkeley Ph.D. student Urmila Mahadev found another approach to enabling a fully classical machine to be used by the verifier without demanding that the prover and application quantum computers are separated by a long distance. Conceptually similar to UBQC, Mahadev's protocol lays traps that expose misbehavior. To enforce blindness, she used a post-quantum cryptography scheme on the assumption that a quantum computer will be unable to crack it except through brute force. This computational assumption could prove false in the long term: although none are known today, a quantum algorithm may be found that efficiently breaks the encryption and destroys the property of blindness.

"Whether verification can be performed with a classical client and a single quantum prover and no computational assumptions remains open," Gheorghiu says. "On the other hand, it is true that one can perform verification without any computational as-

sumptions provided the verifier is not completely classical."

In the current version of Mahadev's protocol, the overhead of verification is higher than with those that make use of quantum-enabled verifiers though she and other researchers say they believe optimizations will improve efficiency. Gheorghiu says the lowest overhead may only come with quantum-enabled verifiers. And the need for fault tolerance for real-world computers may make it difficult to employ a fully classical verifier.

The incremental-proof algorithms developed so far are not expected to work well with today's experimental machines, which provide noisy results. Broadbent says: "The models that we are talking about right now are quite paranoid. Even small noise levels would not work in the sense that the protocols would detect any deviation as malicious. But others are possible where the errors that occur we can assume are not maliciously chosen."

The nature of the errors that near-term machines generate will, in turn, inform the development of practical verification techniques. Gheorghiu says: "Experiments can tell us what kind of noise we can expect for different kinds of implementations and that, in turn, can tell us how to design our protocols to cope with it." □

Further Reading

Gheorghiu, A., Kapourniotis, T., and Kashefi, E. Verification of Quantum Computation: An Overview of Existing Approaches
Theory of Computing Systems (2018), pp. 1–94. DOI: 10.1007/s00224-018-9872-3

Broadbent, A. How to Verify a Quantum Computation
Theory of Computing, Vol. 14(11), 2018, pp. 1–37.

Mahadev, U. Classical Verification of Quantum Computations
59th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2018), October 2018.
arXiv:1804.01082

Aharanov, D., Ben-Or, M., and Eban, E. Interactive Proofs for Quantum Computations
arXiv preprint (2008), arXiv:0810.5375

Chris Edwards is a Surrey, U.K.-based writer who reports on electronics, IT, and synthetic biology.

Code Talkers

Using voice input to write programs.

WHEN TAVIS RUDD decided to build a system that would allow him to write computer code using his voice, he was driven by necessity.

In 2010, he tore his rotator cuff while rock-climbing, forcing him to quit climbing while the injury healed. Rather than sitting idle, he poured more of his energy into his work as a self-employed computer programmer. “I’d get in the zone and just go for hours,” he says. Whether it was the increased time pounding away at a keyboard or the lack of other exercise, Rudd eventually developed a repetitive strain injury (RSI) that caused his outer fingers to go numb and cold, leaving him unable to type/code without pain.

Worried that he would not be able to do his job, Rudd turned to Dragon Naturally Speaking voice recognition software to see if that could help. He quickly discovered that he could insert commands into Dragon using the programming language Python, and that he could use the Python-based application programming interface (API) Dragonfly to create lists of words and link them to specific actions he wanted Dragon to perform.

So he set about creating such a list, known as a grammar, of words that would cause a text editor such as Emacs to take certain actions—insert or delete characters, add a bracket, move the cursor up some number of lines. He created this grammar with strange words, such as *ak* or *par*, to avoid confusing the speech recognition software with common English words and to keep the number of syllables per command down to one or two, so programming this way would be speedy.

It took two or three months to develop a customized grammar, learn how to work with it and what it was capable of, and figure out just which bits of automation he needed. He ended up with a Python file that was about 2,000 lines long and contained about 1,500

commands. Though the end result works well, he says, it is not a program that someone could just download and start using; he calls it a bit “duct-tapey.” “I think if someone wanted to get the basics, it would take them maybe three weeks to a month to get going, but they’d also need to know how to interface that with their editor.”

Back on the Shelf

Rudd’s RSI has cleared up, and his job has changed to include less coding and more management; he’s now principal engineer at Unbounce, a digital marketing agency in Vancouver, Canada. Because of those changes, he no longer uses his system and has not updated it in about five years. He had planned to publish the code, but realized it would take too much time to clean it up.

Indeed, several voice coding projects have followed a similar trajectory. Programmers discover that they have too much pain to type, so they build their own idiosyncratic voice coding system, based on Dragon or Windows Speech Recognition (WSR). They share it with a wider community but never develop it to an easy-to-use, out-of-the-box solution, then move onto other things after a while.

That was the case with Gustav Wengel, a programmer in Denmark who co-founded two startups: the software consultancy Bambuu, and Reccoon, which

is attempting to give consumers a way to digitally track how much waste they generate. In 2016, Wengel developed pain and tingling in his little finger that ran all the way up to his elbow, a complaint often called “Emacs pinkie,” though a doctor would probably diagnose it as cubital tunnel syndrome, in which the ulnar nerve that runs along the arm is compressed or stretched.

He tried switching to an ergonomic keyboard, which reduced the pain but didn’t end it, then tried voice coding with Dragon. His plan was to build the skeleton of a voice control system by keyboard, then continue developing it with his voice. But after a couple of months, the pain began to diminish. “My hands became better and better, and the voice coding just wasn’t worth it anymore,” he says.

Voice coding software, Wengel says, is difficult to develop, difficult to learn, and difficult to use. “A lot of people use it out of a sense of desperation because that’s all they’ve got,” he says.

There are several challenges to creating voice coding software. One is the need to rely on voice recognition engines, which are not optimized for the task. Dragon, owned by Nuance Communications of Burlington, MA, is primarily focused on transcribing speech, based on models of natural language built up through years of machine learn-



ing. That does not really address the needs of programmers. "When you're coding, you're not speaking in sentences." Rudd says. "You don't have a language model that you can leverage."

Furthermore, there's a lot more to programming than simply dictating code. Programmers have to be able to move around within the code, manipulate multiple windows, go to different Web pages, and test and debug their code. "Really, writing code hands-free is a subset of the problem of generally controlling a computer hands-free," says Rick Mohr, a software developer for Azavea in Philadelphia, PA, which creates geospatial Web applications. Mohr created Vocola, a voice coding program he still maintains, after he developed RSI in 2007. Vocola comes in two versions, one using Dragon and the other using Windows Speech Recognition.

A Different Input

Desktop computers were designed to take input from a keyboard and a mouse, and accessibility was an afterthought, Mohr says. He argues that voice is actually a superior way of controlling computers, if only they were designed to make that easy. Speaking, for instance, is much faster than typing, and there are quicker ways to move a cursor than with a mouse. "Anybody who has been forced to climb the learning curve to use a computer hands-free discovers there are many things that are actually more efficient than using their hands," he says. "Mostly the only people that have been willing to climb the curve are those that need to."

Many interfaces require users to click on a button with a mouse, and doing that by voice is difficult, says Ryan Hileman, a software engineer in Mountain View, CA, who designed and is continuing to develop the Mac-based voice coding system Talon. To overcome the mouse problem, he incorporates an eye tracker and noise recognition. The eye tracker allows a user to move a cursor across the screen very quickly, though it does not provide pixel-level accuracy because eyes tend to flicker. A head tracker lets the user refine the location of the cursor, and making a popping sound with his mouth provides the click. Hileman also can drag something across the screen by letting out a long hiss, "which is kind of silly, but it works," he says.

Hileman has been working on Talon full-time since quitting his job in August 2017. He's supporting himself through savings, and collects around \$1,000 per month through the website Patreon to fund his development of Talon. He gives away the software for free, but will not release the code as open source so he can maintain control of how it is developed. He wants to create a plug-in system that will allow users to create their own commands and have them mesh with those that already exist.

There are other voice coding projects in existence, such as Aenea, which runs Dragon in a virtual machine, and Caster, a collection of tools that run on top of Dragonfly. One older project is voice-code.io, a Mac-based coding platform developed by Ben Meyer, which sold for \$300. "He hasn't supported it in years," Hileman says. "You can pick it up and get started and try to use it, but it seems a little janky to me. It doesn't seem polished at all." Another system, also called Voicecode, was launched by the National Research Council of Canada in 1999, but went defunct several years ago.

One fear these developers share is that the underlying software their programs rely on will stop working. The platforms are built on top of Dragon or WSR. The programmers gain access to Dragon through a backdoor interface, NatLink, installed by Dragon's original developers, but Nuance does not support it. Quintijn Hoogenboom, a Dutch software developer, and other enthusiasts try to maintain NatLink. "It gets slightly more crippled with every

release of Dragon, but people have always found a way to keep it going," says Mohr, adding there may come a day when an update breaks it irrevocably.

WSR was developed as part of Windows Vista in the early 2000s, and while it still works, Mohr says there is no guarantee that either it or Natlink will continue. "Either of those could disappear at any time."

Rudd believes coding by voice will become much easier as coding methods in general evolve. "The type of coding that our industry has been doing has been quite low-level. We've been very syntax-focused," he says.

With advances in artificial intelligence and natural language processing, Rudd thinks programming will become less mechanistic. Instead of telling a computer, line by line, how to achieve a result, a programmer will tell it what he/she wants to accomplish, and the machine will search through libraries of functions to find the best way to obtain that result. "I think when that happens, the voice systems that are available for Google Now and Siri and that sort of stuff will be much more suitable for coding in that style," Rudd says.

However programming evolves, Hileman says, it is important that voice be an option for coders. "I need it. It's very important to me," he says. "I can't type sustainably, so if I want to be able to use computers—which I'm very passionate about—I need something like this." □

Further Reading

Hauthorn, C., Reinhold, J., Wengel, G., and Roswall, S. Using Consumer Electronics to Enhance the Experience of Developing Software With Additional Input Modalities; Aarhus Universitet Bachelor Project 2016

Wengel, C. State of Voice Coding – 2017, <https://medium.com/bambuu/state-of-voice-coding-2017-3d2ff41c5015>

Nowogrodzki, A. Speaking in Code: How to Program By Voice, *Nature*, 559, 2018, <https://www.nature.com/articles/d41586-018-05588-x>

Rudd, T. Using Python to Code By Voice, <https://www.youtube.com/watch?v=8SkdfdXWYAI>

Neil Savage is a science and technology writer based in Lowell, MA, USA.

© 2019 ACM 0001-0782/19/5 \$15.00

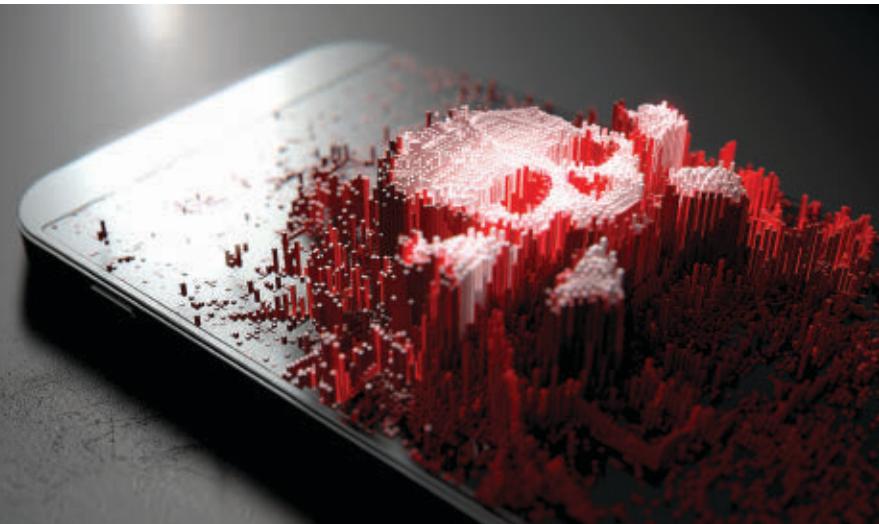
Deep Insecurities: The Internet of Things Shifts Technology Risk

A more connected world sounds alluring, but without better protections, the Internet of Things could lead to disaster.

IT IS HUMAN nature to view technology as a path to a better world. When engineers and designers create devices, machines, and systems, the underlying premise is to deliver benefits. The Internet of Things (IoT) is certainly no exception. Smartphones, connected cars, automated thermostats, smart lighting, connected health trackers, and remote medical devices have made it possible to accomplish things that once seemed impossible. Everything from toothbrushes to tape measures are getting “smart.”

However, at the center of the tens of billions of connected devices streaming and sharing data lies a vexing problem: cybersecurity. It is no secret that hackers and attackers have broken into baby monitors, Web cameras, automobiles, lighting systems, and medical devices. In the future, it is not unreasonable to assume that cybercriminals could take control of a private citizen’s refrigerator or lighting system and demand a \$1,000 ransom in bitcoin in order to restore functionality. It is also not difficult to fathom the threat of a vehicle that won’t brake, or a pacemaker that stops working due to a hack. Hackers might also weaponize devices and take down financial systems and power grids.

The thought is chilling, and the repercussions potentially far-reaching. “All these devices, which now have computing functionality, affect the world in a direct physical manner—and that just changes everything,” observes Bruce Schneier, an independent computer security analyst and author of *Click Here to Kill Everybody: Security and Survival in a Hyper-connected World* (W. W. Norton & Company, 2018). “Today, computers can actually kill you.”



Adds Stuart Madnick, John Norris Maguire Professor of Information Technologies at the Massachusetts Institute of Technology (MIT) Sloan School of Management, “We are entering a dangerous period. We have to wake up to the risks.”

Dangerous Liaisons

What makes the IoT so powerful—and so dangerous—is the fact that devices and data now interconnect across vast ecosystems of sensors, chips, devices, machines, and software. This makes it possible to control and manipulate systems in ways that were never intended.

For example, in December 2015, a massive cyberattack shut down the power grid in the Ukraine. An estimated 230,000 people were left without electricity for a period lasting from one to six hours after hackers compromised systems at three energy distribution companies. In 2018, researchers at Upstream Security Ltd. found that attacks on connected cars had increased six-fold over a four-year period. Hackers

entered through servers, keyless entry systems, onboard diagnostics ports, infotainment systems, mobile apps, wireless connections, sensors, and more.

The stakes continue to grow. In February 2018, security researchers Billy Rios and Jonathan Butts of security consulting firm WhiteScope LLC discovered a vulnerability in the Medtronic CareLink 2090 portable computer system, which uses wireless telemetry and over-the-air programming to control pacemakers and oversee home monitors for cardiac patients. “Someone could sit at a Starbucks and commandeer the pacemaker programmer,” Rios says. “They could alter the way the device works and potentially impact a patient’s safety.” Medtronic, which is subject to U.S. Food and Drug Administration (FDA) regulatory oversight, wound up issuing a recall for the device so it could patch the vulnerability.

Rios, one of the world’s top ethical hackers, says the level of security embedded in most devices is woefully inadequate and the situation shows no

sign of improving. "A huge problem is that computing devices were originally designed to operate in a closed system. The underlying hardware and software weren't designed for a connected world. Devices and communications systems are built on intrinsically insecure protocols. Now we have all these devices that are using these protocols to connect cars, trains, airplanes, and more. You cannot flip a switch or make a small code change and protect these systems."

A cyber-9/11 event, on par with the physical attacks on the U.S. that occurred in 2001, is not outside the realm of possibility, even if some of the concepts seem like science fiction today, experts say. Privacy is yet another concern because today's devices incorporate data recorders and logs, GPS connectivity, health and fitness data, and detailed information about how people live and move about. Already, an interactive toy doll named *My Friend Cayla* has been banned in Germany because it surreptitiously recorded conversations and stored them unprotected on the Internet.

Says Benson Chan, senior partner at consulting firm Strategy of Things, "In a hyperconnected world, the biggest security risk is that there's a lack of transparency. You just have millions of machines talking to each other, making decisions and taking action autonomously based on machine learning algorithms. There is very limited visibility about what data is passed through and how decisions are made."

Finding a Fix

At the center of this frightening scenario is a simple but profound fact: the technical resources largely exist to address the risk of a hyperconnected world, but the political, economic, and social impetus is lagging. "The fundamental problem is that companies are interested in getting products to market quickly. The market does not reward security," Schneier says. Adds Madnick: "It has become apparent that we cannot rely on manufacturers and vendors to consistently include the essential security protections."

Addressing the lack of security and privacy in IoT devices won't be easy. What's more, the repercussions could have enormous impact on product

Security experts say a two-pronged approach is needed: manufacturers must change how they build their products, and new laws must support security and privacy.

liability, particularly if injuries and deaths occur. Ultimately, IoT security experts say a two-pronged approach is required. It's critical to change the way manufacturers build products, but also to introduce regulations and laws that support security and privacy. Business leaders must recognize there are advantages to building a more secure IoT, Madnick says. "At some point—and it's already something that some companies understand—it is much easier and less expensive to build security into every aspect of a product than it is to reverse-engineer it later on."

Enhancing industry security and privacy standards for the IoT is gaining momentum. An industry body, the Online Trust Alliance (OTA), is spearheading efforts to establish security and privacy by design. A "Security by Design" framework has also been promoted by the Open Web Application Security Project (OWASP), which focuses on a multipronged approach: minimizing attack surfaces, establishing secure default settings, adopting the principle of least privilege (granting users only the minimum access they require to accomplish a task), adopting a defense in depth framework (which features a layered series of defensive mechanisms), embracing a zero-trust model, fixing security flaws promptly and correctly, and several other tactics.

Schneier says such a framework must encompass several elements. These include vendor transparency about how products work, software that is patchable, extensive preproduction testing, security out of the box, the

ACM Member News

INTEREST IN ROBOTS LEADS TO R&D FOR SELF-DRIVING CARS



"When I was a kid, I was very interested in reading science fiction about robots," says Li Erran Li, adding

that this early interest would influence the future trajectory of his education and career. Today, Li is chief scientist at Pony.ai, a start-up developing autonomous driving technology.

Li earned his undergraduate degree in automatic control from the Beijing University of Technology, and his master's degree in computer vision from Beijing's Chinese Academy of Sciences. He then came to the U.S. and obtained his Ph.D. in computer science from Cornell University.

Before joining Pony.ai in May 2018, Li was part of the perception team at Uber's Advanced Technology Group, as well as working with that company's machine learning platform team. Prior to Uber, Li spent 14 years working for Bell Laboratories.

Li's current research is primarily focused on machine learning, computer vision, and learning-based robotics, and their applications to autonomous driving.

As chief scientist at Pony.ai, Li leads the company's research efforts. He also serves as vice dean of the Pony.ai Research Institute in Guangzhou, China, where computer scientist and computational theorist Andrew Yao, who received the 2000 ACM A.M. Turing Award, is the honorary dean.

Li says he is passionate about pushing the frontiers of artificial intelligence for autonomous driving, and is active in the machine learning and computer vision communities. He often provides tutorials and organizes workshops on machine learning for autonomous driving at academic conferences, such as the International Conference on Machine Learning (ICML) and the Conference on Neural Information Processing Systems (NIPS).

—John Delaney

ability for systems to fail predictably and safely, the use of standard protocols, the ability to preserve offline functionality, and widespread encryption and authentication of data.

He also believes vendors should support responsible security research. Such security research is severely lacking. The IoT Security Foundation (IoTSF) reported in August 2018 that only 9.7% of companies making IoT products have a public disclosure policy that allows researchers to probe known vulnerabilities.

This voluntary and lackadaisical approach has prompted many, including Schneier, to call for increased industry and government regulation. He would like to see broader and expanded regulations on products and product categories, a licensing system for professionals and products, more stringent testing and certification requirements, and more widespread adoption of industry best practices. This framework would include tax breaks for businesses that do things right, and rules that punish negligence and bad behavior.

Rules and Regulations

Although several past attempts to institute regulations and laws in the U.S. have failed, the landscape is changing. Massachusetts, New York, and California have all stepped up efforts to punish companies for data breaches and similar abuses. In August 2018, California took the boldest step forward when it adopted an IoT law, SB327, which establishes baseline security standards for IoT devices. The law, though intentionally vague, mandates that IoT device manufacturers must equip their products with “reasonable” security features to address wide-ranging issues such as authentication, device use, modification, and destruction. It will go into effect January 1, 2020.

The idea of regulating IoT devices is also gaining momentum elsewhere. For example, in Indonesia, the federal government is finalizing regulations that standardize the use of IoT devices, though the goal is primarily to create a framework for business. Japan, Canada, Mexico, Australia, and other countries also have addressed data governance through regulations, though most countries have not yet established a formal IoT regulatory framework. Ac-

**Says Chan,
“In the end,
the biggest danger
isn’t a device failing
or a grid shutting
down; it’s a loss of
trust in technology.”**

cording to a 2015 study conducted by consulting firm Deloitte, the trend is toward greater regulation for electronic systems, including the IoT. The number of privacy laws has grown from 20 in the 1990s to more than 100 today.

Meanwhile, organizations such as EPIC (the Electronic Privacy Information Center) are stepping up lobbying efforts for the U.S. and Europe to adopt more stringent IoT cybersecurity and privacy standards. Europe, which has emerged as perhaps the most powerful regulatory entity in the world, adopted the General Data Protection Regulation (GDPR) in May 2018. It imposes standards for data use and sharing, along with sizeable fines for non-compliance. Although GDPR doesn’t specifically pertain to the IoT, connected devices play a critical role in the regulatory framework. In addition, the EU is finalizing the ePrivacy regulation, which addresses the use of personal data through entities such as Facebook, SnapChat, and the Web, along with smartphones and other IoT devices.

Ultimately, any single approach is likely to fail, Schneier says; “None of them will work in isolation.” Minimum security standards alone won’t solve the underlying problem, he says; what is needed is a series of mutually reinforcing policies that can slide the dial to greater safety and security.

A Matter of Trust

The clock is ticking, Rios says. “As we move toward a far more connected world, the risk grows. You can’t simply reboot a system and put a train back on the right track, or help a patient that has died because a medical device has failed.” Moreover, as the IoT ripples across devices and systems, entire cit-

ies and groups will likely be affected. “Right now, manufacturers are not rewarded for building cybersecurity into products, but they can definitely be punished. We need to move toward a societal model where they are both rewarded and punished. We need to rethink and revamp the entire framework by which we create, manage, and use IoT devices,” Rios says.

Chan believes it is crucial to think about IoT devices as more than individual components that can be hacked and manipulated. As these devices become more pervasive and entrenched in business and life, everything connecting to them—including data and algorithms—become potential targets for manipulation and abuse. Says Chan, “It’s only a matter of time until we see ransomware, attacks on devices and connected networks, and perhaps even a cyber-9/11 event.

“But, in the end, the biggest danger isn’t a device failing or a grid shutting down; it’s a loss of trust in technology. If you can’t trust devices to operate correctly and safely, then you won’t use them ... when that happens, our world will be a very different place.” C

Further Reading

Schneier, B.

Click Here to Kill Everybody: Security and Survival in a Hyper-connected World.
W. W. Norton & Company (September 4, 2018).

Conti, M., Dehghantanha, A., Franke, K., and Watson, S.

Internet of Things Security and Forensics: Challenges and Opportunities. Future Generation Computer Systems, Volume 78, Part 2, January 2018, pp. 544–546. <https://www.sciencedirect.com/science/article/pii/S0167739X17316667>

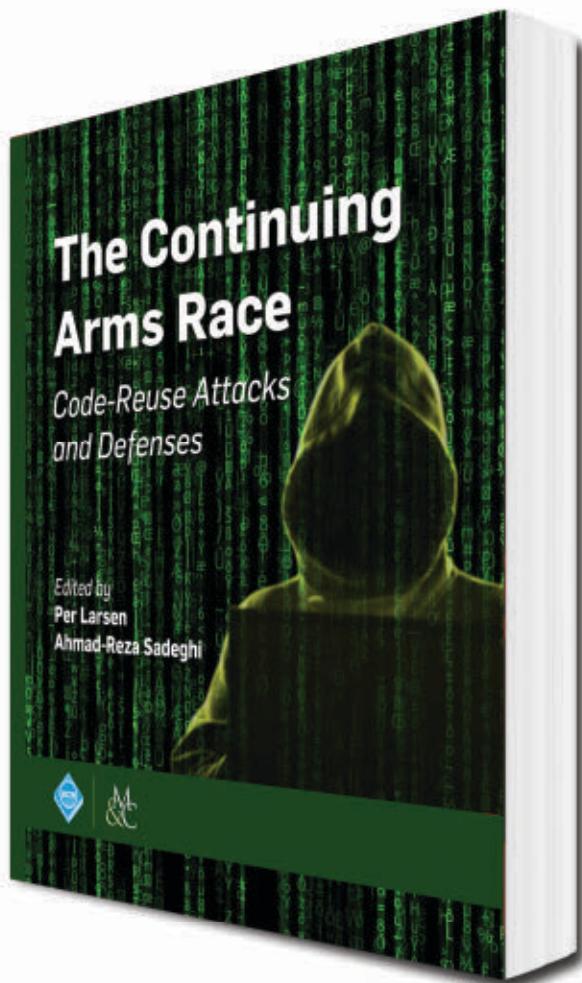
Alaba, F.A., Othman, M., Hashem, A.T., and Alotabi, F.

Internet of Things Security: A Survey. Journal of Network and Computer Applications, Volume 88, 15 June 2017, pp. 10–28. <https://www.sciencedirect.com/science/article/pii/S1084804517301455>

Mosenia, A. and Jha, N.K.

A Comprehensive Study of Security of Internet-of-Things. IEEE Transactions on Emerging Topics in Computing, Volume: 5, Issue: 4, Oct.–Dec. 1 2017, pp. 586–602. <https://ieeexplore.ieee.org/abstract/document/7562568>

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



There is no silver bullet...there IS information you can USE.

EDITED BY

Per Larsen, *Immunant, Inc.*

Ahmad-Reza Sadeghi, *Technische Universität Darmstadt*

As human activities moved to the digital domain, so did all the well-known malicious behaviors including fraud, theft, and other trickery. There is no silver bullet, and each security threat calls for a specific answer. One specific threat is that applications accept malformed inputs, and in many cases it is possible to craft inputs that let an intruder take full control over the target computer system. The nature of systems programming languages lies at the heart of the problem. Rather than rewriting decades of well-tested functionality, this book examines ways to live with the (programming) sins of the past while shoring up security in the most efficient manner possible. We explore a range of different options, each making significant progress towards securing legacy programs from malicious inputs. This book provides readers with some of the most influential works on run-time exploits and defenses.



ISBN: 978-1-970001-80-8 DOI: 10.1145/3129743
<http://books.acm.org>
<http://www.morganclaypoolpublishers.com/acm>

DOI:10.1145/3319077

James Grimmelmann

Law and Technology Continuity and Change in Internet Law

The fundamentals of the field of Internet law have remained consistent, but details have evolved in response to technological innovation.

THIS IS MY first column as editor for *Communications'* Law and Technology column. I am taking over from the very capable Stefan Bechtold, who established the column in its current form and imbued it with his high standards of rigor, relevance, and readability. I thought I might mark this transition with some historical reflections on how the field of Internet law has changed over the last few decades, and what has stayed the same.

Start with the continuity. The basic issues around intellectual property rights in software have been the same for a very long time. In 2014, the U.S. Supreme Court expressed serious skepticism about patents to "do X on a computer" and a federal appeals court allowed Oracle to assert copyright in the Java APIs. Neither issue is new. The Supreme Court was just as skeptical about software patents in 1972 and 1978, and a different federal appeals court held in 1995 that Lotus 1-2-3's macro interface was uncopyrightable.

Modern encryption controversies

would look very familiar to a 1990s technology policy wonk who lived through the Clinton Administration's failed attempt to impose a key escrow scheme that would have enabled government wiretapping of encrypted communications. Can the government force hardware vendors to make unlockable devices? Can criminal suspects be forced to disclose their passwords? Do the police need a warrant to search a computer? Can government hackers break into computers remotely? All of these controversies are in the headlines again.

Similarly, today's legal disputes over network neutrality reflect the definitions Congress used in the Telecommunications Act of 1996. While Congress didn't quite anticipate the Internet, the distinction it drew between "telecommunications" and "information" services was rooted in previous regulation of early pre-Internet online services and in many decades of telephone regulation. Today's networking technology is new, but the debates over networks, monopoly, and nondiscrimination are not.

Everything old is also new again with cryptocurrencies. People have hoped or feared for years that strong cryptography and a global network would make it impossible for governments to control the flow of money. There is a direct line from 1990s-era cypherpunk crypto-anarchism and experiments with digital cash to Bitcoin and blockchains. The regulatory disputes are almost exactly the ones that technologists and lawyers anticipated two decades ago. They just took a little longer to arrive than expected.

In other ways, things look very different today. One dominant idea of the early days of Internet law was that the Internet was a genuinely new place free from government power. As John Perry Barlow wrote in his famous 1996 "Declaration of the Independence of Cyberspace": "Governments of the Industrial World, you weary giants of flesh and steel, I come from Cyberspace, the new home of Mind. ... You have no sovereignty where we gather. ... Cyberspace does not lie within your borders."

If there was a moment that this



Matrix-esque vision was definitively unplugged, it was probably the 2003 decision in *Intel v. Hamidi*. Intel tried to argue that its email servers were a virtual, inviolate space—so that a disgruntled ex-employee who sent email messages to current employees was engaged in the equivalent of breaking into Intel buildings and hijacking its mail carts. The court had no interest in the cyber-spatial metaphor. Instead, it focused on more down-to-earth matters: Intel's servers were not damaged or knocked offline.

“Cyberspace” turned out not to be a good description of how people use the Internet or what they want from it. Most Internet lawsuits involve familiar real-world problems—ugly divorces, workplace harassment, frauds and scams, and an endless parade of drug deals—that have spilled over onto cellphones, Facebook pages, and other digital platforms.

Internet law has fully embraced the idea that the Internet matters, not because it is somewhere new for people to go, but because it is everywhere that people already are. Courts have held

that websites are “places of public accommodation” that must be made accessible to the disabled, just as physical stores are. And local regulators are mostly winning their claims that sharing-economy companies like Uber, Airbnb, and Bird are operating in their cities and must comply with zoning and licensing laws.

Indeed, in the story of governments versus the Internet, governments seem to have the upper hand for now. The Securities and Exchange Commission regularly shuts down fraudulent or unregistered initial coin offerings. The European Union is increasingly confident in its ability to regulate the Internet to protect its vision of its citizens’ welfare and the common good, as with its recently enacted privacy law, the General Data Protection Regulation. And China has quite successfully imposed extensive filtering and surveillance on its domestic portions of the Internet.

A second shift in Internet law is the waning of the file-sharing wars. The battle lines were drawn in the 1990s, with a series of policy battles that cul-

minated in the U.S. with the passage of the Digital Millennium Copyright Act of 1998 (DMCA). Section 512 of the DMCA created a “notice and take-down” system under which content hosts are not liable for infringing user uploads—but only so long as they respond “expeditiously to remove” those uploads when they receive notice from the copyright owner. Section 1201 of the DMCA made it illegal to disable digital rights management (DRM) technology that limits access to copyrighted works. Both were deeply controversial.

Fighting broke out in earnest in 1999 when numerous record companies sued Napster, eventually forcing it to shut down. Movie studios, photographers, book publishers, and other copyright owners filed lawsuits against file-sharing services, Web hosts, hardware makers, search engines, video-game modders, and the creators of DRM-removing software—as well as against less likely targets like replacement toner cartridges and third-party garage-door openers. And this is to say nothing of the many thousands of

suits against individual uploaders and downloaders (often filed in the hope of extracting a quick settlement).

The initial ferocity of these disputes has faded. There are still some large copyright lawsuits, and some raising major legal issues. (The record company BMG's suit against Cox Communications, an Internet service provider, for not cutting off service to copyright infringers, is an example of both.) There is, however, less of a sense that the future of either content creation or technological innovation is at stake. Instead, the Internet has settled into an uneasy detente: many copyright owners and technologists have moved on to other fights.

One reason is that the basic legal compromises in the DMCA have proven surprisingly durable. Copyright owners have not been able to force content hosts to do significantly more than the notice-and-takedown rules of Section 512 require (Viacom lost its lawsuit against YouTube on this point), but they have generally been able to keep them from doing significantly less, either. "Graduated response" or "three strikes" schemes, which would force ISPs to cut off service to unrepentant infringers, have been tried around the world and have mostly failed, but voluntary algorithmic filtering of uploads, like YouTube's ContentID, may be here to stay.

Another reason is that the courts have also been increasingly aware of the value created by innovative digital uses of media. The Authors Guild's suits against Google and its library partners ended with resounding judicial declarations that scanning books to make them searchable and accessible to the blind is protected as legal "fair use," opening the door to large-scale machine learning using copyrighted works. Search engines, plagiarism checkers, video remixers, and meme-makers have generally been blessed by the courts. Although the creators of second-generation decentralized file-sharing services like Grokster and Morpheus were successfully sued for inducing users to infringe, BitTorrent has not met a similar fate.

And finally, the rise of downloadable media and subscription streaming services has created a new and apparently stable revenue stream. Even

Copyright owners no longer fear they must hunt down every last infringing upload: they usually focus their attention on the most egregious cases.

where pirated alternatives are readily available, many people seem perfectly content to pay for Hulu and Spotify subscriptions. Copyright owners no longer fear they must hunt down every last infringing upload: they usually focus their attention on the most egregious cases.

In both of these domains, technology policy has gone from alarm to acceptance. With jurisdiction, society asserted its control over the Internet; with copyright, society learned to live with it. In a third domain, however, the trend is in the other direction: from comfort to concern.

Early online-speech fights were about governments' ham-handed attempts to limit access to pornography. For example the Communications Decency Act (CDA) of 1996, which made it illegal to post "indecent" but legal-for-adults material anywhere online that a child could see it, was obviously unconstitutional. The Supreme Court struck it down in 1997.

The CDA also contained an immunity, Section 230, for Internet intermediaries. Unlike Section 512 for copyright, which applies only if the intermediary responds to takedown requests, Section 230 is nearly absolute. Intermediaries are immune if they leave up harmful content; they are immune if they take it down.

Section 230 was justified in terms of giving websites, search engines, and other such intermediaries the ability to be "good Samaritans" in developing their own content policies.

Without it, they feared that if they made any attempt to enforce policies of truthfulness, decency, or community standards, they would be tagged and held liable for all of the harmful content they did not catch. Early cases showed Section 230's obvious value in enabling platforms like AOL and MySpace to host a huge range of user-generated content without the fear of crippling liability.

In the years since, many lawyers have come to think that Section 230 goes too far. In their view, the absolute immunity gives websites too little incentive to care when bad actors weaponize their platforms. They think, for example, that Twitter might do a better job of preventing neo-Nazis from making death threats against Jewish users if it faced any legal consequences for failing to respond. Other sites, like 4chan and Gab, have been accused of affirmatively fostering toxic cultures in which harmful and blatantly illegal conspiracies are birthed and allowed to grow. Proponents of Section 230 respond that with a weaker immunity, platforms might go to the other extreme, taking down users' speech at the slightest suggestion of controversy.

These debates are mirrored in other debates about free speech online. What counts as a "threat" of harm when users are separated by thousands of miles and the speaker is pseudonymous? Is a coordinated campaign of nasty tweets actionable harassment? How should bullying laws and disciplinary policies developed to deal with the schoolyard be adapted to social media?

Online speech law, which previously embodied a confident pro-speech consensus that the Internet was all bark and no bite, is going through a distinct crisis of faith. Harassment and abuse have become inescapable parts of online life, particularly for women and members of vulnerable groups. It is not yet clear what path forward the legal system will take, but online speech is becoming one of the defining legal issues of our time. □

James Grimmelmann (james.grimmelmann@cornell.edu) is a law professor at Cornell Tech and Cornell Law School, New York, NY, USA.

Copyright held by author.

DOI:10.1145/3319079

Joan Feigenbaum

► Carl Landwehr, Column Editor

Privacy and Security Encryption and Surveillance

Why the law-enforcement access question will not just go away.

IS THE INCREASING use of encryption an impediment in the fight against crime or an essential tool in the defense of personal privacy, intellectual property, and computer security? On the one hand, law-enforcement (LE) agencies complain about “going dark.” On the other hand, computer-security experts warn that forcing law-enforcement access (LEA) features into devices or protocols would impose high costs and create unacceptable risks.

This argument echoes the 1990s “crypto war” about whether strong encryption technology that had been tightly regulated during the Cold War should only have been deregulated if vendors provided “key-escrow” features that prevented criminals from using it with impunity. The opponents of key escrow won that war by convincing the government that key escrow was difficult to implement securely and that foreign competitors of U.S. technology companies could gain an advantage by assuring customers that no third parties would have access to their keys.

Calls for LEA have resurfaced, because, in the wake of the Snowden revelations, technology vendors have been pushing end-to-end encryption protocols deeper into the computing and communications infrastructure; in fact, some products and services are now built so that encryption is automatic and vendors themselves



cannot unlock devices or decrypt traffic unless the owner of the device provides the passcode. This can lead to LE agents’ being unable to access cleartext data even when they are fully authorized to do so, or, in more melodramatic terms, to their “going dark”; they have called for vendors to build in LEA features^a that enable access with

an appropriate warrant but *without* the owner’s passcode.

In this column, I first summarize some of the arguments that have been made for and against LEA and explain why I believe that LEA features should not be mandated *at this time*. I then argue that the question of whether some form of LEA is technically feasible and socially desirable is unlikely to go away and deserves further study.

Encryption and Surveillance as a Policy Question

Many cryptographers, computer-security researchers, and LE officials have chimed in on the LEA controversy. On

^a The term “exceptional access” is often used for this capability, but it connotes something broader in terms of both technical features and potential users. I have used the term “law-enforcement access” to emphasize that the scope of this column is the law-enforcement community’s call for the technical ability to access information when it has warrants.



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

acmmEDIASales@acm.org



one side is the LE view that the technology industry's post-Snowden embrace of default encryption is willfully thwarting the lawful exercise of properly authorized warrants. The FBI's motion to compel Apple to develop software to unlock the iPhone of a dead terrorist perfectly exemplifies this side of the debate.^b Under this view, the salient fact is that individuals and organizations are obligated, under the All Writs Act^c in the U.S. and similar laws in other democratic countries, to assist the government in the execution of warrants.⁴

On the other side is the view, embraced by many technologists and civil-liberties advocates, that, since the 9/11 terrorist attacks, governments have conducted far too much mass surveillance and that the appropriate grass-roots response is mass encryption. Moreover, widespread use of sound encryption is our strongest weapon in the fight against intellectual-property theft, identity theft, and many other online crimes—something that LE should applaud. As in the 1990s crypto war, customers of U.S. technology firms might be driven into the arms of foreign competitors in search of promises that their data will not be decrypted by third parties, even when those third parties are pursuing legitimate goals. This view is explicated and endorsed by, for example, Landau and Schneier in their individual statements in Zittran et al.⁹

Encryption and Surveillance as a Technical Question

That LEA is at best technically difficult and perhaps technically infeasible has been argued eloquently by numerous experts.^{1,5,9} While acknowledging that criminals can use encryption to avoid detection and prosecution and that increasing use of encryption hampers LE, these authors point out that the LE community has not quantified the extent of the problem. They explain that LE often has at its disposal other means

of obtaining the information it needs, for example, vulnerability-based unlocking toolkits or back-up copies that can be decrypted by cloud-service providers. Indeed, the FBI withdrew its motion to compel Apple to assist it in unlocking the dead San Bernardino terrorist's iPhone when it discovered a "gray-hat" hacking toolkit that could unlock the device. As reported in Bellovin et al.,³ the firm Grayshift "will sell law enforcement a \$15,000 tool that opens 300 locked phones or online access for \$30,000 to open as many phones as law enforcement has warrants for."

If LE wants something more general, more powerful, or more rigorously analyzed by the research community, it will need to specify precisely what its LEA requirements are. What range of surveillance tasks does it expect to accomplish in the presence of default encryption? How does it expect LEA technology to interact with legal processes and, in particular, would the technology be available to the more than 15,000 police departments in the U.S.? Would technology vendors be expected to cooperate on LEA not only with the U.S. government but with the governments of all countries in which their products are sold, including authoritarian governments (and, if not, what is to stop criminals from buying their devices in countries with which vendors do not cooperate)?

Notwithstanding the absence of fully fleshed-out requirements, several computer scientists have proposed

Widespread use of sound encryption is our strongest weapon in the fight against intellectual-property theft, identity theft, and many other online crimes.

b See <https://www.clearinghouse.net/detail.php?id=15497>.

c 28 USC 1651(a), 1789: "The Supreme Court and all courts established by Act of Congress may issue all writs necessary or appropriate in aid of their respective jurisdictions and agreeable to the usages and principles of law."

"solutions" to one version of the LEA problem, namely building devices that, in the absence of the device owner's passcode, can still be unlocked, usually with the manufacturer's cooperation, by LE agents who present valid warrants. I have put "solutions" in quotes, because these are often high-level ideas rather than completely specified proposals. The one that has received the most attention is that of Ray Ozzie,⁶ who is the inventor of Lotus Notes and a former Microsoft VP. In Ozzie's scheme, the device's encryption key is stored on the device itself, encrypted under a manufacturer's key. An LE agent who has physical possession of the device and a warrant to unlock it extracts the encrypted device key from the phone and sends it to the manufacturer. The manufacturer decrypts the device key and sends it back to LE, which can then unlock the device. A notable feature of Ozzie's approach is that, when the target device is unlocked, it also "bricks" itself, preventing any further changes to its contents. Bricking both preserves evidence for use in court and informs the owner that someone has unlocked his device so that he can power it down and put it in a safe place, thus preventing subsequent access.

Flaws were quickly found in Ozzie's scheme.³ Of course, early iterations of security protocols often have flaws that are fixed in later iterations. Whether Ozzie's basic approach can be developed into a fully specified, secure protocol remains to be seen. Other early-stage designs for LEA to locked devices were presented at a Crypto '18 workshop² and in related papers.^{7,8}

A Compromise Position

Although neither side in this debate can simply be dismissed, I find the call to implement LEA unconvincing at this time. There has indeed been too much surveillance since 9/11, and it is entirely reasonable for the technology industry to react by enabling its customers to keep data truly confidential. Rather than causing LE to "go dark," locked devices and default-encrypted communications appear to be causing it, in some cases, to use less convenient or more expensive methods than it would prefer to use. Imposition of as-yet-unquantified

Unlike purely technical problems, compelling policy problems are rarely definitively "solved" and no longer discussed.

inconvenience and expense on the LE community is not a good enough reason to mandate LEA features that might render immensely popular products and services less secure, more expensive, or unsellable on world markets.

However, I also believe LEA deserves further study. The desire of many in computer security and related communities for the LEA question to be declared "asked and answered" and simply go away is unrealistic. Unlike purely technical problems, compelling policy problems are rarely definitively "solved" and no longer discussed. The losing position in the 1990s crypto war is still appealing to many people, some of whom were not yet born when that war was won by the opponents of key escrow. If LEA is to be rejected, the argument must continue, and a new generation must be convinced.

My experience with Yale students has revealed two loci of resistance to the ideas that the tech community should not or cannot assist LE. The first is perceived arrogance of the technology industry. Government regulates many consumer products: why not smartphones or computers? The second is technical in nature: Many strong students do not see intuitively why it is infeasible to build personal devices that, in typical circumstances, can only be unlocked by their owners but, in atypical circumstances and with proper judicial authorization, can also be unlocked by a designated third party. If smartphone owners trust cloud-service providers to decrypt back-up copies only under appropriate circumstances, why is there no organization that can be

similarly trusted with the ability to unlock devices? Until clear answers to such questions are more widely disseminated, intuitive resistance to the claim that LEA is technically infeasible will continue.

Indeed, it has not actually been shown that no useful form of LEA can be implemented without creating unacceptable risk. We have heard convincing arguments that mandated LEA capabilities might be ineffective, extremely costly, or hijacked by the very criminals they were built to thwart. However, we have also heard that LE has not precisely specified its requirements. A cryptographic goal that cannot be met in its most general form is sometimes achievable in a weaker but still useful form. Perhaps the final verdict on LEA will be that it cannot be done securely at reasonable cost, but, in order to prove that, we will have to know exactly what the meaning of "it" is. □

References

- Abelson, H. et al. Keys under doormats: Mandating insecurity by requiring government access to all data and communications. *Journal of Cybersecurity* 1, 1 (Jan. 2015), 69–79.
- Affiliated event: Encryption and surveillance. In *Proceedings of the 38th International Cryptology Conference*. IACR, (Aug. 19, 2018); <https://crypto.iacr.org/2018/affevents/legal/page.html>
- Bellovin, S. et al. Op-ed: Ray Ozzie's crypto proposal—a dose of technical reality. *Ars Technica* (May 7, 2018); <https://arstechnica.com/information-technology/2018/05/op-ed-ray-ozzies-crypto-proposal-a-dose-of-technical-reality/>
- Hennessey, S. and Wittes, B. Apple is selling you a phone, not civil liberties. *Lawfare* (Feb. 18, 2016); <https://lawfareblog.com/apple-selling-you-phone-not-civil-liberties>
- National Academies of Sciences, Engineering, and Medicine. *Decrypting the Encryption Debate: A Framework for Decision Makers*. The National Academies Press, Washington, D.C., 2018; <https://doi.org/10.17226/25010>
- Ozzie, R. CLEAR. (Jan. 2017); <https://github.com/rayozzie/clear/blob/master/clear-rayozzie.pdf>.
- Savage, S. Lawful device access without mass surveillance risk: A technical design discussion. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (Toronto, Canada, Oct. 15–19), 2018.
- Wright, C. and Varia, M. Crypto crumble zones: Enabling limited access without mass surveillance. In *Proceedings of the 3rd IEEE European Symposium on Security and Privacy*. IEEE Computer Society, 2018, 288–306.
- Zittrain, J. et al. Don't Panic: Making Progress on the 'Going Dark' Debate. Berkman Center Research Publication 2016-1, (2016); <http://nrs.harvard.edu/urn-3:HUL.InstRepos:28552576>

Joan Feigenbaum (joan.feigenbaum@yale.edu) is the Grace Murray Hopper Professor of Computer Science at Yale University, New Haven, CT, USA.

This work was supported in part by U.S. National Science Foundation grants CNS-1407454 and CNS-1409599 and U.S. Office of Naval Research grant N00014-18-1-2743.

Copyright held by author.

DOI:10.1145/3319081

Emmanuel Schanzer, Shriram Krishnamurthi, and Kathi Fisler

► Mark Guzdial, Column Editor

Education

What Does It Mean for a Computing Curriculum to Succeed?

Examining the expansion, proliferation, and integration of computing education everywhere.

COMPUTING EDUCATION IS suddenly everywhere. Numerous U.S. states and many countries around the world are creating requirements and implementing programs to bring computing to their students. Tech innovators have jumped in, too, sometimes to “disrupt” the educational system. Opinion pieces create parental anxiety that their children are not being trained properly for the future; products claim to mollify these anxieties (while perhaps simultaneously amplifying them). Academics, looking to address the Broader Impact criteria of funding agencies, are eager to burnish their credentials by giving guest lectures at local schools. In certain neighborhoods, toystores feel compelled to stock a few products that claim to enhance “computational thinking.”³

Unfortunately, a lot of current discussion about curricula is caught up in *channels* (including in-school versus after-school courses), *media* (such as blended versus online learning), and *content* (for example, Java versus Python). As computer scientists, we should recognize this phenomenon: a focus on *implementation before specification*. Instead, in sober moments, we should step back and ask what the end goals are for this flurry of activity. Is a little exposure good for everyone? How many Hours of Code will prepare a child for a digital future? If a few requirements are good, are more requirements better? In



short: What does it mean for computing education to succeed?

Specification: Three Worthy Goals

Every program would benefit first from a clear articulation of its goals. These goals should be as close as possible to concrete and measurable (and hence go significantly beyond anodyne phrases). We believe a truly ambitious project would have the trio of goals depicted in the figure in this column.

Readers might wonder if this is a “pick two” situation (or even a “pick one”). Indeed, dropping one or more of these demands greatly simplifies

curriculum design, but with worrisome consequences. One can, for instance, obtain massive scale with a very simplistic curriculum (of which we see a good deal of evidence right now), with a focus on “engagement” but little to no rigor. A few high schools already have very rigorous computing curricula (students take several years of computing, reaching material well beyond the first year of college), but these are extremely difficult to scale. Elective classes can be very rigorous, but can easily lose equity: self-selection easily creates a vicious cycle that reinforces existing biases. Expensive curricula (especially

involving physical devices—such as fancy robots and sensors—that must be bought *and repaired*) are very difficult to scale. Trying to pair teachers with working computing professionals may work fabulously in large cities with a big tech population, but would not scale to most rural areas.

Clearly, the outcomes of compromising are undesirable. Not compromising is, indeed, an intellectual and moral imperative:

- ▶ *Equity* is severely lacking in computing. Large-scale curricula with massive investment that ignore equity can only make the problem much worse.

- ▶ *Rigor* is critical to impart content of value. In its absence, we get the light entertainment that passes for many computing curricula today.

- ▶ *Scale* is essential to get computing into the hands of all of today's students who might be tomorrow's users, creators, or even victims of it.

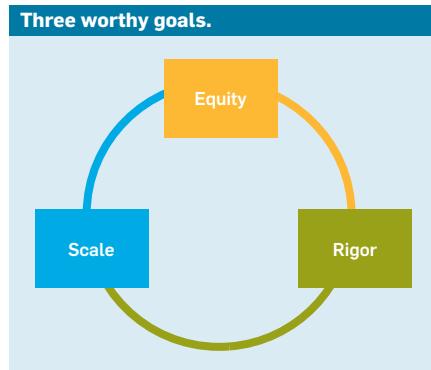
Rather than lay out how their implementation will address Equity, Rigor, and Scale (or other equally worthy goals), many of the players in this space are quick to use the rhetoric of “disruption” to gloss over the challenges outlined in this column. This is not altogether surprising, as many of them share a cultural heritage with (and often financial backing from) a tech industry that is infatuated with the term. Without question, some form of “disruption” is sorely needed—we do, after all, want a much larger and vastly more diverse population to learn rigorous computing—but the question remains which implementation mechanisms will best achieve it. Let's evaluate how two existing avenues fare.

Mechanism 1: Stand-Alone Computing Courses

The most obvious solution seems to be: *add computing courses to every curriculum*. This runs into some natural roadblocks:

- ▶ Schools must find funding to pay for all those new computing teachers.

- ▶ Those teachers need to be qualified, or else rigor will suffer; in a terrific job market, they are extremely difficult to find. (In fact, some great teachers we know have left for industry. Paradoxically, the time when people pay most attention to a field may be the time it is most difficult find enough teachers for it.)



- ▶ Finding qualified teachers can be even more difficult in poor and rural schools than in cities (as we are finding in practice).

- ▶ Schools must make time in the day and space in the building to teach another subject. What will they displace? The humanities? Art? Physics? Statistics?

Some places that are following this route are currently funded generously by the tech industry (usually in return for offering only their chosen curriculum). Since it is unlikely the funding will flow endlessly, what happens when budgets are cut or the largesse dries up? Odds are those courses will be the first to be cut in all but the wealthiest districts, and computing will suffer the same fate as music and art in the USA. Furthermore, because planning interdependent courses is hard, these courses will likely run in a vacuum, making it even simpler to cut them when it becomes convenient to do so.

One growing response is to *mandate* computing courses throughout some geographic region. This automatically achieves equity and scale. However, it comes with its own subtle problems. The problems of funding and qualified teachers do not go away; if anything, they are exacerbated because of the significantly greater demand imposed by a mandate. But there are also subtle problems: if a class is mandatory, there is a perverse incentive to lower the rigor of the course. After all, who wants to see a student held back or lose a scholarship simply because they struggle in their Python class?

Ironically enough, there is not even anything “disruptive” about this model! It more closely resembles an enterprise business deal or a top-down diktat than the kind of organic, bottom-up groundswell the fans of disruption preach. The funding model chosen by disruptive companies turns out to be

Calendar of Events

May 4–9

CHI '19: CHI Conference on Human Factors in Computing Systems, Glasgow, Scotland, UK,
Sponsored: ACM/SIG,
Contact: Geraldine Fitzpatrick,
Email:
geraldine.fitzpatrick@tuwien.ac.at

May 5–6

Expressive '19: Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering, Genoa, Italy,
Sponsored: ACM/SIG,
Contact: Joaquim Jorge,
Email: joaquim.jorge@gmail.com

May 9–11

GLSVLSI '19: Great Lakes Symposium on VLSI 2019, Tysons Corner, VA,
Sponsored: ACM/SIG,
Contact: Baris Taskin,
Email: taskin@coe.drexel.edu

May 13–15

HotOS '19: Workshop on Hot Topics in Operating Systems, Bertinoro, Italy,
Sponsored: ACM/SIG,
Contact: Mirco Marchetti,
Email: mirco.marchetti@unimore.it

May 15–17

WiSec '19: 12th ACM Conference on Security and Privacy in Wireless and Mobile Networks, Miami, FL,
Sponsored: ACM/SIG,
Contact: A. Selcuk Uluagac,
Email: suluagac@fiu.edu

May 17–19

CompEd: ACM 2019 Global Computing Education Conference, Chengdu, China,
Sponsored: ACM/SIG,
Contact: Ming Zhang,
Email: mzhang_cs@pku.edu.cn

May 21–23

I3D '19: Symposium on Interactive 3D Graphics and Games, Montreal, QC, Canada,
Sponsored: ACM/SIG,
Contact: Sheldon Andrews,
Email: sheldon.andrews@gmail.com

the very model they eschewed on their path to success.

Mechanism 2: Integrated Computing

Let's instead consider an alternative model of computing education. It recognizes computing is a new creative medium and vehicle for exploring myriad subjects, ranging from mathematics, biology, and physics to social studies. Why not, then, *integrate* computing into each of these subjects?

Presumably, most people do not believe all other disciplines are going to collapse and be replaced by computing; rather, computing will enrich and enhance those subjects. Therefore, those subjects should start modifying their presentation to show the impact computing will have. In social studies, for instance, there are already well-established means of asking and answering questions (surveys, ethnographic studies, literature reviews, and so forth). Computing does not displace these but rather supplements them, providing a new and rich way to pose questions: *a program is a way of posing a question of a dataset*. In turn, not every student is enamored of computing, either, and a generic introduction to computing is unlikely to sway them. In contrast, a contextual introduction in a subject that already interests them is far more likely to get them to see the value of computing.

Integrated materials can achieve all three of the goals we have described in this column. By embedding into already-required courses (such as math), they achieve the same diversity and scale as required computing courses do, without the same constraints. Rigor follows much more directly because of the existing rigor of subjects it embeds into: teachers in those subjects would not accept a curriculum that does not seem to make a meaningful contribution to how they teach their discipline. All this can be done at far lower cost, because it does not require entire new cadres of teachers to be hired or new classes to be added; the burden shifts to training the teachers already in the system or those entering it.

Curiously, integrated computing adheres far more closely to the model of disruption so beloved in our industry. It is lightweight: it does not require large outlays of time, space, and money. It has few

But integrated computing is imperative for another reason, too: computing should not fall victim to the same peril that befell mathematics.

dependencies, so it is easy to parallelize. It usually follows from bottom-up, grassroots interest. It is “sticky”: it is unlikely to disappear when a generous donor’s priorities change. And it lends itself to strong network effects in multiple ways: teachers within a discipline reinforce and improve the computing integration for their discipline, while teachers within a school support and reinforce student computing education for each other.

This, of course, is the good news. The bad news is that integrating computing is far more difficult than delivering it as a stand-alone subject. Teachers in other disciplines need to be convinced that computing has anything to offer. Airy promises of the power of “computational thinking” are met with appropriate skepticism from teachers in other disciplines, because more than 100 years of quality education research shows the difficulties of achieving transfer across disciplines.¹ Validated research is much more compelling, and this takes time and effort. Also, teachers feel pressure to choose between doing more of their own discipline, or sacrificing some content they know and love to make room for computing. Thus, an injection of computing must be judicious, focusing on content that is *meaningful in the host discipline*; it must also “pay its own way,” providing large value for small investments of time. Achieving all this is difficult.

Difficult, but not impossible. Programs like AgentSheets,^a Project GUTS,^b agent-based modeling,^c and Bootstrap^c

are existence proofs that we are making substantial progress toward our stated goals. Thus, we believe integration is a strategy well worth pursuing, in parallel to stand-alone, required computing.

Mathematics: A Cautionary Tale

In short, integrated computing can achieve all three criteria we have described, which stand-alone approaches struggle to meet. But integrated computing is imperative for another reason, too: computing should not fall victim to the same peril that befell mathematics. While math dramatically impacts numerous disciplines, it is routinely siloed into stand-alone classes; as a result, the connections between math and other disciplines are often invisible to K-12 students. (In contrast, some institutions have tried to institute “writing across the curriculum,” to help students improve their writing in a context meaningful to them.) Computing has a chance to avoid this fate, and the evidence so far is that we can succeed at integration. Moreover, stand-alone courses would be much richer if their intake consisted of students already versed in computing from other disciplines. Thus, with the right models of curricular design, integration strategies, and funding, we can achieve sustainable Equity, Rigor, and Scale. C

References

- Bransford, J.D. and Schwartz, D. Rethinking transfer: A simple proposal with multiple implications. In *Review of Research in Education*, 24, American Educational Research Association, 1999, 61–100.
- Wilensky, U. and Rand, W. *An Introduction to Agent-based Modeling: Modeling Natural, Engineered, and Social Complex Systems with NetLogo*. MIT Press, Cambridge, MA, 2015.
- Wing, J. Computational thinking. *Commun. ACM* 49, 3 (Mar. 2006), 33–35.

Emmanuel Schanzer (schanzer@bootstrapworld.org) works at Brown University, Providence, RI, USA. He is a co-Director and founder of Bootstrap. A former public high school teacher in Boston, MA, USA, he designed Bootstrap:Algebra as a curriculum for his own students after being exposed to the Program by Design methodology in college.

Shriram Krishnamurthi (sk@cs.brown.edu) is a Professor of Computer Science at Brown University, Providence, RI, USA, and a co-Director of Bootstrap. He co-founded the Program by Design project, which inspired Bootstrap.

Kathi Fisler (kathi@bootstrapworld.org) is a Research Professor of Computer Science at Brown University, Providence, RI, USA, and a co-Director of Bootstrap.

DOI:10.1145/3319912

Alberto Bartoli, Eric Medvet, Andrea De Lorenzo, and Fabiano Tarlao

Viewpoint

Enterprise Wi-Fi: We Need Devices That Are Secure by Default

Seeking to increase awareness of WPA2 Enterprise network security technology flaws and reduce risk to users.

WOULD YOU TRUST a security technology that makes it possible (that is, quite likely) to steal the single sign-on enterprise credentials of any specific person in your enterprise by merely walking within 30 meters from that person? The attacker does not need to do any visible activity that might raise suspicions: a 50-euros device in a bag and a few seconds of physical proximity is all that is needed. Active cooperation of the target is not required and Internet connectivity is not required either. Thus, the attack may occur anywhere and the target would not notice anything. The attacker could steal the single sign-on credentials of a large fraction of people of your enterprise that happen to pass within 30 meters from the attacker. Perhaps at the office lunchroom, near a mass-transportation hub, or anywhere outside of the enterprise.

Of course, you would not trust such a security technology. Interestingly, though, a technology of this kind is nearly ubiquitous and implicitly trusted by a lot of people and enterprises: it is WPA2 Enterprise—the suite of protocols for secure communication in enterprise wireless networks. It is necessary to emphasize the relevance of this important and



pervasive yet largely underestimated risk. We need to raise the awareness on a fundamental security technology that is very often deployed by violating its requirements, which creates important risks to users.

Stealing Enterprise Credentials with Evil Twins

The attack scenario described in this Viewpoint may succeed whenever people connect to enterprise wireless networks with devices that are *not configured correctly*. We recently

showed (see Bartoli et al.²) that by just wandering around for a few hours in regions not covered by a wireless network we could have collected 200 enterprise credentials—approximately 25% of them in clear text and the remaining ones in hashed MS-CHAPv2 form—which can generally be decrypted easily.⁹ We also showed that by remaining for a few seconds at less than 35 meters from a specific (voluntary) target whose Wi-Fi device is not configured correctly, there is a very good chance the attacker may

COMMUNICATIONS APPS

Access the latest issue, past issues, BLOG@CACM, News, and more.



Available for iPad, iPhone, and Android



Available for iOS, Android, and Windows

<http://cacm.acm.org/about-communications/mobile-apps>



Association for Computing Machinery

steal his/her enterprise credentials; even when he/she is sitting in a car with closed windows.

The reason why these attacks work is quite simple. A Wi-Fi device (a *supplicant* in WPA2 Enterprise) may connect to an enterprise network only after executing an authentication protocol with the *authentication server* for that network. For the purpose of this Viewpoint, this protocol may be summarized as follows:

1. The supplicant verifies the identity of the authorization server for the network it is attempting to connect to;
2. The supplicant sends user credentials material to the authorization server; and
3. The authorization server demonstrates knowledge of the user credentials to the supplicant.

Execution of the first step relies on certain configuration information that must be stored on the supplicant before connecting. Such information includes the association between the name of the network and the name of the authorization server (for example, in our university these are eduroam and raggio.units.it, respectively); it also includes the association between the name of the authorization server and a certified public key for that server.

Satisfying this configuration requirement is extremely important, because supplicants that are *not* configured correctly will execute the first step with any fraudulent Wi-Fi access point broadcasting the name of an enterprise network (*evil twin*).^{3,4,7,9} The authentication protocol will fail because the evil twin will not be able to prove knowledge of the user credentials. However, the supplicant will disconnect when it is too late, that is, after having already sent credential material to the evil twin. The entire procedure takes just a few seconds and, most importantly, it does not require any engagement from the user, in particular when the user carries a smartphone with the Wi-Fi interface active.

Enterprise Wi-Fi Today

Enterprise credential stealing through evil twins is not based on any vulnerability of WPA2 Enterprise: it is

an obvious consequence of using this technology without satisfying one of its fundamental deployment requirements, that is, correct supplicant configuration.⁸ The problem is, supplicants that connect to enterprise networks without being configured correctly are pervasive. All the analyses we are aware of corroborate this claim unequivocally and even network configuration guides prepared by organizations suggest insecure configuration practices very often.¹ Attacks based on an evil twin have thus a high probability of success.

What makes this risk important and pervasive is that WPA2 Enterprise was specified in 2004 but the world today is very different:

► Virtually all enterprises have migrated to single sign-on architectures. Enterprise network credentials now usually unlock access to *all* enterprise services, including in particular all the services with a Web interface. Network credentials are thus much more attractive and valuable to attackers than they used to be.

► Most people are now permanently carrying a Wi-Fi-enabled smartphone that often contains the user's enterprise credentials and connects to Wi-Fi networks *automatically*. An evil twin may now be assembled with a few tens of euros and can be placed in a bag.² Attacks aimed at stealing network credentials may thus occur potentially anywhere and are virtually impossible to detect: they are executed automatically, in less than a second of proximity to an evil twin and without any need of involving the device owner in a working session.

One of the reasons for the prevalence of incorrectly configured supplicants is because defining an insecure configuration is much simpler and quicker than defining a secure one. A secure configuration requires the presence of certain configuration data on the supplicant, which must be preliminarily obtained through a connection link different from the enterprise network. Insecure configurations are much more straightforward to define as they do not require any prior connection and download: select the name of the enterprise network, insert enterprise credentials, and then play with the network con-

figuration until connecting, the most typical (insecure) options being “skip certificate validation” or “accept any certificate.” Users are not to blame for this behavior, as they cannot appreciate the resulting risks. Most importantly, they have no incentive in selecting a more cumbersome path toward their objective: connecting their device to the network.⁵

One could require enterprises to deploy forms of “entraps” for detecting supplicants that are not configured correctly and then notifying the corresponding users to contact the IT staff. On the other hand, it is a fact that many organizations not only tolerate but also suggest insecure configuration practices,¹ hence assuming the occurrence on a large scale of a spontaneous change that increases the technical and operational burden of IT staff seems to be excessively optimistic. Indeed, procedures of this kind may be implemented already but it is fair to say they are actually deployed quite rarely.

Secure by Default

We call on the technical community to disseminate the awareness that we are systematically deploying a widespread security technology by violating its key requirements, thereby creating important risks to users and organizations. WPA2 Enterprise devices will not disappear anytime soon, thus we must all be aware we will have to cope with the pervasive risks of enterprise Wi-Fi for many years to come.

We also need to devise a transition plan that, in our opinion, necessarily requires a novel design for supplicants. The root cause of the problem is that users invariably attempt to connect their supplicants by providing only the name of the enterprise network, that is, without preliminarily obtaining the additional information required for a secure configuration. Consequently, we advocate a design that makes it *impossible* for those supplicants to connect. We believe this is the only realistic strategy for providing both users and enterprises with adequate incentives for installing the required information on supplicants.

Obviously, supplicant configuration should be sufficiently simple

Although these design principles do not prevent the possibility of insecure configurations, they are sufficiently specific to be actionable.

to make the new framework acceptable in practice and the likelihood of ending up in insecure configurations should be minimized. These generic recommendations can be made more useful and concrete based on *secure by default* design principles:⁶ configuration should not require specific technical understanding and it should require only the insertion of a few short pieces of textual information (to prevent the need of non-obvious actions from the user such as downloading and installing a program or binary data with a device not yet connected to the network). Although these design principles do not prevent the possibility of insecure configurations, they are sufficiently specific to be actionable.

We have proposed a design based on this framework² but we believe any secure by default approach needs strong incentives that cannot be purely technical.⁵ Manufacturers are unlikely to place on the market supplicants whose configuration involves a user experience quite different from the established one. Support from a standards body in the form of certification requirements for supplicants could constitute a strong and probably decisive incentive for adopting a secure by default approach in the context of enterprise Wi-Fi.

There is an important opportunity in this respect, because the Wi-Fi Alliance has announced new security protections will be specified soon as part of the new family of WPA3 pro-

tocols.¹⁰ It is unclear whether the issue of supplicant configuration in enterprise Wi-Fi will be addressed and how: the limited information that is currently available is not very encouraging, as the focus is on new cryptographic protections but crucial issues of supplicant configuration are not mentioned at all.¹⁰ It would be unfortunate if the new standard did not remove security assumptions that have proven to be unrealistic, as the society as a whole would have to cope with the resulting risks without any transition plan on the horizon. We can no longer afford devices that are secure only if configured correctly, but that may be used insecurely anyway and are typically used so. □

References

1. Bartoli, A. et al. (In)Secure Configuration Practices of WPA2 Enterprise Supplicants. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*; <https://dl.acm.org/citation.cfm?id=3230838>
2. Bartoli, A., Medvet, E., and Onesti, F. Evil twins and WPA2 Enterprise: A coming security disaster? *Comput. Secur.* 74 (2018), 1–11.
3. Brenza, S., Pawłowski, A., and Pöpper, C. A practical investigation of identity theft vulnerabilities in Eduroam. In *Proceedings of the 8th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, NY, 2015, 1–14.
4. Cassola, A. et al. A Practical, Targeted, and Stealthy Attack Against WPA Enterprise Authentication. NDSS—Network and Distributed Security Symposium, 2013; <https://www.ndss-symposium.org/ndss2013/ndss-2013-programme/practical-targeted-and-stealthy-attack-against-wpa-enterprise-authentication/>
5. Schneider, F.B. Impediments with policy interventions to foster cybersecurity. *Commun. ACM* 61, 3 (Mar. 2018), 36–38; doi: 10.1145/3180493
6. Secure by Default. National Cyber Security Centre [Internet], (May 2, 2017); <https://www.ncsc.gov.uk/articles/secure-default>
7. Snodgrass, J. BYO-Disaster and Why Corporate Wireless Security Still Sucks. DEFCON 21; <https://www.defcon.org/images/defcon-21/dc-21-presentations/djwifbone-PuNk1nPo0p-DEFCON-21-djwifbone-PuNk1nPo0p-BYO-Disaster-Updated.pdf>
8. Souppaya, M. and Scarfone, K. Guidelines for securing wireless local area networks (WLANs). NIST Spec Pub. belts; 2012:800:153; <https://www.nist.gov/publications/guidelines-securign-wireless-local-area-networks-wlans-0>
9. Weaknesses in MS-CHAPv2 authentication. Microsoft Technet (Aug. 20, 2012); <https://blogs.technet.microsoft.com/srd/2012/08/20/weaknesses-in-ms-chapv2-authentication/>
10. Wi-Fi Alliance introduces security enhancements. In Wi-Fi Alliance (Jan. 8, 2018); <https://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-introduces-security-enhancements>

Alberto Bartoli (bartoli.alberto@units.it) is an associate professor at the University of Trieste, Italy.

Eric Medvet (emedvet@units.it) is an assistant professor at the University of Trieste, Italy.

Andrea De Lorenzo (andrea.delorenzo@units.it) is a research fellow at the University of Trieste, Italy.

Fabiano Tarlao (ftarlao@gmail.com) is a research fellow at the University of Trieste, Italy.

Copyright held by authors.



The many challenges to maintaining stored information and ways to overcome them.

BY RAYMOND BLUM AND BETSY BEYER

Achieving Digital Permanence

DIGITAL PERMANENCE HAS become a prevalent issue in society. This article focuses on the forces behind it and some of the techniques to achieve a desired state in which “what you read is what was written.” While techniques that can be imposed as layers above basic data stores—blockchains, for example—are valid approaches to achieving a system’s information assurance guarantees, this article will not discuss them.

First, let’s define *digital permanence* and the more basic concept of *data integrity*.

Data integrity is the maintenance of the accuracy and consistency of stored information. *Accuracy* means the data is stored as the set of values that were intended. *Consistency* means these stored values remain the same over time—they do not unintentionally waver or morph as time passes.

Digital permanence refers to the techniques used to anticipate and then meet the expected lifetime of data

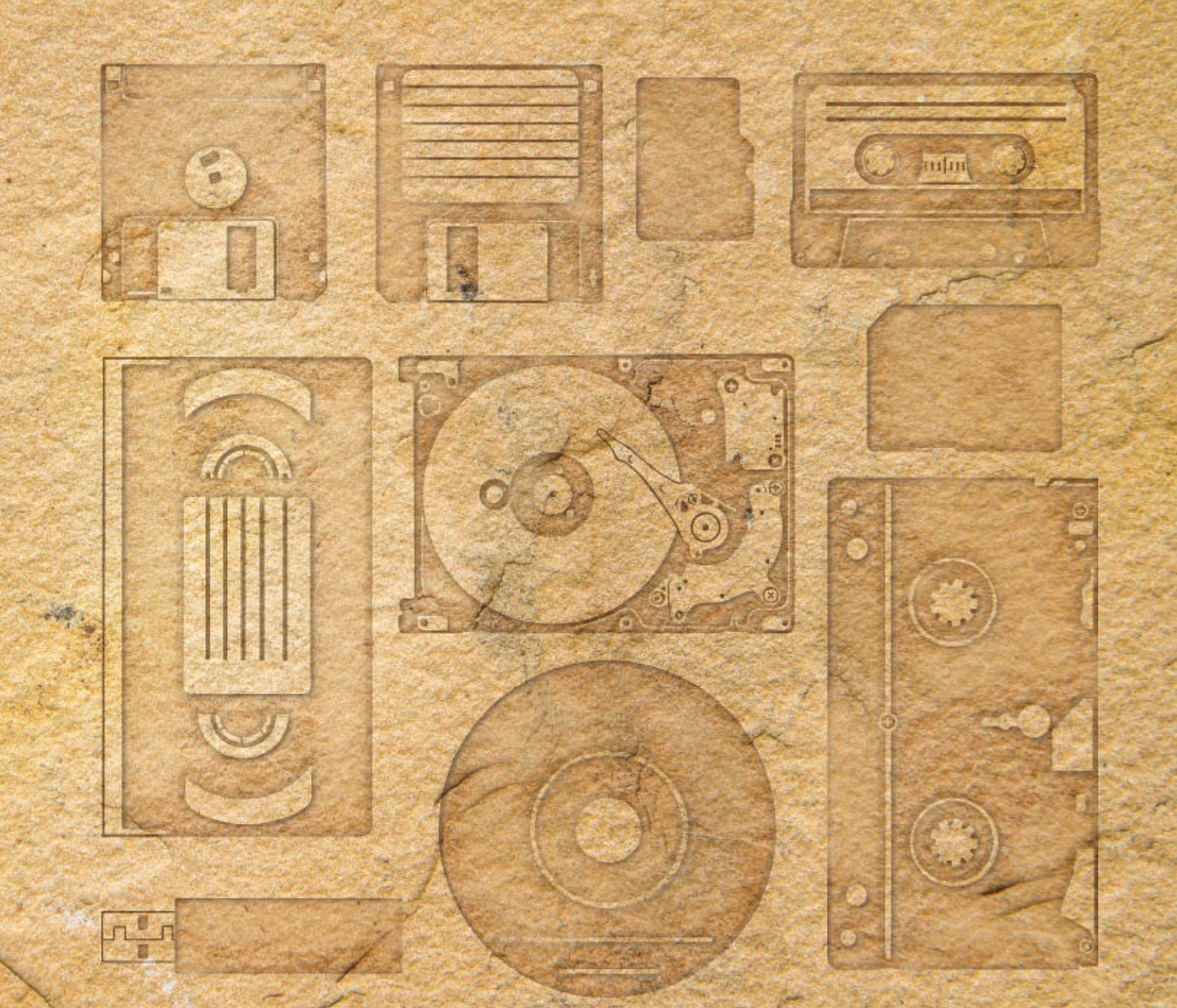
stored in digital media. Digital permanence not only considers data integrity, but also targets guarantees of relevance and accessibility: the ability to recall stored data and to recall it with predicted latency and at a rate acceptable to the applications that require that information.

To illustrate the aspects of relevance and accessibility, consider two counterexamples: journals that were safely stored redundantly on Zip drives or punch cards may as well not exist if the hardware required to read the media into a current computing system isn’t available. Nor is it very useful to have receipts and ledgers stored on a tape medium that will take eight days to read in when you need the information for an audit on Thursday.

The Multiple Facets of Digital Permanence

Human memory is the most subjective record imaginable. Common adages and clichés such as “He said, she said,” “IIRC (If I remember correctly),” and “You might recall” recognize the truth of memories—that they are based only on fragments of the one-time subjective perception of any objective state of affairs. What’s more, research indicates that people alter their memories over time. Over the years, as the need to provide a common ground for actions based on past transactions arises, so does the need for an objective record of fact—an independent “true” past. These records must be both immutable to a reasonable degree and durable. Media such as clay tablets, parchment, photographic prints, and microfiche became popular because they satisfied the “write once, read many” requirement of society’s record keepers.

Information storage in the digital age has evolved to fit the scale of access (frequent) and volume (high) by moving to storage media that records and delivers information in an almost intangible state. Such media has distinct advantages: electrical impulses and the polarity of magnetized ferric compounds can be moved around at great



speed and density. These media, unfortunately, also score higher in another measure: fragility. Paper and clay can survive large amounts of neglect and punishment, but a stray electromagnetic discharge or microscopic rupture can render a digital library inaccessible or unrecognizable.

It stands to reason that storing permanent records in some immutable and indestructible medium would be ideal—something that, once altered to encode information, could never be altered again, either by an overwrite or destruction. Experience shows that such ideals are rarely realized; with enough force and will, the hardest stone can be broken and the most permanent markings defaced.

In considering and ensuring digi-

tal permanence, you want to guard against two different failures: the destruction of the storage medium, and a loss of the integrity or “truthfulness” of the records.

Once you accept that no ideal medium exists, you can guard against both of these failures through redundancy. You can make a number of copies and isolate them in different failure domains so some of them can be counted on to survive any foreseeable disaster. With sufficient copies kept under observation through frequent audits and comparison, you can rely on a quorum of those copies to detect and protect the record from accidental or deliberate alteration.

Copies made for these purposes have different motivating factors, which can

be placed into two categories:

► *Backups*: Copies that protect operations from failures caused by an act of nature or neglect.

► *Archives*: Copies made to preserve the record from the forces of change, be they deliberate or coincidental.

Information Permanence in the Digital Age

Before the 1970s disembodied information did not exist outside of gossip or bardic lore. For thousands of years, knowledge was preserved by altering physical artifacts: from the 3000 BCE rations of Mesopotamian beer to the 1952 tax rolls of the state of Rhode Island, giving permanent life to a fact meant marking a clay tablet, parchment scroll, or paper punch card. Setting aside the

question of its truth, the fact of record existed in plain sight, made permanent in chiseled marks or insoluble ink for the life of the artifact. While fire, flood, or fugitive dye might have challenged the durability of the records, barring destruction or theft, it was reasonable to assume the artifacts of record would remain consistent. A date of birth or tax payment committed to the official record would be the same when recalled for the next decade's census or audit.

In the post-Renaissance and post-Industrial Revolution eras, as humanity embarked upon more and more endeavors with time spans of decades or years, the amount of information that was critical for society to retain exploded. Typesetting and printing processes were optimized and automated to scale up with the increasing need for recorded information. While codices and microfiche use space far more efficiently than clay tablets or scrolls, society cannot dedicate an infinite amount of space to storing copies of birth records and articles of incorporation.

Then came the Information Age.

Suddenly, it seems that nothing can be allowed to slip into obscurity: street maps, bank account records, personal timelines, birthday party videos—all are recorded and stored. While they may lay unused for decades or centuries, we fully expect that the data will be available for later research or perusal. As the volume of historical record surges, the classic model of devoted storage artifacts—be they stone, paper, or plastic—cannot keep up, a perfect manifestation of the adage, “It doesn’t scale.” Paper is too bulky and takes too long to write on and to read. It’s safe to say that recording the history of the world in tangible, physical form is no longer feasible.

Conveniently and not coincidentally, the very same technological advances that created this problem of too much information also led to available solutions. We now have the ability to store information in a “purer” electronic form, broadly and commonly referred to as digital media. The electronic digital representation of information is accomplished with far less energy and space than with older physical or “analog” recording techniques. To use a very coarse measure for comparison: whereas a typical book might weigh 12 ounces and contain 80,000 words, the

Digital permanence not only considers data integrity, but also targets guarantees of relevance and accessibility.

same amount of information can be stored as 3.2 million bits, which occupies 1/10,000th of a commonly available micro SD (secure digital) card that weighs 0.016 ounces. Compared to a paper novel, that SD card has at least 7.5 million times the information per ounce (and this ignores the application of various techniques to increase the efficiency of digital storage space, such as compression and de-duping).

As the density of information has increased, recording and reading rates have also necessarily increased by a similar order of magnitude. If you record 100,000 times more information, but do so at the same rate of transcription, you will accumulate quite a backlog of facts, figures, and news articles to be committed to permanent records. Luckily, it takes far less energy and time to flip the state of submicroscopic bits than to carve notches in stone or to drag a pen to deposit ink on a sheet of paper.

Not surprisingly, while these faster, more fluid storage media is a blessing in one aspect, they are a curse in another. State that is easily set is also easily unset, either unintentionally or maliciously. RAM, flash memory, and magnetic disks can be corrupted through chance interactions that are far more lightweight than actions that can wipe out older, physical media. It might take an intense, persistent building fire to destroy file cabinets full of marriage certificates in the basement of a hall of records, but some stray electromagnetic emission could wipe out the same information stored on a couple of SSDs (solid-state drives). To make matters worse, it’s immediately obvious that your basement was on fire, but you might not know that the contents of your SSDs were corrupted until months or decades later when you need to access the data they once contained.

Concerns over the permanence of recorded information were easily addressed in the past—mechanisms such as stone, archival-quality papers and inks, and fireproof vaults provided well-understood and easily implemented assurances that records would survive for predictable periods of time. The lifetime of encoding techniques was rarely an issue unless you encountered records made in an obsolete language (such as the Egyptian hiero-

glyphs that modern people could not decipher until the Rosetta Stone decree was discovered).

Permanence has become a very real problem as storage techniques and media churn rapidly. While you can rely on a medium such as stone or parchment for a historically demonstrated value of permanence, the impermanence of modern media such as magnetic tape, CD-ROMs, and flash memory has been a surprise. The evolution of paper production in the mid-19th century perhaps foreshadowed this trend. As demand for mass-printed material increased, printers shifted from rag-based paper to more quickly and cheaply produced lignin-rich wood-pulp paper. As a result, archivists and comic-book collectors were surprised and disappointed by the fragility of the cheaper medium.

If paper was a disappointment, at least its permanence faced no challenges beyond the durability of the medium itself: reading a page requires only the sense of sight, which has not changed much since the earliest written records were made. Digital media has introduced new concerns: you cannot directly sense the information on, for example, a flash-memory module; you need specialized equipment to interpret the impressions left on digital storage media, and this equipment must be available and able to provide an interface relevant to current information-processing systems. In short, having well-preserved magnetic tape isn't enough: you need a functioning tape drive, and you must be able to interface the tape drive with your computing system.

In addition to being less permanence-resilient than older, nonelectronic (hereafter, *analog*) storage media, digitally stored data is subject to yet another pressure: the increasing demands for precision in this data-driven world demands unerring reliability. While a measure of 10 acres or three pounds would have been accepted with some understood or even expected margin of error in the past, today's expectations are increasingly precise: 13 ounces or 310 Euros must mean exactly that. The world demands both a growing amount of relevant and necessary data and better "quality" or precision of that data.

Not coincidentally, these demands align with the shift from analog values

and media to their digital counterparts: a drawn line may be perceived as crossing the Y axis at "just around" 10, but a recorded digital metric is either 10 or it is not. When using a slide rule, precision is tied to the perception and visual acuity of the operator, whereas an electronic calculator displays a precise, viewer-agnostic value out to many digits of precision. Modern society also expects immediate results: queries should be answered in real time and transactions should complete almost immediately, so that dependent actions can proceed.

The overall effect of this set of forces is simply summarized: we need to store ever more information (greater breadth), of higher precision or resolution (greater depth), while maintaining or decreasing the latency of access (greater throughput). The increased relevance of the information (greater impact) on lives demands higher fidelity from storage techniques (greater reliability). We need digital permanence.

Categorizing Failure Modes

Any number of triggers can introduce failure modes of storage techniques and media, but there are some broad categories of failure to help identify the most likely vulnerabilities and effective means of mitigation:

- ▶ Latent failure incurred by the passage of time—Staleness of media, bitrot.
- ▶ Failure introduced by *force majeure* events—Typically site disasters such as earthquake, fire, flood, electromagnetic pulse, or asteroid impact.
- ▶ Failures caused by malevolence or ignorance—Most often, exploitations of process deficiencies.
- ▶ Failures caused by usage in unanticipated operation sequence or volume—Usually planning deficiencies.
- ▶ Failures resulting from flaws in systems or their components—Bugs and a lack of isolation or a means to contain their effects.

Failures also have distinct timelines or life cycles:

- ▶ Big bang—Significant amounts of data are affected at once. An event or atomic operation causes systemic harm.
- ▶ Slow and steady—Corruption or loss trickles into a data store at a rate that is probably on the same order of magnitude as normal access, perhaps as a side effect of normal operations.

The scope of a failure can also be classified:

- ▶ Widespread—Large, broad swaths of data are affected, seemingly without regard for discriminators within the data.
- ▶ Narrow and directed—Specific subsets of the stored data store are affected, presumably with some discernable pattern that a domain expert would recognize.

A given failure will have at least one value for each of these three aspects—category, timeline, and scope—so the potential failures can be visualized as a matrix, shown in Table 1.

According to this matrix, a comprehensive view of risk should take up to 20 ($5 \times 2 \times 2$) different failure modes into account. An effective plan for gauging and ensuring digital permanence within this system must include either a means to mitigate each of these possible failure modes, or acknowledgment of unaddressed risks. The likelihood and impact of each failure must also be quantified in some way. No matter how comprehensively (or superficially) you plan on handling a given failure, you should recognize what it is and how much it may cost you. This analysis will help prioritize your budget for ensuring digital permanence and disaster-recovery planning.

Mitigating Risks to Digital Permanence

These failure modes are as similar as chocolate and concrete (apples and or-

Table 1. Failure categorization matrix.

Category	Timeline	Scope
Introduced over/by time		
Force majeure	Big bang	Widespread
Malevolence or ignorance		
Unanticipated usage	Slow and steady	Narrow and directed
Defects		

anges actually do have a lot in common). It follows that appropriate mitigations are also wide ranging. While keeping a full offline data-store copy is a reasonable failsafe for a big bang (for example, widespread loss caused by an asteroid slamming into a datacenter), this tactic isn't ideal for guarding against user error that deletes one account's transactions for the past business day. Your response to this diversity of risks might be to diversify your platforms, avoiding failure caused by a vulnerability specific to one platform. Defense via platform diversity has its appeal but also its drawbacks—stitching together myriad and diverse media, transfer rates, and vendor support levels can become an overwhelming task in itself, leaving little time for your day job.

The complexity of this problem space calls for a well-reasoned strategy for achieving digital permanence in a given system. This section examines methods for codifying coverage of two different aspects of digital permanence in a system, broadly categorized as *data integrity* and *accessibility*.

Preserving data integrity. The data integrity goal is fairly easily stated: If you store some value V in a system, indexed or identified as K , you expect to be able to call up K at some later time and be certain that the value retrieved is, in fact, V . The inherent problem here is one of *trustworthiness*: the system should be relied upon to do its job. If the retrieved value was in fact $V_2 \neq V$, how would you know? If your application is expected to constantly checksum and verify the storage layer's operations, you are experiencing a major abstraction leak and are almost certainly on your way to writing a spectacular *God class*.

A better strategy is to implement a set of guarantees and checks outside of any client application—operations that are conceptually part of the storage system(s). These operations aim to detect and recover from the failures that a storage system may encounter, independent of any current or future client system. Table 1 discusses failure modes and means to address them somewhat generically; specific imple-

mentations will be defined for and by the system under scrutiny.

The examples in Table 2 are not meant to be exhaustive; rather, they provide a sufficiently large example to illustrate the recommended methodology. Column 1 identifies a set of failure modes, and column 2 provides mitigations for each failure mode. The numerals in column 2 identify overlap in the pool of processes and mechanisms so that you can optimize the ROI for each technique used. The goal is to obtain the most coverage for the smallest investment.

To make the best use of this table, you need to be able to weigh the different failures and mitigations so you can prioritize solutions. Table 3 rates the impact of each failure mode.

Table 4 shows the relative cost of fully implementing each proposed mitigation technique.

Now that you understand the options, their relative costs, and their relative values, you can optimize to find the best coverage per cost. The final set of mitigation techniques is optimized with the following parameters:

- All failure modes with an impact other than low must be addressed, but you should provide mitigation techniques for all failure modes if there is no additional cost.

- The lowest-cost option to mitigate a given failure mode is preferred.

- A mitigation technique, once implemented, is applicable to all failure modes for which it is effective.

- Implement as few mitigation techniques as possible in order to minimize the operational complexity of the system.

Table 5 combines the data from tables 2, 3, and 4. The data in Table 5 can be sliced to reveal both the mitigations that provide the broadest coverage and the lowest-cost mitigation for each failure. Note that column [e] is considered optional because failure modes of this category typically have relatively low impact. It's a welcome bonus if you can cover column [e] by piggybacking on mitigations already being implemented for other failure modes.

Broad coverage. Consider a complete data integrity plan to include any set of rows (mitigations) from Table 5 that together provide a value in every column (failure modes). For example, by implementing mitigations in rows

Table 2. Example set of failures and their mitigations.

Failure mode	Means to mitigate
a. Force majeure x big bang x widespread	1. Standby failover serving site 2. Remote data store mirror
b. Introduced over/by time x slow and steady x narrow and directed	3. Parameterized snapshot restore and manual adjustments
c. Introduced over/by time x big bang x widespread	4. Re-create data store from log replay
d. Defects x slow and steady x narrow and directed	3. Parameterized snapshot restore and manual adjustments
e. Force majeure x big bang x narrow and directed	1. Standby failover storage site 2. Remote data store mirror 3. Parameterized snapshot restore and manual adjustments 4. Re-create data store from log replay
f. Malevolence or ignorance x big bang x widespread	4. Re-create data store from log replay

Table 3. Impact of each failure mode.

Failure mode	Impact
a. Force majeure x big bang x widespread	Catastrophic
b. Introduced over/by time x slow and steady x narrow and directed	Medium
c. Introduced over/by time x big bang x widespread	Catastrophic
d. Defects x slow and steady x narrow and directed	Medium
e. Force majeure x big bang x narrow and directed	Low
f. Malevolence or ignorance x big bang x widespread	Catastrophic

[2], [3], and [4], you can achieve complete coverage because each failure mode (column) is addressed.

Lowest-cost mitigation. In addition to coverage, you should consider the total cost of a set of mitigations. For example, the relative costs of rows [1] and [2] might lead you to exclude row [1], as it has a higher cost and provides no additional coverage. If you were optimizing to mitigate failure mode [e], you would choose [3], the lowest-cost applicable mitigation technique.

This exercise does not take into account the likelihood of given failure modes. This factor is highly variable based on the specific failure of a given category: for example, “Asteroid Impact” as an instance of failure mode [a] or “Bad Software Release” as an instance of [d]. The specific failures that a system may experience and their likelihood are dependent on the details of the system being evaluated. When planning data integrity for a given system, prioritizing relevant work, and allocating resources, the individual failures in each category and their likelihood of occurrence should be enumerated and averaged or summed to account for the likelihood of failure.

Now that a framework has been established for preserving the integrity of data stores, let’s turn to a second aspect of digital permanence, called *relevance* or *accessibility*.

Maintaining accessibility. No matter how securely you have locked away hermetically sealed copies of your information, placing every conceivable safeguard in place, there are two surprisingly common snafus that cause the best-laid plans to go awry:

- You can no longer read the data in its preserved form.
- Restoring the data is too expensive to be feasible.

The first issue, one of obsolescence, is well illustrated by an example already given: the ancient Egyptians placed great importance on the fidelity of religious texts and recorded them in stone—the most permanent information storage available. They failed to anticipate that their chosen encoding scheme—hieroglyphs—would be obsolete by the fourth century C.E. As a result, their information, although preserved with high integrity, would be as good as gone for millennia, inde-

Table 4. Cost of each mitigation.

Means to mitigate	Cost
1. Standby failover serving site	High
2. Remote data store mirror	Medium
3. Parameterized snapshot restore and manual adjustments	Low
4. Re-create data store from log replay	Medium

Table 5. Cost vs. coverage of mitigation techniques.

Failure x mitigation	a. Force majeure x big bang x widespread	b. Introduced over/by time x slow and steady x narrow and directed	c. Introduced over/by time x big bang x widespread	d. Defects x slow and steady x narrow and directed	e. Force majeure x big bang x narrow and directed	f. Malevolence or ignorance x big bang x widespread
1. Standby failover serving site	High					High
2. Remote data store mirror	Medium					Medium
3. Parameterized snapshot restore and manual adjustments		Low		Low	Low	
4. Re-create data store from log replay			Medium		Medium	Medium

cipherable until a translation function in the form of the Rosetta Stone was recognized in 1799. Closer to home, consider the family photos stored on several Zip disks along with a spare Zip drive and EISA (Extended Industry Standard Architecture) card in a fire-proof box. While this was a seemingly thorough archive strategy in 1995, it wasn’t thorough enough to make those photos readily accessible using 2018 technology.

You can most simply keep all means of access for all of your data relevant through exercise: employ full end-to-end tests or rotate the live or shadow service through different data stores to validate them. Do it often enough to provide time to address a deprecated storage medium or strained network route before they become totally inaccessible or useless.

The second issue that affects accessibility is one of scale. Somewhat

obviously, the more information you have to process in a given operation, the more resources the processing will take. While transactions are written one at a time, perhaps resulting in a few kilobytes of information per storage operation, restoring a snapshot of data accumulated over months could result in a single storage “operation” from the storage user’s point of view—a restore that must process terabytes or petabytes of information.

That doesn’t come cheap. At transfer rates of common buses such as USB 3.0, the *theoretical* minimum transfer time for a petabyte of data is close to 56 hours. If you are restoring your customer-facing online service’s data, you are not likely to have the luxury of more than two days of unavailability.

At some point, you will have to exploit the classic trade-off of space vs. time, designing parallelism into your data integrity processes to make sure

that the information remains accessible and relevant within acceptable time thresholds. While you may not be able to escape the worst-case scenario of needing to transfer that petabyte, you could perform that transfer with 100 concurrent workers, reducing the 56 hours to less than an hour of wall time, saving your users and your business.

Of course, this strategy is easier said than provisioned. Ultimately, you need to examine the total cost of recovery vs. cost to your business to find the sweet spot. It's a good idea to model a range of scenarios to guide you in determining the resources to devote to data integrity operations. This process is well modeled in a spreadsheet. To return to the previous example: at one end of the spectrum you model the cost of 100 provisioned workers plus the total cost to the business of a one-hour outage; the other end of the spectrum includes the relatively low cost of one provisioned worker plus the presumably high cost to the business of a 56-hour outage. You should include intermediate points such as 10 workers and an outage of close to six hours in the analysis to help find the optimal parameters of your provisioning, communications plans, and playbooks.

Defense in depth. “When it rains it pours,” “Trouble comes in threes,” “Le disgrazie non vengono mai sole;” there’s no shortage of idioms that warn against taking a breather from threats to digital permanence. These threats never go away. There are myriad ways for this pessimistic prediction to manifest. Multiple failures in the failsafe are a common and especially capricious twist of fate: just as you breathe a sigh of relief in the middle of a disaster recovery because you have diligently backed up your data to tape, the tape breaks in the drive. Or you might experience a perfect storm of failures: a network outage causes intermittent timeouts of write operations for users accessing application servers in western Europe, while at the same time, the system that stores transaction logs goes offline when a blue heron flies into an open transformer panel at a datacenter.

Roll your eyes and laugh now, but what can happen will happen, so your plans should employ the principle of defense in depth to protect

your systems from compound or overlapping failures. Remember that these points of failure don’t know about each other and are as likely to happen concurrently as they are to happen at different times.

Bitrot: The forces of decay and neglect. Obsolescence of some critical function or component of a mitigation plan is the most common root cause of disaster-recovery failures. When you have worked hard to come up with a plan to address an unpleasant, annoying, or even painful issue, it’s natural and reasonable to want to put it out of your head and punt follow-up from your calendar. Unfortunately, it’s dangerous to do so. Any system in motion is changing and evolving, so it’s important to respond with accordingly flexible plans. If your plans don’t match the elasticity of the situations they are meant to deal with, the mismatch will lead to decreasing relevance of the plan as the system diverges ever further from its former state.

Bitrot can manifest in many ways: access-control lists expire, resource reservations become obsolete or unavailable, or playbooks are unfamiliar to new staff. There is one simply stated guideline to detect and counter bitrot: practice, practice, practice.

Practicing your recovery plans. A backup should not be taken for granted or viewed as an end goal: try restoring from it; replay transaction logs periodically; failover between alternate sites. These are the operations that you should care about, so make sure that they still actually work as designed. Perform mitigation exercises with a frequency determined by the failures that they address. For example, failover between sites is used to mitigate big bang failures, and therefore should be performed on a noncontinuous basis, perhaps weekly or monthly. Log replay is used to recover from steady-state failures. Therefore, more frequent, continuous, or O(days) end-to-end tests of this operation are appropriate.

In addition to establishing how often to exercise data integrity operations to ensure your expected digital permanence, you need to define the proper scope of these test exercises. The closer your exercise is to a full end-to-end operation, the greater your confidence in it will be. For a failover be-

tween alternate sites, consider actually switching among alternate sites regularly, rather than viewing one site as primary and others as failover or backup sites. Running log recovery against test accounts or regularly selected sets of accounts will either assure you that log replay is currently a trustworthy operation or point out its shortcomings so you can fix any problems or at least know not to rely on this strategy in the event of a failure.

Making It Last and Keeping It True

Every era has introduced new societal challenges when developing and dealing with technological advances. In the Industrial Age, machining methods evolved to produce more, better, and previously undreamt of machines and tools. Today’s Information Age is creating new uses for and new ways to steward the data that the world depends on. The world is moving away from familiar, physical artifacts to new means of representation that are closer to information in its essence.

Since we can no longer rely on the nature of a medium to bestow permanence, we must devise mechanisms that are as fluid and agile as the media to which we are entrusting our information and ever increasing aspects of our lives. We need processes to ensure both the integrity and accessibility of knowledge in order to guarantee that history will be known and true. □

Related articles on queue.acm.org

How Do I Model State?

Let Me Count the Ways

Ian Foster, et al.

<https://queue.acm.org/detail.cfm?id=1516638>

Keeping Bits Safe: How Hard Can It Be?

David S.H. Rosenthal

<https://queue.acm.org/detail.cfm?id=1866298>

META II: Digital Vellum in the Digital Scriptorium

Dave Long

<https://queue.acm.org/detail.cfm?id=2724586>

Raymond Blum leads an engineering team in Google’s Developer Infrastructure that is charged with keeping thousands of Google engineers productive. He was previously a site reliability engineer at Google.

Betsy Beyer is a technical writer for Google Site Reliability Engineering in New York, NY, and the editor of *Site Reliability Engineering: How Google Runs Production Systems*. She has written documentation for Google’s datacenter and hardware operations teams.



Achieving consistency where distributed transactions have failed.

BY MARTIN KLEPPMANN, ALASTAIR R. BERESFORD,
AND BOERGE SVINGEN

Online Event Processing

FOR ALMOST HALF a century, ACID transactions (satisfying the properties of atomicity, consistency, isolation, and durability) have been the abstraction of choice for ensuring *consistency* in data-storage systems. The well-known *atomicity* property ensures that either all or none of a transaction's writes take

effect in the case of a failure; *isolation* prevents interference from concurrently running transactions; and *durability* ensures that writes made by committed transactions are not lost in the case of a failure.

While transactions work well within the scope of a single database product, transactions that span several different data-storage products from distinct vendors have been problematic: many storage systems do not support them, and those that do often perform poorly. Today, large-scale applications are often implemented by combining several distinct data-storage technologies that are optimized for different access patterns. Distributed transactions have failed to gain adoption in most such settings, and most large-scale applications instead rely on ad hoc, unreliable approaches for maintaining the consistency of their data systems.

In recent years, however, there has been an increase in the use of event

logs as a data-management mechanism in large-scale applications. This trend includes the event-sourcing approach to data modeling, the use of change data capture systems, and the increasing popularity of log-based publish/subscribe systems such as Apache Kafka. Although many databases use logs internally (for example, write-ahead logs or replication logs), this new generation of log-based systems is different: rather than using logs as an implementation detail, they raise them to the level of the application-programming model.

Since this approach uses application-defined events to solve problems that traditionally fall in the transaction-processing domain, we name it OLEP (online event processing) to contrast with OLTP (online transaction processing) and OLAP (online analytical processing). This article explains the reasons for the emergence of OLEP and shows how it allows ap-

plications to guarantee strong consistency properties across heterogeneous data systems, without resorting to atomic commit protocols or distributed locking. The architecture of OLEP systems allows them to achieve consistent high performance, fault tolerance, and scalability.

Application Architecture Today: Polyglot Persistence

Different data-storage systems are designed for different access patterns, and there is no single one-size-fits-all storage technology that is able to serve all possible uses of data efficiently. Consequently, many applications today use a combination of several different storage technologies, an approach sometimes known as *polyglot persistence*.

For example:

- *Full-text search.* When users need to perform a keyword search on a dataset (for example, a product catalog), a full-text search index is required. Although some relational databases, such as PostgreSQL, include a basic full-text indexing feature, more advanced uses generally require a dedicated search server such as Elasticsearch. To improve the indexing or search result ranking algorithms, the search engine's indexes may need to be rebuilt from time to time.

- *Data warehousing.* Most enterprises export operational data from their OLTP databases and load it into a data warehouse for business analytics. The storage layouts that perform well for such analytic workloads, such as column-oriented encoding, are very different from those of OLTP storage engines, necessitating the use of distinct systems.

- *Stream processing.* Message brokers allow an application to subscribe to a stream of events as they happen (for example, representing the actions of users on a website), and stream processors provide infrastructure for interpreting and reacting to those streams (for example, detecting patterns of fraud or abuse).

- *Application-level caching.* To improve the performance of read-only requests, applications often maintain caches of frequently accessed objects (for example, in memcached). When the underlying



The architecture of online event processing systems allows them to achieve consistent high performance, fault tolerance, and scalability.



data changes, applications employ custom logic to update the affected cache entries accordingly.

Note these storage systems are not fully independent of each other. Rather, it is common for one system to hold a copy or materialized view of data in another system. Thus, when data in one system is updated, it often needs to be updated in another, as illustrated in Figure 1.

OLTP transactions are predefined and short. In the traditional view, as implemented by most relational database products today, a transaction is an interactive session in which a client's queries and data modification commands are interleaved with arbitrary processing and business logic on the client. Moreover, there is no time limit for the duration of a transaction, since the session traditionally may have included human interaction.

However, reality today looks different. Most OLTP database transactions are triggered by a user request made via HTTP to a Web application or Web service. In the vast majority of applications, the span of a transaction extends no longer than the handling of a single HTTP request. This means that by the time the service sends its response to the user, any transactions on the underlying databases have already been committed or aborted. In a user workflow that spans several HTTP requests (for example, adding an item to a cart, going to checkout, confirming the shipping address, entering payment details, and giving a final confirmation), no one transaction spans the entire user workflow; there are only short, noninteractive transactions to handle single steps of the workflow.

Moreover, an OLTP system generally executes a fairly small set of known transaction patterns. On this basis, some database systems encapsulate the business logic of transactions as *stored procedures* that are registered ahead of time by the application. To execute a transaction, a stored procedure is invoked with certain input parameters, and the procedure then runs to completion on a single execution thread without communicating with any nodes outside of the database.

Heterogeneous distributed transactions are problematic. It is important to distinguish between two types of distributed transactions:

- Homogeneous distributed transactions are those in which the participating nodes are all running the same database software. For example, Google's Cloud Spanner and VoltDB are recent database systems that support homogeneous distributed transactions.

- Heterogeneous distributed transactions span several different storage technologies by distinct vendors. For example, the X/Open XA (extended architecture) standard defines a transaction model for performing 2PC (two-phase commit) across heterogeneous systems, and the JTA (Java Transaction API) makes XA available to Java applications.

While some homogeneous transaction implementations have proved successful, heterogeneous transactions continue to be problematic. By their nature, they can only rely on a lowest common denominator of participating systems. For example, XA transactions block execution if the application process fails during the *prepare* phase; moreover, XA provides no deadlock detection and no support for optimistic concurrency-control schemes.³

Many of the systems listed here, such as search indexes, do not support XA or any other heterogeneous transaction model. Thus, ensuring the atomicity of writes across different storage technologies remains a challenging problem for applications.

Building Upon Event Logs

Figure 1 shows an example of polyglot persistence: an application that needs to maintain records in two separate storage systems such as an OLTP database (for example, an RDBMS) and a full-text search server. If heterogeneous distributed transactions are available, the system can ensure atomicity of writes across the two systems. Most search servers do not support distributed transactions, however, leaving the system vulnerable to these potential inconsistencies:

- *Non-atomic writes.* If a failure occurs, a record may be written to one of the systems but not the other, leaving them inconsistent with each other.

- *Different order of writes.* If there are two concurrent update requests A and B for the same record, one system may process them in the order A, B while the other system processes them in the order B, A. Thus, the systems may disagree on which write was the latest, leaving them inconsistent.

Figure 2 presents a simple solution to these problems: when the application wants to update a record, rather than performing direct writes to the two storage systems, it appends an *update event* to a log. The database and the search index each subscribe to this log and write updates to their storage in the order they appear in the log.⁴ By sequencing updates through a log, the database and the search index apply the same set of writes in the same order, keeping them consistent with each other. In effect, the database and the search index are *materialized views* onto the sequence of events in the log. This approach solves both of the aforementioned problems as follows:

- Appending a single event to a log is atomic; thus, either both subscribers see an event, or neither does. If a subscriber fails and recovers, it resumes processing any events that it has not processed previously. Thus, if an update is written to the log, it will eventually be processed by all subscribers.

- All subscribers of the log see its events in the same order. Thus, each of the storage systems will write records in the same serial order.

In this example, the log serializes writes only, but the application may read from the storage systems at any time. Since the log subscribers are asynchronous, reading the index may return a record that does not yet exist in the database, or vice versa; such transient inconsistencies are not a problem for many applications. For those applications that require it, reads can also be serialized through the log; an example of this is presented later.

The log abstraction. There are sev-

Figure 1. Record written to a database and to search index.

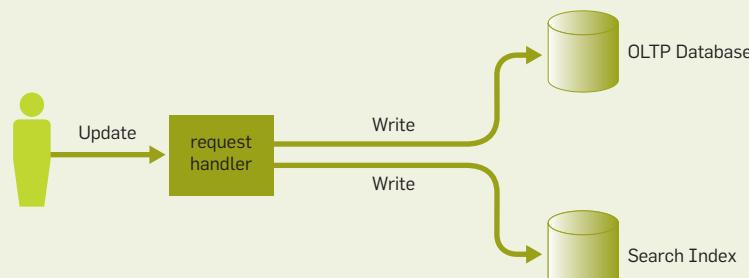
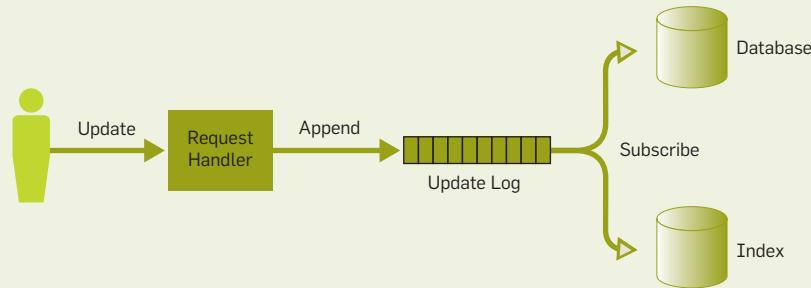


Figure 2. Database and Index use a log to maintain consistency.



eral log implementations that can serve this role, including Apache Kafka, CORFU (from Microsoft Research), Apache Pulsar, and Facebook’s LogDevice. The required log abstraction has the following properties:

- **Durable.** The log is written to disk and replicated to several nodes, ensuring that no events are lost in a failure.

- **Append-only.** New events can be added to the log only by appending them at the end. Besides appending, the log may allow old events to be discarded (for example, by truncating log segments older than some retention period or by performing key-based log compaction).

- **Sequential reads.** All subscribers of the log see the same events in the same order. Each event is assigned a monotonically increasing LSN (log sequence number). A subscriber reads the log by starting from a specified LSN and then receiving all subsequent events in log order.

- **Fault-tolerant.** The log remains highly available for reads and writes in the presence of failures.

- **Partitioned.** An individual log may have a maximum throughput it can support (for example, the throughput of a single network interface or a single disk). The system can be assumed to scale linearly, however, by having

many *partitions*—that is, many independent logs that can be distributed across many machines—and to have no ordering guarantee across different log partitions. Multiple logical logs may be multiplexed into a single physical log partition.

The following assumptions are made about subscribers of a log:

- A subscriber may maintain state (for example, a database) that is read and updated based on the events in the log, and that survives crashes. Moreover, a subscriber may append further events to any log (including its own input).

- A subscriber periodically checkpoints the latest LSN it has processed to stable storage. When a subscriber crashes, upon recovery it resumes processing from the latest checkpointed LSN. Thus, a subscriber may process some events twice (those processed between the last checkpoint and the crash), but it never skips any events. Events in the log are processed at least once by each subscriber.

- The events in a single log partition are processed sequentially on a single thread, using deterministic logic. Thus, if a subscriber crashes and restarts, it may append duplicate events to other logs.

These assumptions are satisfied by existing log-based stream-processing

frameworks such as Apache Kafka Streams and Apache Samza. Updating state deterministically based on an ordered log corresponds to the classic *state machine replication* principle.⁵ Since it is possible for an event to be processed more than once when recovering from a failure, state updates must also be *idempotent*.

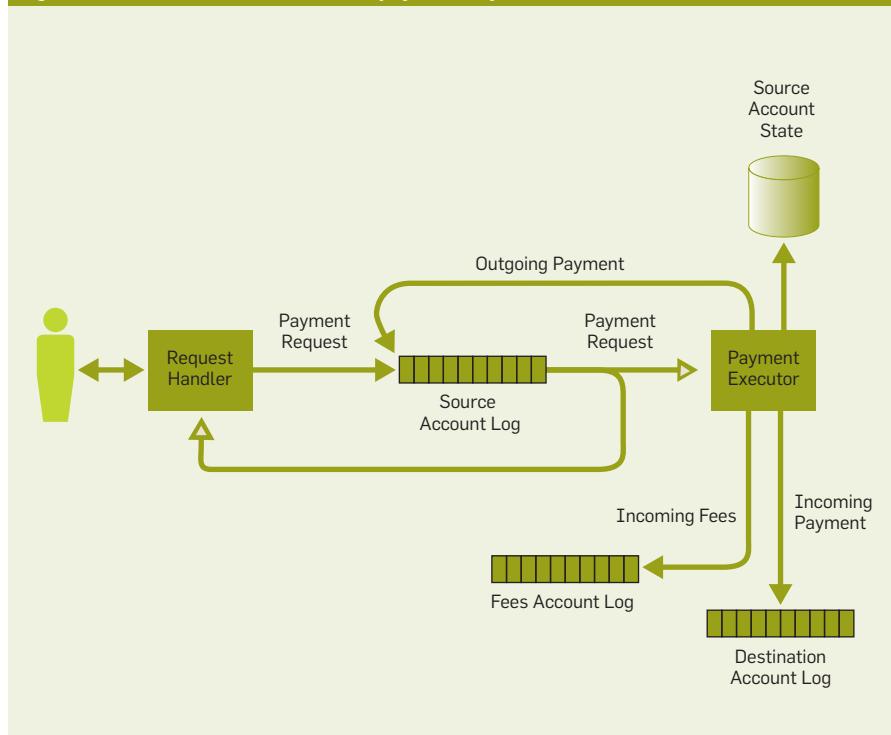
Aside: Exactly-once semantics. Some log-based stream processors such as Apache Flink support so-called *exactly-once semantics*, which means that even though an event may be processed more than once, the effect of the processing will be the same as if it had been processed exactly once. This behavior is implemented by managing side effects within the processing framework and atomically committing these side effects together with the checkpoint that marks a section of the log as processed.

When a log consumer writes to external storage systems, however, as in Figure 2, exactly-once semantics cannot be ensured, since doing so would require a heterogeneous atomic commit protocol across the stream processor and the storage system, which is not available on many storage systems, such as full-text search indexes. Thus, frameworks with exactly-once semantics still exhibit at-least-once processing when interacting with external storage and rely on idempotence to eliminate the effects of duplicate processing.

Atomicity and enforcing constraints. A classic example where atomicity is required is in a banking/payments system, where a transfer of funds from one account to another account must happen atomically, even if the two accounts are stored on different nodes. Moreover, such a system typically needs to maintain consistency properties or invariants (for example, an account cannot be overdrawn by more than some set limit). Figure 3 shows how such a payments application can be implemented using the OLEP approach instead of distributed transactions. Arrows with solid heads denote appending an event to a log, while arrows with hollow heads denote subscribing to the events in a log. It works as follows:

1. When a user wishes to transfer funds from a source account to a destination account, he or she first appends

Figure 3. Flow of events in a financial payments system.



a *payment request* event to the log of the source account. This event merely indicates the intention to transfer funds; it does not imply that the transfer has been successful. The event carries a unique ID to identify the request.

2. A single-threaded *payment executor* process subscribes to the source-account log. It maintains a database containing transactions on the source account and the current balance. This process deterministically checks whether the payment request should be allowed, based on the current balance and perhaps other factors. This log consumer is very similar to the execution of a stored procedure.

3. If the executor decides to grant the payment request, it writes that fact to its local database and appends events to several different logs: as a minimum, an outgoing payment event to the source account log and an incoming payment event to the log for the destination account. If a fee is due for this payment (for example, because of an overdrawn account or currency conversion), an additional outgoing payment event for the fees may be appended to the source-account log, and a corresponding incoming payment event may be appended to the log of a fees account. The original event ID is included in all of these generated events so that their origin can be traced.

4. Since the executor subscribes to the source-account log, the outgoing payment event will be delivered back to the executor. It uses the unique event ID to determine that it has already processed this payment and recorded it in its database.

5. The payment events on other accounts, such as the incoming payment on the destination account, are similarly processed by single-threaded executors, with a separate executor per account. The event processing is made idempotent by suppressing duplicates based on the original event ID.

6. The server handling the user's request may also subscribe to the source-account log and thus be notified when the payment request has been processed. This status information can be returned to the user.

If the payment executor crashes and restarts, it may reprocess some payment requests that were partially processed

Heterogeneous transactions continue to be problematic. By their very nature, they can only rely on a lowest common denominator of participating systems.

before the crash. Since the executor is deterministic, upon recovery it will make the same decisions to approve or decline requests, and thus potentially append duplicate payment events to the source, destination, and fees logs. Based on the ID in the events, however, it is easy for downstream processes to detect and ignore such duplicates.

Multipartition processing. In this payment example, each account has a separate log and thus may be stored on a different node. Moreover, each payment executor only needs to subscribe to events from a single account, and different executors handle different accounts. These factors allow the system to scale linearly to an arbitrary number of accounts.

In this example, the decision of whether to allow the payment request is conditional only on the balance of the source account; you can assume that the payment into the destination account always succeeds, since its balance can only increase. For this reason, the payment executor needs to serialize the payment request only with respect to other events in the source account. If other log partitions need to contribute to the decision, the approval of the payment request can be performed as a multistage process in which each stage serializes the request with respect to a particular log.

Splitting a “transaction” into a multistage pipeline of stream processors allows each stage to make progress based only on local data; it ensures that one partition is never blocked waiting for communication or coordination with another partition. Unlike multipartition transactions, which often impose a scalability bottleneck in distributed transaction implementations, this pipelined design allows OLEP systems to scale linearly.

Advantages of event processing. Besides this scalability advantage, developing applications in an OLEP style has several further advantages:

- Since every log can support many independent subscribers, it is easy to create new derived views or services based on an event log. For example, in the payment scenario of Figure 3, a new account log subscriber could send a push notification to a customer's smartphone if a certain spending limit on the customer's credit card is

reached. A new search index or view over an existing dataset can be built simply by consuming the event log from beginning to end.³

- If an application bug causes bad events to be appended to a log, it is fairly easy to recover: subscribers can be programmed to ignore the incorrect events, and any views derived from the events can be recomputed. In contrast, in a database that supports arbitrary insertions, updates, and deletes, it is much harder to recover from incorrect writes, potentially requiring the database to be restored from a backup.

- Similarly, debugging is much easier with an append-only log than a mutable database, because events can be replayed in order to diagnose what happened in a particular situation.

- For data-modeling purposes, an append-only event log is increasingly preferred over freeform database mutations; this approach is known in the domain-driven design community as *event sourcing*.² The rationale is that events capture state transitions and business processes more accurately than insert/update/delete operations on tables, and those state updates are better described as side effects resulting from processing an event. For example, the event “*student cancelled course enrollment*” clearly expresses intent, whereas the side effects “*one row was deleted from the enrollments table*” and “*one cancellation reason was added to the student feedback table*” are much less clear.

- From a data analysis point of view, an event log is more valuable than the state in a database. For example, in an e-commerce setting, it is valuable for business analysts to see not only the final state of the cart at checkout, but also the full sequence of items added to and removed from the cart, since the removed items carry information, too (for example, one product is a substitute for another, or the customer may return to buy a certain item on a later occasion).

- With a distributed transaction, if any one of the participating nodes is unavailable, the whole transaction must abort, so failures are amplified. In contrast, if a log has multiple subscribers, they make progress independently from each other: if one subscriber fails,

Debugging is much easier with an append-only log than a mutable database because events can be replayed in order to diagnose what happened in a particular situation.

that does not impede the operation of the publisher or other subscribers, so faults are contained.

Disadvantages of the OLEP approach. In the previous examples, log consumers update the state in data stores (the database and search index in Figure 2; the account balances and account statements in Figure 3). While the OLEP approach ensures every event in the log will eventually be processed by every consumer, even in the face of crashes, there is no upper bound on the time until an event is processed.

This means if a client reads from two different data stores that are updated by two different consumers or log partitions, then the values read by the client may be inconsistent with each other. For example, reading the source and destination accounts of a payment may return the source account after the payment has been processed, but the destination account before it has been processed. Thus, even though the accounts will eventually converge toward a consistent state, they may be inconsistent when read at one particular point in time.

Note that in an ACID context, preventing this anomaly falls under the heading of *isolation*, not *atomicity*; a system with atomicity alone does not guarantee that two accounts will be read in a consistent state. A database transaction running at “read committed” isolation level—the default isolation level in many systems including PostgreSQL, Oracle DB, and SQL Server—may experience the same anomaly when reading from two accounts.³ Preventing this anomaly requires a stronger isolation level: “repeatable read,” snapshot isolation, or serializability.

At present, the OLEP approach does not provide isolation for read requests that are sent directly to data stores (rather than being serialized through the log). Hopefully, future research will enable stronger isolation levels such as snapshot isolation across data stores that are updated from a log.

Case Study: The New York Times

The *New York Times* maintains all textual content published since the newspaper’s founding in 1851 in a single log

partition in Apache Kafka.⁶ Image files are stored in a separate system, but URLs and captions of images are also stored as log events.

Whenever a piece of content (known as an *asset*) is published or updated, an event is appended to this log. Several systems subscribe to this log: for example, the full text of each article is written to an indexing service for full-text search; various cached pages (for example, the list of articles with a particular tag, or all pieces by a particular author) need to be updated; and personalization systems notify readers who may be interested in a new article.

Each asset is given a unique identifier, and an event may create or update an asset with a given ID. Moreover, an event may reference the identifiers of other assets—much like a normalized schema in a relational database, where one record may reference the primary key of another record. For example, an image (with caption and other metadata) is an asset that may be referenced by one or more articles.

The order of events in the log satisfies two rules:

- ▶ Whenever one asset references another, the event that publishes the referenced asset appears in the log before the referencing asset.
- ▶ When an asset is updated, the latest version is the one published by the latest event in the log.

For example, an editor might publish an image and then update an article to reference the image. Every consumer of the log then passes through three states in sequence:

1. The old version of the article (not referencing the image) exists.
2. The image also exists but is not yet referenced by any article.
3. The article and image both exist, with the article referencing the image.

Different log consumers will pass through these three states at different times but in the same order. The log order ensures that no consumer is ever in a state where the article references an image that does not yet exist, ensuring referential integrity.

Moreover, whenever an image or caption is updated, all articles referencing that image need to be updated in caches and search indexes. This can easily be achieved with a log con-

sumer that uses a database to keep track of references between articles and images. This consistency model lends itself very easily to a log, and it provides most of the benefits of distributed transactions without the performance costs.

Further details on the *New York Times*'s approach appear in a blog post.⁶

Conclusion

Support for distributed transactions across heterogeneous storage technologies is either nonexistent or suffers from poor operational and performance characteristics. In contrast, OLEP is increasingly used to provide good performance and strong consistency guarantees in such settings.

In data systems it is very common for logs (for example, write-ahead logs) to be used as internal implementation details. The OLEP approach is different: it uses event logs, rather than transactions, as the *primary application programming model* for data management. Traditional databases are still used, but their writes come from a log rather than directly from the application. This approach has been explored by several influential figures in industry, such as Jay Kreps,⁴ Martin Fowler,² and Greg Young under names such as event sourcing and CQRS (Command/Query Responsibility Segregation).^{1,7}

The use of OLEP is not simply pragmatism on the part of developers, but rather it offers a number of advantages. These include linear scalability; a means of effectively managing polyglot persistence; support for incremental development where new application features or storage technologies are added or removed iteratively; excellent support for debugging via direct access to the event log; and improved availability (because running nodes can continue to make progress when other nodes have failed).

Consequently, OLEP is expected to be increasingly used to provide strong consistency in large-scale systems that use heterogeneous storage technologies.

Acknowledgments. This work was supported by a grant from The Boeing Company. Thanks to Pat Helland for feedback on a draft of this article. □

Related articles on queue.acm.org

Consistently Eventual

Pat Helland

<https://queue.acm.org/detail.cfm?id=3226077>

Evolution and Practice: Low-latency

Distributed Applications in Finance

Andrew Brook

<https://queue.acm.org/detail.cfm?id=2770868>

It Isn't Your Father's Real Time Anymore

Phillip Laplante

<https://queue.acm.org/detail.cfm?id=1117409>

References

1. Betts, D., Domínguez, J., Melnik, G., Simonazzi, F. and Subramanian, M. *Exploring CQRS and Event Sourcing*. Microsoft Patterns & Practices, 2012; <http://aka.ms/cqrs>.
2. Fowler, M. Event sourcing, 2005; <https://martinfowler.com/eaaDev/EventSourcing.html>.
3. Kleppmann, M. *Designing Data-intensive Applications*. O'Reilly Media, 2017.
4. Kreps, J. The log: What every software engineer should know about real-time data's unifying abstraction. LinkedIn Engineering, 2013; <https://bit.ly/199MwY>.
5. Schneider, F.B. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys* 22, 4 (1990), 299–319; <https://dl.acm.org/citation.cfm?doid=98163.98167>.
6. Svingen, B. Publishing with Apache Kafka at the *New York Times*, (Sept. 5 2017); <https://open.nytimes.com/publishing-with-apache-kafka-at-the-new-york-times-7f0e3b7d2077>.
7. Vernon, V. *Implementing Domain-driven Design*. Addison-Wesley, 2013.

Martin Kleppmann is a distributed-systems researcher at the University of Cambridge and author of *Designing Data-Intensive Applications* (<http://dataintensive.net/>). Previously he was a software engineer, cofounding two startups and working on large-scale data infrastructure at LinkedIn.

Alastair R. Beresford is a reader in computer security at the University of Cambridge. His work examines the security and privacy of large-scale distributed computer systems, with a particular focus on networked mobile devices.

Boerge Svingen a director of engineering at the *New York Times*. He was a founder of Fast Search & Transfer (alltheweb.com, FAST ESP) as well as a founder and CTO of Open AdExchange.



Cloud-delivery networks could dramatically improve blockchains' scalability, but clouds must be provably neutral first.

BY ALEKSANDAR KUZMANOVIC

Net Neutrality: Unexpected Solution to Blockchain Scaling

THERE IS A growing expectation, or at least a hope, that blockchains possess a disruptive potential in numerous domains because of their decentralized nature (that is, no single entity controls their operations). Decentralization comes with a price, however: blockchains do not scale—they are incapable of processing a large, or even moderate, number of transactions in a timely manner. For example, bitcoin processes three transactions per second.

The root of the problem—and the limiting factor for blockchains—is a trustless peer-to-peer network

model, in which information must be suboptimally propagated to—and validated at—every hop in the network. Undoubtedly, cloud-delivery networks (for example, Akamai or YouTube), which resolved similar performance challenges in other domains (for example, Web and video delivery), could help scale blockchains as well. The problem is that such large centralized infrastructures disturb the decentralized nature of blockchains, hence eliminating their disruptive potential. The question is, can cloud-delivery networks be used to scale blockchains without upsetting their decentralized nature? The answer is positive, and the key to the solution lies in an advanced version of an existing concept: net neutrality.

Blockchain and the cryptocurrency revolution initiated by bitcoin in 2008⁸ are thriving. The market capitalization of prominent cryptocurrencies, while highly volatile, continues to be measured in hundreds of billions of dollars. A unique feature of blockchains is the lack of centralized administration. They rely on third-party mediation, (that is, a global peer-to-peer network of participants who validate and certify all transactions). Given the purely distributed and decentralized design of blockchains, many people believe that such systems have a disruptive potential in other areas beyond cryptocurrencies, including health care, government, manufacturing, retail, insurance, the Internet of things, and the sharing economy. Numerous high-tech companies, big and small, are closely watching the blockchain space, analyzing how the new technology could affect their existing or future operations.

A major problem for blockchains is scalability. The blockchain system throughput is measured in the number of transactions per second (TPS) a system can support. Bitcoin's current average throughput of three TPS compares to 2,000TPS average throughput in Visa's centralized system, 4,000TPS daily peak, and 56,000TPS maximum



capacity. Without scalability, cryptocurrency systems will hardly become mainstream, and blockchains are unlikely to realize their disruptive potential in any other areas.

What Is a Blockchain?

A blockchain is a public distributed ledger that stores all past transactions and is fundamentally a type of database created and shared by multiple (tens of thousands) nodes connected in a peer-to-peer network. To achieve consensus regarding the correct copy of the database, certain rules about writing to the database must be imposed. Although the rules may vary, they generally include the following:

► *Transactions must be valid.* A transaction, which typically passes some amount of cryptocurrency from one user to another, must contain digital

signatures from the participants for authentication purposes.

► *Transactions must be added in sequence.* Transactions are not added to the ledger individually; rather, they are added in batches, known as *blocks*. For example, the bitcoin blockchain requires that each new block contain a solution of a hashing “puzzle” that is unique to the combination of the last block of transactions on the chain and the current block being added.

► *Adding blocks to the blockchain is expensive and competitive.* Parties who want to add blocks to the blockchain must invest either cryptocurrency or the computing power necessary to solve a cryptographic puzzle (for example, the hashing puzzle required by bitcoin). Such a party is called a *miner*, and the process of adding new blocks to a blockchain is referred to as *mining*.

► *The longest blockchain available is the up-to-date version.* When combined with the previous rules, this makes a blockchain very expensive to forge successfully. Even copying an existing blockchain and attempting to modify the last few blocks can quickly become prohibitively expensive. Once blocks get sufficient confirmations on the network, deleting or modifying a block becomes mathematically improbable. Effectively, transactions can only be added to the blockchain; they can never be deleted.

► *Independent verification is required.* A node should be able to verify independently that all the previous rules have been complied with when it inspects a copy of the blockchain database. If each user can verify the blockchain independently, this allows all users to come to a consensus about the correct blockchain.

► Adding blocks to the blockchain is rewarding. Because writing blocks to the blockchain is hard, not all nodes will participate in the process. Many users will create transactions but then just request that they be written to the network, often offering a fee as an incentive. In addition, miners are rewarded with the ability to distribute new cryptocurrency to themselves whenever they win a round of the mining process and get the chance to add a block to the blockchain.

Forks can happen, but they are resolved via the longest blockchain rule. Reaching a consensus on the blockchain is not immediate, and sometimes a fork (a different copy of the database) may arise in the blockchain, where different versions of the blockchain's public ledger coexist, and diverge after a common history. By se-

lecting the longest blockchain on the network, however, nodes work to resolve these forks.

The Blockchain Scalability Problem

Before explaining the blockchain scalability problem, let's first see how it manifests in reality. Figures 1 and 2 show the transaction backlogs for bitcoin and ethereum, two leading cryptocurrencies. You can see that tens of thousands of transactions are regularly waiting to be processed by a blockchain. To increase the likelihood of being selected by miners and get "on-chain," users increase the size of the fees they (voluntarily) include in their transactions. As a result, fees are far from negligible, and they can grow considerably during times of high congestion.

To understand where the bottleneck is, let's compute the block-

chain throughput first. The system throughput depends directly on two parameters: the block size B (that is, the number of bytes that can contain transactions in each block), and the interblock time interval T (that is, the average time required for the system to mine a new block). In bitcoin, $B = 1$ MB and $T \sim 600$ seconds, which allows approximately three TPS. On-chain throughput can be improved through the following options: increase B to include more transactions; reduce T so that blocks are mined at a higher rate; or both. The problem is that these parameters cannot be arbitrarily changed, as detailed later.

Obviously, it is the blockchain's distributed nature that causes the problems. Indeed, if blocks and transactions were to be instantly propagated among nodes, immense blocks could be mined at a rapid pace, until the limitation of designated processing units and flash storage arrays was reached.⁴ In reality, however, blockchain nodes—tens of thousands of them or more—are distributed around the world. Hence, *the network is the bottleneck*.

Nodes in a blockchain network communicate in a peer-to-peer fashion. This, unfortunately, works against the goal of high-throughput, low-latency communication in the following ways:

► The information is transmitted from one node to another; hence, it takes multiple hops for the information to be propagated through the entire network. Given that each node in the network is distrustful of every other node, the information being propagated must be independently validated at every hop. This typically involves a cryptographic operation at each hop, which adds latency and hurts throughput.

► The performance *variance* of nodes in a blockchain network is high, which means that a single slow node on the critical path can inflate the propagation time.

► Finally, nodes in a peer-to-peer network are randomly formed; hence, they are not organized for optimal propagation. This means that data travels through suboptimal paths in the network.

As a result, the average time needed to propagate a 1MB block to 90% of the nodes in the bitcoin network

Figure 1. Bitcoin transactions backlog.

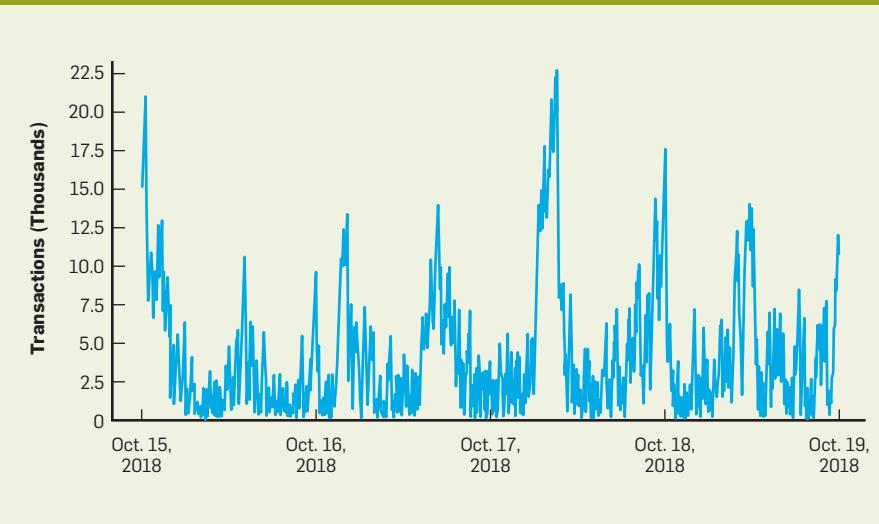


Figure 2. Ethereum transactions backlog.

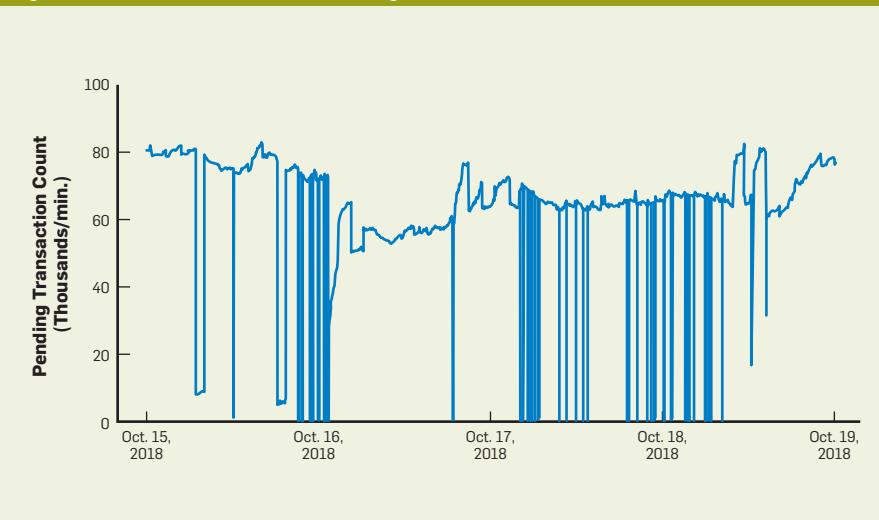
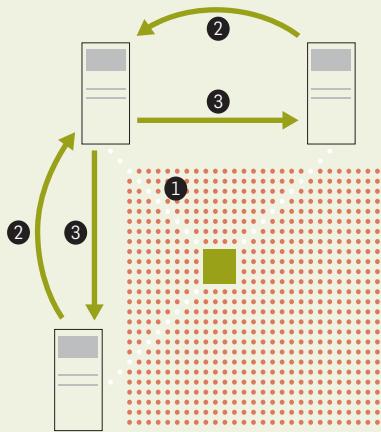
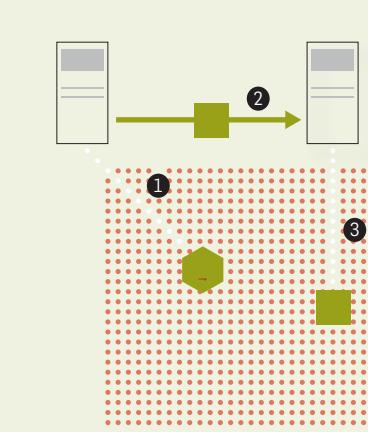
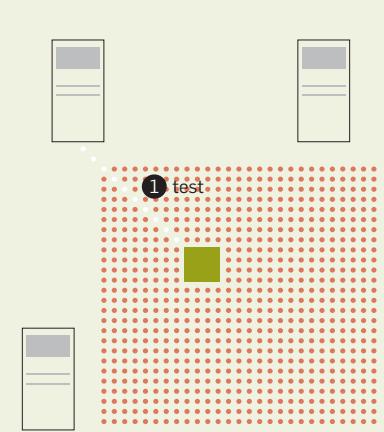


Figure 3. Encrypted block.**Figure 4. Indirect relay.****Figure 5. Test block.**

is 11.6 seconds, which was the average propagation time observed in March, 2017.¹ This is, unfortunately, just a part of the problem. It has been shown, both in theory⁷ and in practice,^{4,5} that increasing the block size B by a factor of X also increases the time required for a block to propagate by the same factor X . Similarly, decreasing the interblock interval T by a factor of X has exactly the same effect. This means that the block-propagation time increases proportionally with each of these two parameters.

For example, increasing the block size tenfold would increase the block-propagation times tenfold as well, making them longer than 100 seconds. Likewise, increasing the block size by a factor of 100 would lead to block-propagation times longer than 1,000 seconds. Such a propagation time exceeds the time between blocks, *causing a fork every time a new block is mined*. Indeed, in this scenario, forks will not be resolved by the mining of the following block, and instead the blockchain will unravel to forks, and forks-of-forks, and forks-of-forks-of-forks, until nodes and miners do not know which fork is the “true” chain—and the blockchain breaks. This is the blockchain scalability problem caused by the networking bottleneck.

Cloud-Delivery Networks

Cloud-delivery networks were very successful in resolving performance problems on the Internet. Such networks distribute content via an im-

mense infrastructure that can consist of hundreds of thousands of servers worldwide (for example, Akamai). In addition, they perform extensive network and server measurements and use them to redirect clients to nearby servers. This helps the Internet operate at the immense scale it does.⁶ As an example, YouTube alone has more than a billion users, and a whopping 70% of North American Internet traffic in peak evening hours comes from streaming video and audio sites such as Netflix and YouTube. This would not be possible without cloud-delivery networks.

This is in striking contrast to the state of affairs with blockchains. Indeed, as explained earlier, propagating a 1MB block through a blockchain network is a time-consuming task, and increasing the block size much more could lead to unrecoverable problems. Yet cloud-delivery networks manage to send terabytes of data every second, and that is considered ordinary. Can such networks be used to scale blockchains?

Undoubtedly, cloud networks could improve the performance of blockchains. The issue is trust. In a blockchain ecosystem, a node does not trust its immediate peers, so how will it trust a cloud network, which is far more powerful than any individual node? Cloud-delivery networks are centralized systems that can censor transactions, blocks, or miners of a blockchain network. For example, the cloud-delivery network administrators may reject blocks

that contain transactions among unauthorized parties, or blocks mined by unauthorized miners, according to their own policies, business interests, or legal requirements.

Thus, the key question is whether it is possible to make cloud-delivery networks trustless, such that they can be used to scale blockchain networks, without the ability to exercise censorship and other powers previously mentioned here. This concept is called *provable net neutrality*. Without diving into formalism, this article outlines the key properties associated with this concept.

First, the network should not be able to censor information based on the content of blocks. Second, the network should not be able to censor nodes. Third, nodes should be able to audit these properties continuously, and in case of network misbehavior, to abandon and replace the network. How do you enable such properties?

A Provably Neutral Blockchain-Distribution Network

Consider a cloud-distribution network that aims to enable blockchain systems (not necessarily cryptocurrencies only) to scale to thousands of on-chain transactions per second. Moreover, it aims to provide scalability to numerous cryptocurrencies and blockchains simultaneously, using a global infrastructure to support distributed blockchain systems in a provably neutral fashion. This is known as a BDN (blockchain distribution network). This section outlines

the system's trust model and then describes the key mechanisms necessary to fulfill the neutrality properties.

Reversed trust model. BDN's trust model is based on two observations: first, long block-propagation times will not ever allow trustless peer-to-peer blockchains (for example, bitcoin), to scale substantially; second, small centralized systems scale very well by placing trust in a small subset of participants and passing them the control over the transactions included in the blockchains (for example, Ripple and EOS).

Such centralization, however, defeats the single most notable aspect of blockchains: the distribution and decentralization of control over transactions. Providing control over a blockchain's transactions to a limited number of participants allows participants to collude, censor, and discriminate among users, nodes, and miners. A limited participant set also reduces the number of nodes a malicious actor has to compromise to control the system.

BDN addresses this trade-off by reversing the direction of trust in centralized systems. While centralized systems place trust in a subset of nodes to enable scalability, BDN enables scalability by using a small set of servers that place trust in the entire network instead. The resulting system can enable scaling, yet nodes need not place any trust in the BDN. Instead, the BDN blindly serves the nodes, without knowledge of the blocks it propagates, their origin, or their destination. Moreover, the nodes it serves constantly audit its behavior, and it is incapable of discriminating against individual nodes, blocks, and transactions. While such a design places the BDN at a disadvantage compared with the nodes it serves, its robustness allows it to withstand dishonest and malicious behavior.⁷

Provable network neutrality. In short, BDN can only propagate all blocks to all the blockchain nodes fairly, and it is incapable of discrimination because of the auditing performed by the blockchain nodes, still connected in a peer-to-peer fashion.

Encrypted blocks. To prevent BDN from stopping the propagation of any block based on its content (see Figure 3), blocks are propagated after being

encrypted (step 1). BDN's encryption also alters the block size, hiding the number of transactions and their total size. After the block has been propagated, the receiving peer nodes inform the sender by sending a hash of the block (step 2). Finally, a block's encryption key is revealed and is propagated directly over the blockchain peer-to-peer network (step 3). The encryption key's tiny size, only several bytes, allows it to quickly propagate directly over the peer-to-peer network, and BDN is powerless to stop it.

Indirect replay. To ensure BDN is not preventing individual nodes from propagating their blocks, nodes do not have to propagate blocks directly to BDN (see Figure 4). In case a block is not propagated by BDN (step 1), the sending node will propagate it to a peer on the peer-to-peer network (step 2), which will relay it to BDN (step 3), obscuring the block's origin from BDN. For example, a node that mined a block in China could relay it to a node in Europe, which then sends the block via BDN. In addition to relaying blocks indirectly to BDN, nodes may request their peers to relay to them incoming blocks arriving from BDN. This ensures that BDN cannot discriminate against nodes through late delivery of blocks since nodes are not required to interact directly with BDN in order to benefit from its service.

Auditing via test blocks. While BDN is oblivious to which node originates each block, it may attempt to prevent or stall blocks arriving from some subset of nodes, affecting all the blocks they relay. In order to detect and prevent such behavior, nodes must be capable of continuously monitoring BDN's service. Such monitoring is achieved by allowing nodes to send encrypted invalid blocks, *test blocks*, directly to BDN (Figure 5) and measuring the time required for peers to report the arrival of the test blocks. BDN is unable to employ discriminatory policies over valid blocks alone and faithfully propagate test blocks, since the two are indistinguishable until their keys are published.

Thus, by using traffic encryption and indirect traffic relaying, and by explicitly auditing BDN, blockchain nodes are capable of restricting the BDN's ability to misbehave, effectively

decoupling a BDN operator's authority from the BDN infrastructure. If a BDN ceases to deliver blocks completely, or delivers blocks only to a small subset of nodes, the blockchain nodes can abandon the BDN.

Since nodes are constantly using test blocks to infer the best source from which to receive blocks, any node that BDN discriminates against will simply be receiving blocks from its peers. Thus, if BDN is maliciously discriminating against many or all peers, peers will simply form their own peer-to-peer network until a different system takes its place. Additionally, if the discrimination is caused by a large-scale system failure, the peers will return to using BDN once the failure is resolved.

Performance

In essence, BDN deploys a *broadcast primitive*, meaning it enables efficient transmission of data from a single source node to all other nodes in a blockchain network. In contrast to a peer-to-peer network, where each blockchain node is connected to numerous other nodes, often spread around the world, a blockchain node replaces this one-to-many communication with one-to-one communication. This is because a blockchain node connects to a single BDN server.

With large TPS rates, using a single connection vs. many connections helps with scaling. Necessarily, blockchain nodes still need to be connected in a peer-to-peer network to audit the BDN effectively. The bulk of the data, however, is transmitted to and from the BDN. Following are several ways in which BDN helps scale blockchains.

Transactions caching. In a blockchain system, such as bitcoin or ethereum, each node receives transactions twice: once as raw transactions when initially propagated through the network, and the second time when they are included in blocks. A BDN can effectively distribute transactions through the cloud, index them, and then utilize indexes (instead of raw transactions) when transmitting blocks. This effectively compresses the block size by more than 100 times, given that the raw transaction is approximately 500 bytes long, while an index can be four bytes or less.

Transactions caching is an existing idea in the blockchain ecosystem, and it has been adopted by certain projects,³ but it has been deployed only by endpoints, not by the network. As a result, given that not all transactions in a pure blockchain system reach all endpoints,⁸ even a slight desynchronization can lead to significant increases in the block size (not all transactions are “compressed”); hence, the performance suffers. In contrast, BDN effectively transmits and indexes blockchain transactions.

Cut-through routing. In contrast to blockchain nodes, BDN *cannot* check the validity of blocks flowing through the network, because they are encrypted. This helps with swift transmission of blocks through the network. In particular, before a BDN node receives all bits of a block, the BDN can already start transmitting received bits of a block to the rest of the network. This is called *cut-through routing*, and it has been widely adopted in network switches for decades. Still, it can significantly speed up the data transmission, particularly when blocks are large.

Transactions incast problem. Transactions need to be broadcast in a blockchain network. In the absence of a BDN, at higher TPS rates this creates a so-called *incast* problem: the *same* transactions are received at a high rate from multiple sources. This can significantly affect a node’s resources and impact overall blockchain performance. BDN eliminates this problem given that the bulk of data, including transactions, is propagated to and from a single BDN server.

Related Blockchain Scaling Attempts

Alternative approaches to scaling blockchains are described here:

Off-chain scaling solutions. One alternative approach (for example, the Lightning Network), which uses *off-chain transactions*, aims to reduce some of the redundancy on the main blockchain. Generally speaking, an off-chain scaling solution will open up a payment channel between two parties (that is, have the parties exchange funds while keeping track of intermediate balances) and then post a settlement transaction on the blockchain.

Such a solution is agnostic to BDN’s proposition. Indeed, an off-chain scaling solution still fundamentally requires on-chain capability. Also, the potential scaling benefits are *multiplicative*. If the underlying blockchain can support 1,000 times the number of transactions as before thanks to BDN, and if off-chain transactions increase the throughput by another factor of 1,000, then that blockchain’s throughput has increased by six orders of magnitude.

On-chain scaling solutions. On-chain scaling solutions typically involve modifying the consensus protocol in some way to achieve higher throughput. One such approach, known as *sharding*, splits the blockchain into several smaller shards, which are maintained and interleaved such that the blockchain’s original security properties are preserved while requiring only a full node to track one shard instead of the full blockchain. Numerous other ideas exist in this space.² While these approaches show potential, their robustness, security, and usability in practice remain to be seen.

Still, all on-chain scaling solutions will perform strictly better with a faster network layer, and this is where BDN improves their performance. Indeed, in every distributed consensus protocol, every protocol-compliant node must reach the same decision. Thus, every such peer must obtain information about each transaction in the system, independently from the consensus protocol. BDN focuses on this particular problem, which is fundamentally a broadcast problem, since every valid piece of information must be propagated to every peer in the system. BDN is thus agnostic to a native consensus protocol, and it is capable of boosting the performance, often dramatically, of *any* blockchain.

Conclusion

Provably neutral clouds are undoubtedly a viable solution to blockchain scaling. By optimizing the transport layer, not only can the throughput be fundamentally scaled up, but also the latency could be dramatically reduced. Indeed, the latency distribution in today’s data centers is already biased toward *microsecond* times-

scales for most of the flows, with millisecond timescales residing only at the tail of the distribution. There is no reason why a BDN point of presence would not be able to achieve a similar performance.

Adding dedicated optical infrastructure among such BDN points of presence would further alleviate throughput and reduce latency, creating the backbone of an advanced BDN. The key to this vision, however, lies in establishing trust by the blockchain ecosystem into the underlying networking infrastructure. This, in turn, is achieved by decoupling authority from infrastructure via a provably neutral network design. □

Related articles on queue.acm.org

Research for Practice: Cryptocurrencies, Blockchains, and Smart Contracts

Arvind Narayanan and Andrew Miller

<https://queue.acm.org/detail.cfm?id=3043967>

Better, Faster, More Secure

Brian Carpenter

<https://queue.acm.org/detail.cfm?id=1189290>

A Purpose-Built Global Network: Google’s Move to SDN

<https://queue.acm.org/detail.cfm?id=2856460>

References

1. Bitcoinstats.com. Data propagation; www.bitcoinstats.com/network-propagation/.
2. Cachin, C. and Vukolic, M. Blockchain consensus protocols in the wild. arXiv, 2017; <https://arxiv.org/pdf/1707.01873.pdf>.
3. Clifford, A., Rizun, P., Suisani, A., Stone, A. and Tschipper, P. Towards massive on-chain scaling: presenting our block propagation results with Xthin, 2016; https://medium.com/@peter_r/towards-massive-on-chain-scaling-presenting-our-block-propagation-results-with-xthin-da54e55dc0e4.
4. Croman, K. et al. On scaling decentralized blockchains. *International Conference on Financial Cryptography and Data Security*. Springer, 2016, 106–125.
5. Decker, C., Wattenhofer, R. Information propagation in the Bitcoin network. In *Proceedings of the 13th IEEE International Conference on Peer-to-Peer Computing*, 2013; <https://ieeexplore.ieee.org/document/6688704>.
6. Internet Live Stats; <http://www.internetlivestats.com/one-second/>.
7. Klarman, U., Basu, S., Kuzmanovic, A. and Sirer, E.G. bloXroute: A scalable trustless blockchain distribution network; <https://bloxroute.com/wp-content/uploads/2018/03/bloXroute-whitepaper.pdf>.
8. Nakamoto, S. Bitcoin: a peer-to-peer electronic cash system. Bitcoin.org, 2008; <https://bitcoin.org/bitcoin.pdf>.

Aleksandar Kuzmanovic is a professor of computer science at Northwestern University, Evanston, IL. He is a cofounder of bloXroute Labs, a blockchain scaling startup, where he serves as a chief architect.

contributed articles

DOI:10.1145/3319422

A positive image would inspire the capable but underrepresented who might otherwise give up on computing.

BY FAY COBB PAYTON AND ELENI BERKI

Countering the Negative Image of Women in Computing

DESPITE INCREASED KNOWLEDGE about gender (in) equality,^{7,27,38} women in STEM disciplines are still portrayed in stereotypical ways in the popular media. We have reviewed academic research, along with mainstream media quotes and images for depictions of women in STEM and women in computing/IT. We found their personality and identity formation continues to be influenced by the personas and stereotypes associated with role images seen in the media. This, in turn, can affect women's underrepresentation and career participation, as well as prospects for advancement in computing fields.

The computer science Degree Hub¹⁵ in 2014 published its list of the 30 most influential, living

computer scientists, weighing leadership, applicability, awards, and recognition as selection criteria. The list included only one female, Sophie Wilson, a British computer scientist best known for designing the Acorn Micro-Computer, the first computer sold by Acorn Computers Ltd. in 1978. A fellow elected to the prestigious Royal Society, Wilson is today the Director of IC Design at Broadcom Inc. in Cambridge, U.K., listed as number 30 of the 30 on the list.

Several other observations are notable about the list. The other 29 are all male, with one from Mexico—Miguel de Icaza—and one from Japan—Yukihiro Matsumoto. In general, 15 of the 30 at the time worked in industry, at major tech organizations, including Dropbox, Facebook, Microsoft, PayPal, and Google. Others were academic scholars, along with several entrepreneurs. However, women tend to seek alternatives to traditional computing/IT industry and academic careers by establishing IT-related or entrepreneurial venues.^{38,40} Nonetheless, the fact remains that with only one female on the list and one male from a traditionally underrepresented majority minority country does communicate as to who influences, controls, and ultimately is expected to participate in computing. Computing awards and recognitions often seem to derive from the well-recognized relationship between masculinity and technology, particularly in terms of what are considered technical skills and masculine identity.^{2,7} Such acknowledgments in the media often function as an unofficial recruitment policy, influencing academic and career choices.

» key insights

- The lack of sustained gender and ethnic participation persists in computing.
- Media images produce mental models of who participates and what participation even looks like.
- One common image is that men can be ordinary while women must be exceptional, and women of color must be better than exceptional.



Recent popular media attention has begun to focus on the lack of gender and ethnic diversity in Silicon Valley. The experience of women in these settings was documented by Mundy²⁸ and is available in the online appendix “Media Representations of Women in Computing Through Text and Images” (dl.acm.org/citation.cfm?doid=3319422&picked=formats).

Mundy²⁸ pointed out that tech companies have spent millions of dollars to recruit and improve the workplace environment and conditions only to find little has changed for women in tech, even today. Mundy²⁸ reflected on women and ethnic minorities abandoning tech careers, misogynistic and “bro” culture, hostile workplace conditions, lack of access to leadership roles, and biases as barriers to inclusion.^{4,39}

The Valley is home to Facebook, Google, Apple, Yahoo, LinkedIn, Twitter, and many other organizations that recruit from the talent pool of undergraduate students in nearby universities who earn degrees in information systems/sciences and computer science. Relevant to the talent-pool recruitment policies and results is the data reported by the National Science Foundation,²⁹ showing several professional trends:

Black and Latina women. Black and Latina women combined (in 2016) earned only 4.4% of all undergraduate computing degrees in the U.S.; women overall earned 28.5%, 25.1%, and 18.1% of all bachelor-level computer science degrees in 1995, 2004, and

2014, respectively. Computer science was termed a “low participation” field for these underrepresented minorities while Latina women’s completion of undergraduate computer science degrees in the U.S. was flat between 1995 (1.75%) and 2014 (1.79%). For black women, these numbers were 5.10% and 2.61%, respectively.

For 2012, the National Science Foundation reported 70% of all black computing degree earners were male, and 81% of Latino computer science degree earners were male. Within the tech workforce, 25% of all computing professionals were female, with only 4% of them African American (3%) or Latina (1%).

Similar numbers were reported by The Chartered Institute of the U.K. in its 2014 *Women in IT Scorecard*.¹⁰ Of the 753,000 people working in the IT sector at the time, only 20% were women. In 2013, 11% of IT specialists were women. Women were much more likely to hold technician/engineer-grade positions than men (34% vs. 20%, respectively)¹⁰ and less likely to be working in “professional,” or primarily development-related, occupations (46% vs. 57%).

As early as the 1980s, barriers to work participation for women and underrepresented minorities in science, along with the need for intervention programs and strategies for increasing participation, were being documented;²¹ for example, an earlier (1978) observation by Austing et al.⁵ about the dearth of women and underrepresent-

ed minority computer science faculty were documented in *Communications*.⁵ Yet, a comparison of 2014 statistics and facts²⁹ reported a downward trend for participation by female and underrepresented groups in computing/IT careers despite the counternarratives of these statistics from the 1960s to the early 1980s. Abbate¹ documented the history of computer science in 2012, noting that programming was seen as “women’s work” in the years following World War II. The media and entertainment industries have taken note of women programmers’ achievements (in such movies as *Ghostbusters*, *Hunger Games*, and even *Star Trek*) despite common stereotypes and other obstacles that largely ignored black women in the narrative, until the 2016 release of *Hidden Figures*. Moreover, the image of the computer geek emerged in the 1980s with depictions of “hackers” and “nerds” in the mass media, depicting these characteristics as masculine.

Sociocultural Gender Barriers

Stereotyping refers to the attributes the general public thinks characterize a particular group. Studies of systemic stereotyping dimensions refer to traditional content of what people in a society think of others and try to reveal the systemic (holistic) reasons and mechanisms involved in stereotyping. Researchers, including Fiske et al.,¹⁴ categorized four types of stereotypes and their systemic dimensions resulting from feature combinations of perceived warmth and competence, as outlined in the figure here.

Accordingly, social status¹⁴ predicts competence, and competence yields only modest warmth. Some stereotyped groups (such as the elderly) are depicted as being unable and useless, whereas others (such as Asians) are respected for their excessive, yet potentially threatening, dedicated competence. These pairings can also affect how women view themselves and how the broader culture perceives gender status and competence, as well as computing skills and capabilities.

Fiske et al.’s classification of status¹⁴ is not without some qualification. Even Hofstede’s classification²⁰ needs to be reconsidered and enriched³³ to acknowledge that stereotypes are distinguished by national and broader social

Societal stereotypes.

		Competence	
		Low	High
Warmth	High	Paternalistic stereotype low status, not competitive (such as housewives, elderly people, and disabled people)	Admiration high status, not competitive (such as the in-group and close allies)
	Low	Contemptuous stereotype low status, competitive (such as welfare recipients, poor people)	Envious stereotype high status, competitive (such as Asians, Jewish people, rich people, feminists)

culture, considering several persistent geopolitical factors:

Power distance. The extent hierarchies and unequal power distribution are accepted;

Uncertainty avoidance. The extent a society feels threatened by ambiguous situations and tries to avoid them by establishing rules, believing in absolute truths, and refusing to tolerate deviance;

Masculinity vs. femininity. The relationships among masculine assertiveness, competitiveness, and materialism as opposed to the feminine concern for quality of relationships, nurturing, and social well-being; and

Individualism vs. collectivism. The relationship between individual independence and the group's collective interdependence.

Other researchers, including Cheryan et al.,¹² have examined the stereotypes of professionals in computing to determine if changing them can affect women's interest in the field. Cheryan et al.¹² reported computer scientists are conventionally viewed as introverted, narrowly focused, and "non-feminine." However, when these stereotypes were dispelled by exposing young women in the study to media representations of non-stereotypical female computer scientists, they expressed more interest in computing. Cheryan et al.¹² concluded stereotypes about certain academic disciplines can influence who chooses to enter them, and intentional media messages that dispel the stereotypes can help attract more women into the field.

Women themselves can promote female stereotypes in tech and exhibit biases that produce self-defeating normalized thinking. Researchers, including Thornham and MacFarlane,³⁶ have described how teenage and adult women portray themselves in a "habitual feminine position of incompetence" regarding tech. Women participating in such surveys often evoke a more-traditional and less-progressive view of femininity and thereby restrict their own performance. Women in these instances use their gender as a tool to distance themselves from tech and demonstrate disinterest.³⁶ Gender stereotyping exposures should thus be addressed much earlier and more broadly in the lives of female



students in light of the education and media images that inform and influence mental models held by all, regardless of gender.

The influence of stereotypes is not limited to women in computing but can affect women in leadership roles as well. Researchers, including Simon and Hoyt,³⁴ conducted two experimental studies in 2012 to understand how media images that portray women in non-stereotypical roles can influence women's gender-role mind-sets, finding women who see media images of females in non-stereotypical roles subsequently reported less negative self-perception, greater leadership aspiration, and stronger belief that women can take on nontraditional gender roles.³⁴ Such attitudes have powerful implications for the role media, both on- and offline, can play in promoting gender participation and broadening participation efforts, as well as women taking on leadership roles in the field.

Stereotyping in Computing and IT

Because our research motivation is to explicate and report role stereotyping phenomena beyond statistical data, we consider aspects of how gender is depicted in the media. To explore biases in the field of computing, it is useful to start with the connection between masculinity and technology. The connection is not inherent in gender differences but rather a result from the historical and sociocultural construction of gender.¹² We thus elaborate on the background that is necessary to understand this perspective through media sources.

We used three such sources to explore the role of images of women in computing: interviews from the *Huffington Post*,¹⁷ stories on National Public Radio (NPR),³⁵ and reports from the U.K. Women in Tech Council.^{9,10,38} They provide publicly available evidence to assess how women are depicted in computing, as selected by us based on their reporting of women in computing, as well as their broad media reach and engagement. In 2012, the *Huffington Post* website had 84 million unique visitors worldwide,¹⁷ and NPR had 25.6 million unique monthly listeners through more than 1,000 public U.S. radio stations.³⁵ The U.K. Women in Tech Council is a member organization consisting of 850 companies that collectively employ more than 700,000 professionals in the U.K. and focus on issues of gender and leadership in the technology sector.³⁸

Females, computing, and image interpretation. Research studies, including by Gioia et al.,¹⁸ provided guidance for determining qualitative rigor in the inductive research we report here. Researchers, including Gioia et al.,¹⁸ demonstrate how first-order concepts in qualitative data appear to fly off in all directions and even become unmanageable. Second-order themes can then follow and explain the phenomena (such as images of women in computing) under investigation. Lastly, the aggregated dimensions can be presented as a visual representation to show a progression from raw data to higher-order themes, including bias, stereotypes, and "intersectionality."

We sought out and identified in-

sightful quotes from the following media posts under the following headlines:

"Is The Stereotype That 'Women Can't Be Geniuses' Causing Gender Gaps?" (*Huffington Post*, 2015);¹⁷

"The Forgotten Female Programmers Who Created Modern Tech" (National Public Radio blog, 2014);³⁵ and

"The U.K. Women in Tech Council (British Computing Society—Women in IT Interviews)" (2014).³⁸

Media posts and projected images. To help comprehend the theoretical analysis of the role of cultural stereotypes, as outlined earlier, we include the media-source quotes and relevant second-order themes in a table in the online appendix while offering images of women online in IT and computing, along with qualitative rigor.¹⁸ Text analyses with second-order themes, including media/tech acculturation, entrepreneurship alternatives, shifts in role/identity work, and the intersectionality associated with women of color, at the intersection of race and gender. These second-order themes map to the literature related to gender and tech.

The table in the online appendix outlines the themes and clichés concerning women in tech that are dominant in the mainstream media and fall into three categories: extraordinary achievement; intellectual differences; and hypermasculinity in traditional

old-boys-club personas. The following section relates these themes to challenging theories concerning women's historical underrepresentation and participation in computing/IT.

The Media and Underrepresentation

Beyond media posts and raw data in the table in the online appendix, the research literature also provides evidence of stereotypical images associated with women and underrepresented minorities in computing and IT. The following research demonstrates the exceptional or extraordinary depictions of intellectual differences and hypermasculine descriptions we found in scholarly works, even in the most current literature.

A 2010 study of 86 male and female scientists in the U.K. found women in computing are portrayed in ways that overemphasize their appearance and sexuality. Men and women are not represented equitably, with women often portrayed as "exceptional," implying female participation in computing means being uniquely incomparable and to some degree abnormal and a transgression in the male-dominated culture of work. "Ordinary women" are perceived as less than capable, while ordinary men can and do fully participate in the computing disciplines. Additionally, when the media (re)pres-

ents women, it is done to "sex up" technology to increase the popularity of the field,^{13,16} with minimal focus on intellectual aptitude,²⁸ as in the images in the online appendix.

These gender-clipped images and media text references continue to be portrayed online by Web authors despite designers and developers having progressive, non-stereotypical views of the roles of women in tech. For example, Mendick and Moreau²⁷ found this happens when Web authors work within a constrained journalistic or scientific culture that lacks gender equity. Web authors (or designers) thus frame and articulate Web content within these confines of gender-imbalance, even though they do not endorse the gender stereotypes themselves.^{7,27} Regarding race and ethnicity, Payton³¹ explored the importance of co-creation of online content to reach typically overlooked and underrepresented groups, highlighting the need to amplify damped voices, as well as self-creation in content development. While the context of such content related to dissemination of health care, black women engaged more with an online platform designed by and for black women, and when a degree of co-ownership and agency inspired positive (uplifting) imagery, nonbiased messaging, and active user participation.³¹

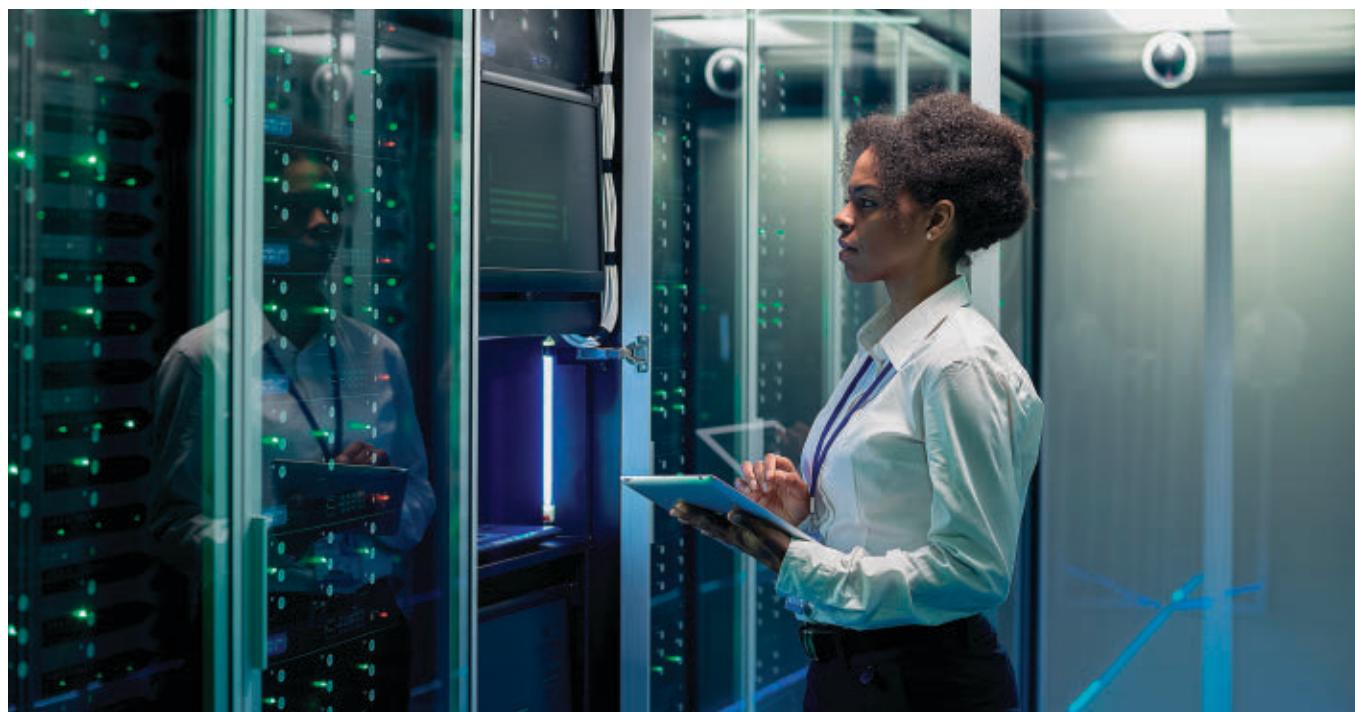


IMAGE BY FRAME STOCK FOOTAGES

Gender-clichéd images and gender symbolism. The online appendix includes examples of historical stereotypes that have been commercially available for some time, influencing the perceptions of men and women alike. Our research found masculinity and femininity are often used when the description of someone's behavior and skills is highlighted and interpreted through a gender-symbolic lens. Gender symbolism can be seen in the connection between, say, masculinity and tech;³⁷ it does not explicitly state that technology reflects the capacities of specific (real) men but more likely hegemonic masculinity,^{23,25} or traditions and attitudes (such as stereotypes) that legitimize the male-dominant position in society, thus reinforcing gender inequality. Hegemonic masculinity is reflected in ordinary human behavior. For example, some researchers considering gender and computing and IT^{2,25} have recently acknowledged that when publishing descriptions of men with computers, the frame of reference included engagement of superior knowledge and intellectual capacity, as well as technological mastery and power that may strengthen a man's masculine self-image, whereas women with computing skills were mostly associated with typewriting, calculators, and office work, as in the online appendix. As concluded by some researchers, including Bhatt et al.,⁷ gendered stereotypes not only influence perceptions of "associated type of work" but also who is able to even participate in computing.

Media portrayals of people, male and female alike, in science and math tend to create and support specific, gendered stereotypes about what a scientist is supposed to "look like." These stereotypes can significantly influence young women's decisions to pursue STEM subjects in school and later professionally. Here, we consider two prevalent stereotypes in media: women are inherently worse at science and math than men, and being a scientist involves character traits like being asocial that are inherently unappealing to young women.⁷

Gender and race in media stereotypes. With regard to race and par-

"Ordinary women" are perceived as less than capable, while ordinary men can and do fully engage and participate in the computing disciplines.

ticipation in computing, Nelson³⁰ provided an intensive examination of the scholarly literature, concluding researchers often ignore mainstream publications to gain a richer historical perspective of the field. For example, Nelson³⁰ referenced *Ebony Magazine* in a study of race and gender using "prosopography," or a description of a person's appearance, personality, and career. Considered a nontraditional source, *Ebony Magazine* can provide insights into how people view themselves and others. Nelson³⁰ documented at least 57 black professionals in computing from 1959 to 1996.³⁰ *Ebony Magazine* was founded in 1945 by black entrepreneur John H. Johnson and is a leading mainstream black publication covering how politics, science, business, the arts, and education affect the black community in the U.S. and globally.

There is also the issue of intersectionality of gender and race as generally experienced by women of color. For women of color in tech, the double-bind obstacles remain, even after the groundbreaking 1976 study by Shirley Malcom and Lindsey Malcom in the *Harvard Educational Review* as analyzed years later in "The Double Bind: The Price of Being a Minority Woman in Science."²⁶ The image and degree-attainment issues were and continue to be even more visible at the doctorate level, as reflected in the following quote from a related 2011 AAAS press release:²⁶ "They have made broad gains in social and behavior sciences, but lack badly in other fields. In computer science, for example, 2.1% of 2008 Ph.D.'s went to minority women scientists—just 14 women in all. While records show that no women received engineering Ph.D.'s in 1975, the number in 2008 had risen only to 91, or 2.9% of the total."

In the 2016 report *Barriers and Bias: The Status of Women in Leadership*, the American Association of University Women³ further underscored the effects of an intersectional context, indicating that black, Hispanic, and Asian women are often subjected to stereotypes and unconscious bias more widely than their white counterparts.³ These effects are similar and do not vary widely from corporate, nonprofit, govern-



A scene from a recent CS summer research group at Harvey Mudd College, Claremont, CA, where female participants outnumbered the males.

ment, and academic environments. It also discussed the effects of limited role models, even as it countered negative, often stereotypical, images in the mass media.³⁴ Given the lack of representation of women of color in computing, as outlined earlier and in the double-bind framework,²⁶ our understanding of media images is vital for men and women alike to better appreciate how the field has and has not progressed in terms of inclusive representation and participation while embracing leadership and entrepreneurial pathways.

Despite this call for inclusive participation and a longstanding invisibility prism associated with women of color, some researchers, including Blickenstaff,⁸ have described the prism as a lack of opportunity as seen through the lens of a leaky pipeline or alternatively gendered filter.⁸ When considering gender, race, and other factors affecting career selection and participation, we should also embrace efforts to broaden participation in media messaging with “intentionality.” Intersectional perspectives²⁴ are central to augmenting identity work frameworks of women in computing. The matter of ordinary males and exceptional females raises a new challenge: minority women have an even greater challenge, along with the one self-imposed and/or reinforced by computing culture—to be significantly better than their peers. This notion raises the question: How can minority women be better than exceptional?

Underrepresented professionals and turnover due to unfair treatment. Gender stereotyping can contribute to women’s and underrepresented minorities’ lack of participation in computing and IT, reflecting discriminatory treatment, as well as being detrimental to innovation, creativity, and personal motivation.

Besides being pushed out or voluntarily leaving the field, U.S. women leave computing and IT at a rate 45% higher than their male counterparts. For example, in 2014 and 2017, women in Brazil, China, India, and the U.K. reported they would quit within a year ranged from 20% to 30%.^{11,15} In a representative sample of U.S. adults who left an IT job between 2015 and 2017, a Kapor Center for Social Impact and Harris Poll study reported unfair treatment cost the U.S. tech sector \$16 billion annually, concluding: “Unfair treatment is the single largest driver of turnover affecting all groups, and most acutely affects underrepresented professionals.”²²

Bro culture, stereotypes, family issues, working conditions, organizational climate, and lack of sponsors and allies were, along with outright personal bias, were identified as barriers to navigating careers in computing and IT, according to the Center for Talent Innovation, 2014,¹¹ and other sources.^{7,24,32,39} Despite such social and structural issues facing women and underrepresented groups in computing and IT,³² some historically underrepresented groups are finding entrepreneurship to be a promising alternative

career path. *The State of Women-Owned Businesses American Express 2016 Report* stated there were 11.3 million women-owned businesses in the U.S. as of 2016, a 45% increase over 2007. These figures also reflected the growth of female entrepreneurship in Europe, as of 2016.⁴⁰

Conclusion

We observed popular media images and references to women in computing and IT focusing on the “likeness” of how much women exhibit and reflect masculinity and asocial features. These limited perceptions of women (of themselves and/or by others) have been found to influence women’s personal decisions about educational and professional life, including matriculation, research priorities, and career trajectories. Educational institutions and industry will continue to experience participation fractures within historically underrepresented gender and ethnic groups. Biased images and perceptions of who participates and ultimately succeeds in computing will continue to be constrained by an often-biased worldview. We analyzed facts and preconceptions in order to provide a more realistic understanding and awareness of the insidious effects of bias, focusing on how to positively reshape and enrich the identity narratives of women’s roles in computing and IT.

We collected and analyzed stereotypical roles of women in computing and IT as included in news reporting and imagery, identifying stereotypes and associated themes. Their frequent appearance in the media continues to reflect the stereotypes of society in general, and now also in computing and IT education and employment. Moreover, stereotyping is constantly evolving and adapting. Moreover, regarding gender equality in computing/IT education and career choices, exposure to role images continues to influence decisions women make about their educational and professional lives.

We interpreted our observation results through the lens of today’s dynamic IT and computing workforce that, for the sake of equity among all people, considers both genders when recruiting. In reality, though, gendered, race-targeted unfair treatment remains a barrier to gainful long-term employment, since underrepresented minorities and women often feel compelled to

leave their jobs due to unfavorable corporate/academic environments.

The nonrealistic profiles and non-positive images of women in the popular media, along with underrepresented minorities and sociocultural groups, continue to warrant thorough investigation and redirection. Rather than be distracted by stereotypical (negative or controversial) images, researchers and media leaders should focus on the story (re)telling process and untold narratives to capture lived experiences and establish an inclusive climate for those historically underrepresented in the field. As research cited in this article indicates, imagery influences both future and current participation in computing and IT in terms of retention and reducing the number of “tech leavers.” Other narratives of interest include underemployed and unemployed women and how the negative stereotypes seen in mass-media images might influence participation in such emerging career pathways as data science, cybersecurity/privacy, artificial intelligence, virtual reality, and machine learning. Computer science and related tech industries, as well as society in general, must promote equitable computing/IT roles and the associated images needed to represent inclusive media acculturation, role/work identity, and intersectionality.^{3,13,26} Along with audiovisual artifacts with proper work-life balance,⁸ inclusive women’s roles in a multicultural context should be on the research and development agendas of educators, STEM leaders, and education policymakers. Inclusive, realistic role images/models could help increase the number of underrepresented minorities in academia and in the tech work force while strengthening and inspiring entrepreneurial mind-sets and pathways. □

References

1. Abbate, J. *Recoding Gender: Women's Changing Participation in Computing*. MIT Press, Cambridge, MA, 2012.
2. Adam, A., Griffiths, M., Keogh, C., Moore, K., Richardson, H., and Tattersall, A. Being an ‘it’ in IT: Gendered identities in IT work. *European Journal of Information Systems* 15, 4, 2006, 368–378.
3. American Association of University Women. *Barriers and Bias: The Status of Women in Leadership*, 2016; <https://www.aauw.org/research/barriers-and-bias/>
4. Ashcraft, C., McLain, B., and Eger, E. *2016 Update Women in Tech: The Facts: See What's Changed and What Hasn't*. National Coalition of Women in Info Tech; https://www.ncwit.org/sites/default/files/resources/nctic_women-in-it_2016-full-report_final-web06012016.pdf
5. Austing, R., Barnes, B., Bonnette, D., Engel, G., and Stokes, G. Curriculum '78: Recommendations for the undergraduate program in computer science. *Commun. ACM* 22, 3 (Mar. 1979).
6. Berki, E. and Payton, F.C. Work-life balance and identity in a virtual world: Facts, tensions and intentions for women in IT. Chapter in *Lost and Found in Virtual Reality: Women and Information Technology*, H. Isomäki and A. Pohjola, Eds. University of Lapland Press, Rovaniemi, Finland, 2005, 275–296.
7. Bhatt, M., Blakely, J., Mohanty, N., and Payne, R. *How Media Shapes Perceptions of Science and Technology for Girls and Women*. Fem Inc., 2015; <https://learcenter.org/wp-content/uploads/2014/10/femSTEM.pdf>
8. Blückenstaff, J.C. Women and science careers: Leaky pipeline or gender filter? *Gender & Education* 17, 4 (Oct. 2005), 369–386.
9. British Computing Society. *Policy and Influence*. The U.K. Women in Tech Council. Women in IT Interviews; <https://www.bcs.org/upload/pdf/women-in-it.pdf>
10. British Computing Society. *The Women in IT Scorecard*. The Chartered Institute for IT, 2014; <https://www.bcs.org/upload/pdf/Women%20in%20IT%20scorecardv2.pdf>
11. Center for Talent Innovation. *Athena 2.0: Accelerating Female Talent in Science, Engineering and Technology*, Feb. 1, 2014; <http://www.talentinnovation.org/publication.cfm?publication=1420>
12. Cheryan, S., Plaut, V.C., Handron, C., and Hudson, C. The stereotypical computer scientist: Gendered media representations as a barrier to inclusion for women. *Sex Roles: A Journal of Research* 69, 1–2 (2013), 58–71; <https://psycnet.apa.org/record/2013-22691-001>
13. Chimba, M. and Kitzinger, J. Bimbo or boffin? Women in science: An analysis of media representations and how female scientists negotiate cultural contradictions. *Public Understanding of Science* 19, 5 (Sept. 1, 2010), 609–624.
14. Fiske, S.T., Cuddy, A.J., Glick, P., and Xu, J. A model of (often mixed) stereotype content: Competence and warmth respectively follow from perceived status and competition. *Journal of Personality and Social Psychology* 82, 6 (2002), 878–902; https://cos.gatech.edu/facultyres/Diversity_Studies/Fiske_StereotypeContent.pdf
15. Fox, K. *The 30 Most Influential Computer Scientists Alive Today*. Computer Science Hub, Dec. 2014; <http://www.computersciencedegreehub.com/30-most-influential-computer-scientists-alive-today/>
16. Fox, M.F. Women and men faculty in academic science and engineering: Social-organizational indicators and implications. *American Behavioral Scientist* 53, 7 (Feb. 9, 2010), 997–1012.
17. Gholipour, B. Is the stereotype that ‘women can’t be geniuses’ causing gender gaps? *Huffington Post* (Jan. 20, 2015); https://www.huffingtonpost.com/2015/01/20/women-geniuses_n_6508908.html
18. Gioia, D.A., Corley, K.G., and Hamilton, A.L. Seeking qualitative rigor in inductive research. *Organizational Research Methods* 16, 1 (Jan. 1, 2013), 15–31.
19. Hannon, K. Inspired or frustrated, women go to work for themselves. *New York Times* (Oct. 3, 2017); <https://www.nytimes.com/2017/10/03/business/women-entrepreneur-career.html>
20. Hofstede, G. Dimensionalizing cultures: The Hofstede Model in context. *Online Readings in Psychology and Culture* 2, 1 (Jan. 12, 2011), article 8; <https://doi.org/10.9707/2307-0919.1014>; <https://scholarworks.gvsu.edu/cgi/viewcontent.cgi?article=1014&context=orpc> and The 6-D model of national culture; <https://geerthofstede.com/culture/geert-hofstede-gert-jan-hofstede/6d-model-of-national-culture/>
21. Humphreys, S.M., Ed. Women and minorities in science: Strategies for increasing participation. In *Proceedings of the AAAS Selected Symposium 66*. Westview Press, Inc., Boulder, CO, 1982.
22. Kapor Center for Social Impact. *The 2017 Tech Leavers Study Online Report*; <https://www.kaporcenter.org/tech-leavers/>
23. Kereithi, P.J. *Hegemonic Masculinity in Media Contents*. UNESCO, 2014; http://www.unesco.org/fileadmin/MULTIMEDIA/HQ/CI/CI/pdf/publications/gamag_research_agenda_kereithi.pdf
24. Kvasny, L., Trauth, E., and Morgan, A. Power relations in IT education and work: The intersectionality of gender, race and class. *Journal of Information, Communication and Ethics in Society* 7, 2/3 (2009), 96–118; <https://doi.org/10.1108/14779960910955828>
25. Lie, M. Technology and masculinity: The case of the computer. *The European Journal of Women's Studies* 2, 3 (Aug. 1, 1995), 379–394; <https://journals.sagepub.com/doi/10.1177/135050689500200306>
26. Malcom, S. and Malcom, L. *Thirty-Five Years After 'The Double Bind,' Obstacles Remain for Minority Women in STEM*. As reported by the American Association for the Advancement of Science, Aug. 15, 2011; <http://www.aaas.org/news/thirty-five-years-after-double-bind-obstacles-remain-minority-women-stem>
27. Mendick, H. and Moreau, M. New media, old images: Constructing online representations of women and men in science. *Engineering and Technology, Gender and Education* 25, 3 (2013), 325–339; <10.1080/09540253.2012.740447>
28. Mundy, L. Why is Silicon Valley so awful to women? *The Atlantic* (Apr. 2017), 60–73; <https://www.theatlantic.com/magazine/archive/2017/04/why-is-silicon-valley-so-awful-to-women/517788/>
29. National Science Foundation. *Women, Minorities, and Persons with Disabilities in Science and Engineering: Special Report NSF 17-310*. National Center for Science and Engineering Statistics, Arlington, VA, 2017; <https://www.nsf.gov/statistics/2017/nsf17310/>
30. Nelsen, R.A. Race and computing: The problem of sources, the potential of prosopography, and the lesson of *Ebony Magazine*. *IEEE Annals of the History of Computing* 39, 1 (Jan.–Mar. 2017), 29–51.
31. Payton, F.C. Cultures of participation & design - @ myhealthimpact: For students, by students. *Information Systems Journal* 26, 4 (Aug. 14, 2015), 319–338.
32. Quora (contributor). Why women leave the tech industry at a 45% higher rate than men. *Forbes* (Feb. 28, 2017); <https://www.forbes.com/sites/quora/2017/02/28/why-women-leave-the-tech-industry-at-a-45-higher-rate-than-men/#f6e2db5744216>
33. Siakas, K.V., Berki, E., and Georgiadou, E. CODE for SQM: A model for cultural and organisational diversity evaluation. In *Proceedings of the European Software Process Improvement Conference* (Graz, Austria, Dec. 10–12). Verlag der Technischen Universität: Graz, Austria, 2003, IX.1–11.
34. Simon, S. and Hoyt, C.L. Exploring the effect of media images on women's leadership self-perceptions and aspirations. *Group Processes & Intergroup Relations* 16, 2 (Mar. 1, 2013), 232–245.
35. Sydell, L. *The Forgotten Female Programmers Who Created Modern Tech*. National Public Radio blog, Oct. 6, 2014; <https://www.npr.org/sections/alltechconsidered/2014/10/06/345799830/the-forgotten-female-programmers-who-created-modern-tech>
36. Thornham, H. and McFarlane, A. Cross-generational gender constructions. Women, teenagers and technology. *The Sociological Review* 59, 1 (Feb. 2011), 64–85.
37. Tiainen, T. and Berki, E. The reproduction process of gender bias: A case of ICT professors through recruitment in a gender-neutral country. *Journal of Studies in Higher Education* 44, 1 (2019), 170–184.
38. The U.K. Women in Tech Council. *Policy and Influence*. The Chartered Institute for IT, British Computing Society—Women in IT Interview, 2014; <http://www.bcs.org/category/17537>
39. Williams, J.C. Hacking tech's diversity problem. *Harvard Business Review* 92, 10 (Oct. 2014), 94–100.
40. Womenable. *The 2016 State of Women-Owned Businesses Report*; http://www.womenable.com/content/userfiles/2016_State_of_Women-Owned_Businesses_Executive_Report.pdf

For more, see the online appendix “Media Representations of Women in Computing Through Text and Images” (<https://dl.acm.org/citation.cfm?doid=3319422&picked=format>)

Fay Cobb Payton (fcpayton@ncsu.edu) is a professor of information technology and analytics and a university faculty scholar in the Poole College of Management at North Carolina State University, Raleigh, NC, USA.

Eleni Berki (eleni.e.berki@jyu.fi) is an adjunct professor of software quality and formal modeling in the Department of Computer Science and Information Systems at the University of Jyväskylä, Finland.

© 2019 ACM 0001-782/19/5 \$15.00



Watch the authors discuss this work in the exclusive Communications video. <https://cacm.acm.org/videos/stereotypical-gendered-media-images>

contributed articles

DOI:10.1145/3274276

Although smart contracts are Turing complete, it is a misconception that they can fulfill all routine contracts.

BY YONGGE WANG AND QUTAIBAH M. MALLUHI

The Limit of Blockchains: Infeasibility of a Smart Obama-Trump Contract

BLOCKCHAINS HAVE BECOME a buzzword, and many blockchain proponents believe a smart contract is a panacea for redefining the digital economy. But the community has a misconception that any kind of contract could be implemented as a blockchain smart contract. There is no doubt that Turing-complete scripting languages in blockchain techniques, such as Ethereum, can be used to draft many important smart contracts, but the digital economy is much more than Turing-complete smart contracts. Many protocols/contracts in our daily lives could not be implemented using Turing-complete smart contracts. As an example, we have formulated an Obama-Trump contract and show such a contract cannot be implemented using blockchain smart-contract techniques.

As the Internet increasingly becomes part of our daily lives, it will be convenient to have a digital payment system or design digital currency for society. It is generally easy to design an electronic cash system using public key infrastructure (PKI) systems. But PKI-based electronic cash is also easy to trace. Theoretically, banknotes could be traced using sequence numbers, though there is no convenient infrastructure to trace banknote sequence numbers back to users. Banknotes thus maintain sufficient anonymity.

An electronic cash system must avoid double spending and is preferred to be nontraceable and convenient for carrying out small transactions of, say, even a few cents. Such electronic cash systems could be designed using Chaum's blind signatures for untraceable payments.⁴ Assume the bank has an RSA public key (e, N) and a private key d . In order for Alice to withdraw \$10 from her bank account and convert it to a digital coin m of \$10, the system carries out the following protocol:

- ▶ Alice chooses a random number r and computes $m' = m \cdot r^e \pmod{N}$.
- ▶ The bank generates a signature $s' = (m')^d$ on m' .
- ▶ Alice calculates a signature s on m as $s = s' \cdot r^{-1} = (m \cdot r^e)^d \cdot r^{-1} = m^d$.
- ▶ Alice spends (m, s) as \$10, although the bank cannot link this coin m to Alice's account.

» key insights

- Smart contracts are self-executing Turing-complete programs stored permanently on the blockchain, triggered by blockchain transactions and able to read/write data from/to the blockchain database.
- Expectations for what smart contracts can do are inflated; Turing completeness does not imply they are a comprehensive tool for implementing contracts, as some contracts in our daily lives cannot be realized through smart contracts.
- The limit of smart contracts is theoretically proven by using impossibility results in secure multiparty computation to show the infeasibility of implementing an Obama-Trump contract as a smart contract.



There are various challenges to such a blindsignature-based electronic cash system. The first is what happens if Alice asks the bank to sign $m' = 100 \cdot r^e \pmod{N}$ instead of $m' = 10 \cdot r^e \pmod{N}$? It could be resolved by requiring that all coins have the same value or by using the following probabilistic approach:

- Alice generates 100 blind coins: $m'_i = m_i \cdot r_i^e \pmod{N}$ for $i = 1, \dots, 100$.
- The bank randomly selects $1 \leq j_1 < \dots < j_{99} \leq 100$.
- Alice reveals the values m_{j_i}, r_{j_i} to the bank for $i = 1, \dots, 99$.

- The bank issues a signature on the remaining m' only if the $m_{j_i} = 10$ and $m' = m_{j_i} \cdot r_{j_i}^e \pmod{N}$ for $i = 1, \dots, 99$.

The second challenge for Chaum's blind-signature-based electronic cash system is a seller must contact the bank to make sure the coin m has not been spent yet before accepting the coin m from Alice. This requires that the bank remains online at all times. Chaum et al.⁵ constructed an electronic cash that does not need the bank to be online. Let H_1, H_2 be hash functions and k be a fixed even integer. Assume Alice has an account u with a bank,

and the bank keeps a counter number v for Alice. In order for Alice to get a digital coin from the bank, the following steps are carried out:

- Alice chooses random a_i, c_i, d_i , and r_i for $1 \leq i \leq k$.
- Alice sends k blind candidates $B_i = r_i^e \cdot H_1(x_i, y_i)$ to the bank where
- $x_i = H_2(a_i, c_i)$
 - $y_i = H_2(a_i \oplus (u \parallel (v+i)), d_i)$
- The bank chooses a random subset $R \subset \{1, \dots, k\}$ of size $k/2$ and sends R to Alice.
- Alice reveals a_i, c_i, d_i , and r_i for all $i \in R$.

► Bank signs $S = \prod_{i \in R} B_i^d$, deducts the dollar from Alice's account, and increases v by k .

► Alice extracts coin $C = S \cdot (\prod_{i \in R} r_i)^{-1} = \prod_{i \in R} (H_1(x_i, y_i))^d$.

When Alice wants to make a payment to Bob, Alice sends C to Bob. Assume

$$\{i_1, \dots, i_{k/2}\} = \{1, 2, \dots, k\} \setminus R.$$

Bob sends random bits $z_{i_1}, \dots, z_{i_{k/2}}$ to Alice. For $j = 1, \dots, k/2$, Alice responds as follows:

1. If $z_{i_j} = 0$, then Alice sends a_{i_j}, c_{i_j} and y_{i_j} to Bob. In this case, Bob is able to compute $H_1(x_{i_j}, y_{i_j}) = H_1(H_2(a_{i_j}, c_{i_j}), y_{i_j})$.

2. If $z_{i_j} = 1$, then Alice sends $x_{i_j}, a_{i_j} \oplus (u \parallel (v + i_j))$, and d_{i_j} to Bob. In this case, Bob is able to compute

$$H_1(x_{i_j}, y_{i_j}) = H_1(x_{i_j}, H_2(a_{i_j} \oplus (u \parallel (v + i_j)), d_{i_j})).$$

After receiving these values, Bob is able to verify that C is a signature on the message $\prod_{j=1}^{k/2} H_1(x_{i_j}, y_{i_j})$.

In this transaction process, Alice's bank does not need to be online. In order for Bob to cash the coin C at Alice's bank, Bob sends the coin C together with Alice's response to Alice's bank. One may wonder: If Alice's bank is not online, how can we avoid double spending? If Alice spends the same coin both at Bob's shop and at Charlie's shop, then the challenge sequences $z_{i_1}, \dots, z_{i_{k/2}}$ from Bob and Charlie are different with high probability. Assume the challenge bit $z_{i_1} = 0$ for Bob and $z_{i_1} = 1$ for Charlie. Then Alice has revealed $a_{i_1}, c_{i_1}, y_{i_1}, x_{i_1}, a_{i_1} \oplus (u \parallel (v + i_1))$, and d_{i_1} . That is, Alice's account number u could be recovered from these revealed values. Or if Alice double spends, her identity will be revealed.

Many other non-PKI-based digital cash systems have been proposed in the literature. For example, Rivest and Shamir¹² proposed the PayWord and MicroMint payment schemes. In PawWord, Alice computes a sequence of binary strings $w_0, w_1, w_2, \dots, w_n$ such that $w_i = H(w_{i+1})$, where H is a secure cryptographic hash function. Alice then commits w_0 to the bank that cannot be spent. Assume each payment is one cent, then the i -th cent is spent as (i, w_i) . In MicroMint, there is a central broker to mint the coins. For example, in order for the broker to mint 2^{30} coins, it will use an array of 2^{30} . The broker will repeatedly hash randomly selected binary strings r and put the pair $(r, H(r))$ in the bin labeled $H(r)$. The mint process is finished when each of these bins contains four entries. Each bin is considered as one coin. That is, each coin is a tuple (x_1, x_2, x_3, x_4) such that $H(x_1) = H(x_2) = H(x_3) = H(x_4)$.

Bitcoin

The cryptographic currencies in the preceding section have never been adopted in practice. The situation has changed as the cryptographic currency Bitcoin was introduced in a paper by pseudonym "Satoshi Nakamoto."¹⁰ Since 2009, Bitcoin has been in operation and widely adopted as one of the major cryptographic currencies in the market. The cryptography behind Bitcoin is quite simple. The start coinbase by Satoshi Nakamoto is a binary string w_0 . In order to mine the first Bitcoin BTC, one needs to find a random number r_0 such that the first 32 bits of $w_1 = H(w_0, r_0)$ are 0...0 (that is, $w_1 < 2^{|w_0|-32}$). Anyone who finds this r_0 is rewarded with a few BTCs. The next person who finds another r_1 such that the first

two bits of $w_2 = H(w_1, r_1)$ are 0...0 will also be rewarded with a few BTCs. This process continues, and new blocks w_{i+1} keep adding to the existing block chain w_0, \dots, w_i . If the frequency of finding a BTC block is less than 10 minutes, the community initiates a voting process to increase the number of 0s in the required prefix of the hash outputs. The Bitcoin is a chain w_0, w_1, \dots, w_n , where w_n is the current Bitcoin head everyone works on. The Bitcoin network is a peer-to-peer (P2P) network with all participants working on the longest chain. There is no benefit for one to work on a shorter chain, as it is a waste of time and the transaction included in these chains will not be valid. The transactions of Bitcoins are included in the hash inputs so they can be verified later. Specifically, we have

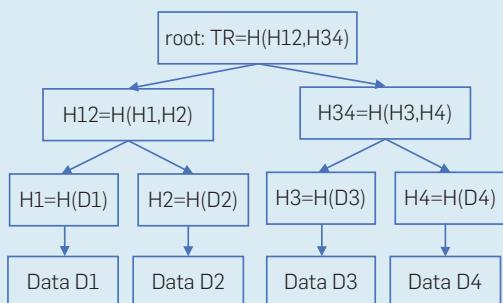
$$w_{i+1} = H(w_i, \text{TR}, r_i)$$

where TR is the Merkle hash of the transactions one wants to include and r_i is a random number one finds to make w_{i+1} have a certain number of 0s; the Merkle hash tree is outlined in the figure here.

In the Bitcoin system, a user is identified by a public key, and a transaction is in the format of "Alice pays x BTC to Bob." Alice achieves a transaction by signing the message "reference number, Bob's pub key, BTC amount x ," where the reference number refers to a block w_i in the current BTC chain w_0, w_1, \dots, w_n , where Alice received at least x BTCs in a transaction with the given reference number included in w_i . For example, the block w_i includes a transaction with this given reference number showing Alice received certain amount of BTCs. Bitcoin transactions are described using Forth-like scripts. The scripts enable smart contracts, such as "the transaction will be valid two days after all three persons have signed the contract." The Forth-like scripts are a stack-based script language and was used mainly in calculators. For example, in order to compute $25 \times 10 + 50$, one needs to initialize the stack as "[top] 25, 10, *, 50, + [bottom]."

Though it is argued that if the majority of users are honest, then the Bitcoin protocol should be reliable,¹⁰ Eyal

Merkle hash tree.



and Sirer⁷ showed this may not be true. In Eyal's and Sirer's attack, the adversary controls 1/3 computing power of the entire Bitcoin community and does not reveal the block it mined if it leads. The other 2/3 of users will waste their time on a chain that will be abandoned at some point when the adversary reveals its own leading chain. As users could choose arbitrary public keys for Bitcoins, it is claimed that user privacy is preserved in Bitcoins to some degree.¹⁰ There have been significant efforts to analyze the privacy issues in Bitcoin systems, and the conclusion is that a significant amount of private information could be recovered from Bitcoin chains. There have been many proposals concerning privacy-preserving solutions in Bitcoin networks. Androulaki et al.² tried to give a privacy definition in Bitcoin networks based on the traditional definition of privacy in computer networks. Following these definitions, Androulaki et al.² implemented a simulated Bitcoin network and observed a 40% user profile could be identified in the simulated environments. Ober et al.¹¹ analyzed some global properties of Bitcoin networks and their impact on user privacy. Möser⁹ analyzed three mixing services for Bitcoin networks: BTC Fog, BitLaundry, and Shared Wallet from Blockchain.info. Möser⁹ observed that among these three services, BTC Fog and Shared Wallet have good privacy protection, and tainted analysis could be used to trace Bitcoins in BitLaundry due to its lower volume per day. Moore and Christin⁸ analyzed 40 Bitcoin exchange centers, observing the smaller the volume, the shorter the lifetime of the exchange center. On the other hand, more recent work by Ahmed et al.¹ showed serious attacks against public cryptocurrency-mining pools, such as Minergate and Slush Pool. In them, an attacker needs only a small fraction (such as one millionth) of the resources of a victim-mining pool to render the victim-mining pool nonfunctional.

Ethereum

Though Forth-like scripts in Bitcoin are sufficient for designing various kinds of smart contracts, it has a limited capability. One underlying philosophy in Ethereum is to include a

If the contract language is Turing-complete, then the required validation systems are equivalent to the problem of deciding whether a universal Turing machine halts on a specific input.

Turing-complete programming language within the blockchain system so any kind of smart contract can be supported in the blockchain. Ethereum was designed as an Internet Service Platform with the goal that anybody can upload programs to the Ethereum World Computer, and anybody can request an uploaded program be executed. There are mainly two new functions in Ethereum compared with Bitcoin:

- Ethereum is a blockchain with a built-in Turing-complete programming language, allowing anyone to write smart contracts and decentralized applications where they can create their own arbitrary rules for ownership, transaction formats, and state transition functions.

- Bitcoin supports only "proof of work," whereas Ethereum supports both "proof of stake" and "proof of work," where "proof of stake" calculates the weight of a node as proportional to its currency holdings, not its computational resources.

The runtime environment for smart contracts in Ethereum is based on the Ethereum virtual machine (EVM). The EVM can run any operations that are created by the user using the Turing-complete Ethereum scripting language called Solidity. An Ethereum account is a 20-byte string with four fields: nonce, ether balance, contract code (optional), and storage (empty by default). There are two kinds of Ethereum accounts: Ethereum externally owned accounts (EOAs) and contract accounts. An EOA is linked to a private key, and a contract account can be "activated" only by an EOA. A contract account is governed by its internal smart-contract code programmed to be controlled by an EOA with a certain address. A smart-contract program within a contract account executes when a transaction is sent to that account. Senders of a transaction must pay for each step of the "program" they activate. This includes both computation and memory storage costs. Users can create new contracts by deploying code to the blockchain.

Infeasibility of a Smart Obama-Trump Contract

As blockchains use Turing-complete script languages to draft smart

contracts, many people might have the misconception that any kind of contract can be implemented in blockchains. Though most financial contracts can be implemented using Turing-complete script languages, there are challenges in implementing smart contracts with private inputs. In this section, we analyze the limit of smart contracts that can be implemented in blockchains. In particular, we show it is theoretically impossible to implement the so-called Obama-Trump contract.

In the legal system, there are four types of classifications of contracts with various bases: formation, nature of consideration, execution, and validity.

1. On the basis of formation, there are three types of contracts: express, implied, and quasi contracts. For an express contract, there is an expression or conversation. For an implied contract, there is no expression. For example, sitting in an airplane incurs an implied contract between the passenger and the airline. For a quasi contract, there are no contractual relations between the partners. This kind of contract is created by virtue of law.

2. On the basis of the nature of the consideration, there are two types of contracts: bilateral and unilateral. A bilateral contract requires considerations in both directions to be moved after the contract, whereas a unilateral contract requires considerations to be moved in only one direction after the contract. An example of a bilateral contract is “Alice delivers goods to Bob on January 1st and Bob pays Alice on January 15th.”

3. On the basis of execution, there are two types of contracts: executed and executory. In an executed contract, the performance is completed. In an executory contract, the contractual obligations are to be performed in the future.

4. On the basis of validity, there are five types of contracts: valid, void, voidable, illegal, and unenforceable. A contract that is enforceable in a court of law is called a valid contract, and a contract that is not enforceable in a court of law is called a void contract, as in, say, a contract between Alice and Bob where Bob is a minor who has no capacity to contract is a void contract. A voidable contract is deficient only in terms of free consent. For example, the contract between Alice and Bob

where Bob has forcibly made Alice involved in the contract is a voidable contract at the option of Alice. An illegal contract contains an unlawful object. An unenforceable contract has not properly fulfilled legal formalities.

With the classification of these contract types, it is important to design validation systems to check the validity of smart contracts. We would like to see the following validation systems:

- Check whether one transaction is an implied contract;
- Check whether one transaction follows a quasi contract; and
- Check whether a contract is valid, void, voidable, illegal, or unenforceable.

If the contract language is Turing-complete, then the required validation systems are equivalent to the problem of deciding whether a universal Turing machine halts on a specific input. It is thus infeasible to design efficient validation systems to carry out these tasks due to the nondecidability of the universal Turing machine halting problem. In the Ethereum EVM, gas is needed to evaluate a contract. If the gas runs out before the contract is validated, the contract will not be honored.

Furthermore, not all such contracts could be enforced in blockchain as smart contracts. In particular, when privacy does not have a reasonable price tag, it is generally difficult to formulate a smart contract with private inputs. As an example, we show that a bilateral contract is difficult to implement if the second consideration in the bilateral contract is not a digital cash (such as not an Ethereum ETH). In April 2011, Donald Trump made the comment¹³ in an interview with ABC's George Stephanopoulos: “Maybe I'm going to do the tax returns when Obama does his birth certificate ... I'd love to give my tax returns. I may tie my tax returns into Obama's birth certificate.” Based on this comment, we formulate the following Obama-Trump contract and show this kind of bilateral contract is impossible to implement as a blockchain smart contract.

Obama-Trump Contract: Donald Trump releases his tax return forms as soon as Barack Obama releases his birth certificate.

The infeasibility of implementing the Obama-Trump contract as a

blockchain smart contract can be mathematically proved using the infeasibility results in secure multiparty computations. We first review Cleve's result⁶ on the limits of coin flips when half of the participants are faulty.

Theorem 4.1 (Cleve⁶) *If at least half of the participants are faulty, then there is no protocol to allow an asynchronous network of participants to agree on random (unbiased) bits.*

Cleve⁶ defines a two-processor bit-selection scheme as a sequence of pairs of processors $\{(A^n, B^n)\}_{n=1}^{\infty}$ with the following properties. For each n , A^n and B^n each has access to a private supply of random bits and they can communicate with each other. If the system is executed, then A^n and B^n will output bits a and b , respectively, within a polynomial time. Assume the system consists of $r(n)$ rounds where each round consists of the following events: A^n performs some computations and sends a message to B^n , and then B^n performs some computations and sends a message to A^n . The two-processor bit-selection scheme is said to be “correct” if after the scheme is run, we have $a \neq b$ with a negligible probability. The two-processor bit-selection scheme is said to be “random” if the scheme is correct and if after the scheme is run, the value $|\text{Prob}[a=0] - \frac{1}{2}|$ is negligible. If one of the two processors is faulty, then it is unrealistic to expect the correctness of the scheme as the faulty processor could output a bit that is independent of the scheme that was run. However, it is desirable that the output of the honest processor is still random. Cleve⁶ defines a two-processor bit-selection scheme to be *secure* if the following holds: For each n , if one of A^n, B^n is faulty, then $|\text{Prob}[c=0] - \frac{1}{2}|$ is negligible where c is the output of the honest processor. Cleve⁶ shows that no secure two-processor bit-selection scheme exists when one of the processors is faulty. A similar construction as in Cleve's proof⁶ could be used to show the following theorem, as outlined here.

Theorem 4.2 *Obama-Trump smart contract cannot be enforced on blockchain.*

Theorem 4.2 shows it is infeasible to implement an Obama-Trump smart contract on blockchains. On the other hand, if a trapdoor function exists, then coin-flipping protocols (see Blum³ and Cleve⁶) can be used to design weakly secure Obama-Trump smart contracts over blockchain.

Smart Contract Scenarios

The results in the preceding section show that not all contracts can be implemented as a blockchain smart contract. However, blockchain smart contracts could do better than other technologies in many practical contract scenarios where the contract execution process takes a significant amount of time. For example, insurance-claim processing involves many manual operations and much human action. Blockchain smart contracts could help reduce these manual steps by including some measurable parameters, such as earthquake magnitude, within the contracts. When an insured event occurs, the event information is converted to smart contract input parameters, and the claim process is triggered immediately.

Smart contracts can also be used in many other scenarios where a lot of paperwork and coordination are required. For example, in trade finance, the process of letter-of-credit issuance requires numerous physical documents. As another example, in the rental-property application process, the applicant needs to submit numerous documents, including income certificates, rental credit reports, eviction history, and other related documents to the landlord. Note the user may need to submit identical documents to both the trade-finance vendor and the landlord at different times if the user is involved in both processes. It is thus preferred for a user to keep all these documents in a central blockchain account and submit only appropriate reference numbers to the documents for each application. The system should be designed in such a way that the user needs only to disclose minimal mandatory information to each vendor for a specific application. For example, for a user to apply for a rental property, the system should disclose only user income, rental credit

reports, and eviction reports to the landlord. The system should not disclose user eviction reports to the trade finance organization.

As information stored in the blockchain is publicly accessible, it is necessary to encrypt user documents in the user account. We may assume each document in a user profile has been certified by a related agency that is also a user account in the blockchain. As an example, the user Alice's master profile may look like this:

Alice Profile: $\text{DOC}_1, \text{DOC}_2, \dots$

where each document DOC_i is in the following format:

$$\text{DOC}_i = \text{S}.\text{Enc}_{\text{K}}(F, \text{Sign}_{\text{Agency}.pk}(F)), \\ \text{P}.\text{Enc}_{\text{Alice}.pk}(\text{K}), \text{Agency}.pk$$

where the document F is certified by the agency with a digital signature $\text{Sign}_{\text{Agency}.pk}(F)$ using the agency public key $\text{Agency}.pk$. The certified document $(F, \text{Sign}_{\text{Agency}.pk}(F))$ is then encrypted using a symmetric encryption scheme $\text{S}.\text{Enc}_{\text{K}}(\cdot)$ with a key K . The symmetric key K is encrypted using a public encryption scheme $\text{P}.\text{Enc}_{\text{Alice}.pk}(\cdot)$ with Alice's public key $\text{Alice}.pk$. In order for Alice to disclose the certified document $(F, \text{Sign}_{\text{Agency}.pk}(F))$ to the landlord, Alice needs to provide the document reference number DOC_i and the symmetric key K to the landlord.

Other Sophisticated Smart Contracts

A blockchain smart contract is generally written using a blockchain scripting language, such as Solidity. The algorithms within the smart contract are thus available for public review. In some applications, such as the insurance industry, the vendor may not want the public to learn the claim-processing algorithms used in the smart contract. Software obfuscation techniques may be used by smart contracts to hide these algorithms. Indeed, using reusable garble circuit techniques or fully homomorphic encryption (FHE) techniques are preferred for writing smart contracts in these scenarios. However, there are challenges in employing garbled circuits or FHE techniques in these scenarios, as it is difficult to convert plaintext

inputs into garbled inputs for garbled circuits or into encrypted inputs for FHE schemes.

PKI is the core component of the secure Internet infrastructure. Note, a PKI system based on blockchain smart-contract systems may be established to replace the current certificate authority (CA)-based PKI systems for Internet infrastructure. It depends on the corresponding cost and security characteristics for one to consider whether to use the current CA-based PKI system or blockchain-based PKI system for Internet infrastructure.

Acknowledgment

The work reported in this article is supported by Qatar Foundation Grant NPRP X-063-1-014. □

References

1. Ahmed, M., Wei, J., Wang, Y., and Al-Shaer, E. A poisoning attack against cryptocurrency mining pools. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 2018, 140–154.
2. Androulaki, E., Karame, G.O., Roeschlin, M., Scherer, T., and Capkun, S. Evaluating user privacy in Bitcoin. In *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, 2013, 34–51.
3. Blum, M. Coin flipping by telephone a protocol for solving impossible problems. *ACM SIGACT News*, 15, 1 (1983), 23–27.
4. Chaum, D. Blind signatures for untraceable payments. In *Proceedings of Crypto*. Springer, 1983, 199–203.
5. Chaum, D., Fiat, A., Naor, M. Untraceable electronic cash. In *Proceedings of Crypto*. Springer-Verlag, New York, 1990, 319–327.
6. Cleve, R. Limits on the security of coin flips when half the processors are faulty. In *Proceedings of 18th ACM Symposium on Theory of Computing*. ACM, 1986, 364–369.
7. Eyal, I., Sirer, E.G. Majority is not enough: Bitcoin mining is vulnerable. In *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, 2014, 436–454.
8. Moore, T. and Christin, N. Beware the middleman: Empirical analysis of Bitcoin-exchange risk. In *Proceedings of the International Conference on Financial Cryptography and Data Security*. Springer, 2013, 25–33.
9. Moser, M., Bohme, R., and Breuker, D. An inquiry into money-laundering tools in the Bitcoin ecosystem. In *eCrime Researchers Summit*. IEEE, 2013, 1–14.
10. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system, 2008; <https://bitcoin.org/bitcoin.pdf>
11. Ober, M., Katzenbeisser, S., and Hamacher, K. Structure and anonymity of the Bitcoin transaction graph. *Future Internet* 5, 2 (2013), 237–250.
12. Rivest, R.L. and Shamir, A. PayWord and MicroMint: Two simple micropayment schemes. In *Proceedings of the International Workshop on Security Protocols*. Springer, 1996, 69–87.
13. Trump, D. I will release my tax returns when Obama releases his birth certificate. 2011; <http://www.businessinsider.com/donald-trump-tax-returns-obama-birth-certificate-2011-4>

Yongge Wang (Yongge.wang@gmail.com) is in the Department of Software and Information Systems, University of North Carolina, Charlotte, NC, USA.

Qutaibah M. Malluhi (qmalluhi@qu.edu.qa) is in the Department of Computer Science and Engineering, Qatar University, Qatar.

review articles

DOI:10.1145/3319408

Tracing some of the latest advancements in algorithmic randomness.

BY ROD DOWNEY AND DENIS R. HIRSCHFELDT

Algorithmic Randomness

CLASSICAL PROBABILITY THEORY gives all sequences of fair coin tosses of the same length the same probability. On the other hand, when considering sequences such as

and

1011010111010111100001010100010111...,

none but the most contrarian among us would deny that the second (obtained by the first author by tossing a coin) is more random than the first. Indeed, we might well want to say that the second sequence is entirely random, whereas the first one is entirely nonrandom. But what are we to make in this context of, say, the sequence obtained by taking our first sequence, tossing a coin for each bit, and if the coin comes up heads, replacing that bit by the corresponding one in the second sequence? There are deep and fundamental questions involved in trying to understand why some sequences should count as “random,” or “partially random,” and others as

“predictable,” and how we can transform our intuitions about these concepts into meaningful mathematical notions.

One goal of the theory of algorithmic randomness is to give meaning to the notion of a random *individual* (infinite) sequence. Questions immediately arise: How should we define randomness? How can we measure whether one sequence is more random than another? How are computational power and randomness related? Is a theory of randomness for individual sequences at all useful? How does such a theory relate to classical probability theory?

The modern development of the theory of algorithmic randomness goes back to the 1960s (with even earlier roots, as we will discuss), but there has been a particular surge of development in the last couple of decades. In this article, we hope to give some of the flavor of this work, though we will be able to mention only a few samples of what is by now a vast area of research. Our book,¹¹ for example, is over 800 pages long and still manages to cover only a fraction of the area ... Another book covering some of the work we discuss here is Nies.³⁵ Li and Vitányi²⁷ is broader in scope and focuses more on Kolmogorov complexity of finite strings.

For simplicity, we assume sequences are binary unless we say otherwise. We use the terms *sequence* for an infinite sequence and *string* for a finite one. We write $|\sigma|$ for the length of a string σ ,

» key insights

- Computational theory can be used to give precise meaning to the notion of randomness for individual objects such as finite strings, infinite sequences, and real numbers.
 - This theory also allows us to calibrate partial randomness. Notions of partial randomness can then be used to assign dimensions to individual points, as opposed to sets of points, leading to useful tools for establishing new results in areas such as the theory of fractal dimensions.
 - It also gives us insight into how much and what kinds of randomness are needed for certain results in computer science and mathematics.



write $X(n)$ for the n th bit of the sequence X (beginning with the 0th bit $X(0)$), and write $X|n$ for the string consisting of the first n bits of X . We identify a real number x in the interval $[0, 1]$ with the binary sequence X such that $x = 0.X$. There are reals that have two binary expansions, but they are all rational and will not be relevant.

Historical Roots

Borel. In the beginning of the 20th century, Émile Borel was interested in sequences that satisfy the (*strong*) *law of large numbers*, which says that if we repeat an experiment with a numerical result many times, the average value of the result should be close to its expected value. If we toss a fair coin many times, for example, we expect the frequency of heads to be about $\frac{1}{2}$. Let X be a sequence representing infinitely many such tosses. After s many coin tosses, we can see how we are doing so far by looking at how many heads we have seen in the first s tosses compared to s , that is, the ratio

$$\frac{|\{X(k)=1|k < s\}|}{s},$$

where we think of a 1 as representing heads. If this is indeed a fair coin, this ratio should get closer and closer to $\frac{1}{2}$ as s increases. Moreover for the *strong law*, any length k subsequence, such as 1011 (of length 4), should appear with frequency approaching $\frac{1}{2^k}$.

More generally, we say that an n -ary sequence sequence X is (*Borel*) *normal* if it has the same property relative to an “ n -sided coin,” in other words, if for any length m sequence $\sigma = a_1 a_2 \dots a_m$ of digits between 0 and $n - 1$,

$$\lim_{s \rightarrow \infty} \frac{\text{number of times } \sigma \text{ appears as a substring of } X|s}{s} = \frac{1}{n^m}.$$

Borel defined a real number to be *normal to base n* if its base n representation is normal, and *absolutely normal* if it is normal to every base. Borel observed that almost every real number is absolutely normal. Mathematically, this fact can be expressed by saying the collection of absolutely normal numbers has Lebesgue measure 1, which corresponds to saying that if we threw a dart at the real line, with probability 1, it would hit an absolutely normal number. We

would thus expect a random sequence to be normal, and indeed (recalling that we identify the sequence X with the real number $0.X$) we would expect a random sequence to be absolutely normal.

Von Mises and Ville. The late 1920s and early 1930s saw the development, particularly by Andrey Kolmogorov, of an adequate foundation for probability theory, using measure theory and based on the idea of the expected behavior of events in a probability space. This theory does not give any meaning to the idea of randomness of an individual object, such as a particular sequence of coin tosses. Tossing a fair coin n times takes place in a “space of possibilities” (in this case, the collection of all binary strings of length n), and we assign any sequence of length n the probability 2^{-n} of occurring. For example, as we are taught in school, any particular sequence of three coin tosses occurs with probability $2^{-3} = \frac{1}{8}$.

In the infinite case, we might look at the event that a sequence has a certain string, say 101, as an initial segment. The probability that we begin a sequence of coin tosses with heads, tails, heads is $2^{-3} = \frac{1}{8}$. The mathematical way to express this fact is that the (*uniform*) *measure* (also known as the *Lebesgue measure*) of the set of sequences beginning with 101 is 2^{-3} , or, more generally, the measure of the set of sequences beginning with any particular string of length n is 2^{-n} . Probability theory is of course a vast and complex field, but for our purposes, this simple example suffices.

It is less commonly known that Kolmogorov’s work came after earlier attempts to give meaning to the notion of randomness for individual objects such as infinite sequences. This idea is completely contrary to the approach in which all sequences are equally likely, but is quite reasonable when thinking about the difference between sequences like the two that open this article. The question is how to differentiate between a sequence like 01101 11001011101110001001101010111100..., the base 2 version of *Champernowne’s sequence*, obtained by listing the binary representations of the natural numbers in order and clearly nonrandom, and one arising from a random source. There are tests we can apply to a sequence to try to verify its apparent

randomness. For example, a random sequence should be normal in the sense of the previous section. However, that is not a sufficient condition, as the aforementioned sequence is known to be normal to base 2, but is highly predictable.

In 1919, Richard von Mises^a attempted to give a definition of randomness for a sequence X based upon a generalization of the law of large numbers. His idea was to require normality not only of X itself, but also of (certain) infinite subsequences of X . The point here is that the base 2 Champernowne sequence is normal, but if we computably select every $[g(n) = 1 + \sum_{j \leq n} j(2^j - 2^{j-1})]$ -th bit of this sequence, the resulting subsequence 1111 ... is no longer normal. It is not reasonable that selecting such bits of a random sequence should result in all 1s, so our sequence fails this randomness test.

Von Mises generalized this idea as follows. Let $f: \mathbb{N} \rightarrow \mathbb{N}$ be an increasing function. We think of f as a *selection function* for determining a subsequence of a given sequence. That is, $f(i)$ is the i th place selected in forming this subsequence. In the law of large numbers itself, where we consider the entire sequence, $f(i) = i$. In the nonrandomness argument in the previous paragraph, $f(i) = g(i)$. Von Mises proposed replacing the ratio $\frac{|\{X(k)=1|k < s\}|}{s}$ coming from the law of large numbers by

$$\frac{|\{X(f(k))=1|k < s\}|}{s},$$

the ratio of the number of *selected places* at which X has value 1 to the total number of selected places. For base 2 and each choice of f , the requirement that this ratio approach $\frac{1}{2}$ as s goes to infinity constitutes a randomness test.

So when should X be regarded as random? We could perhaps try to say that X is random if and only if it passes this test for all possible selection functions, reflecting the idea that in a sequence of coin tosses, there should be no way to select a subsequence ahead of time that will have a greater proportion of heads than tails. There is a big problem with this idea, though. No sequence X can be random for *all* selection functions. As

^a See Downey and Hirschfeldt¹¹ for references to this and other sources mentioned in this section.

any nontrivial X has infinitely many 0s, there is an f that chooses the positions of the 0's of X in increasing order. But surely this counterexample is unfair to the spirit of von Mises' idea: we are trying to capture the notion that we should not be able to *predict* the values of bits of X , and this f is chosen *after* defining X . It is always easy to predict the answer if you know it in advance! The question then is what kinds of selection functions should be allowed, to capture the notion of prediction. A reasonable intuition is that prediction is somehow a computational process, and hence from a modern perspective we might want to restrict ourselves to *computable* selection functions, a suggestion later made by Alonzo Church.

Von Mises' work predated the definition of computable function, however, so he had no canonical choice of "acceptable selection rules" and left his definition mathematically vague. But Abraham Wald showed that for any countably infinite collection of selection functions, there is a sequence that is random in the sense of passing all tests corresponding to the functions in this collection.

However, von Mises' program was dealt a major blow in 1939 by Jean Ville, who showed that for any countable collection of selection functions, there is a sequence X that passes all of the resulting tests, but such that for each n , there are always more 0s than 1s in $X \upharpoonright n$. If we were told that there would always be more tails than heads in a sequence of coin flips, we would not believe the coin to be a standard one, and could use this information to make some money betting on its flips. Thus, Ville's sequence is random in the sense of von Mises, but certainly not random in the intuitive sense.

Ville suggested adding versions of another law (the law of the iterated logarithm) to the list of tests that a sequence would need to pass to be considered random. Perhaps von Mises' tests together with these additional tests would capture the notion of algorithmic randomness. But this all begins to look very ad hoc, and immediately raises the natural question of whether there is a Ville-like counterexample for this new set of laws. (As it turns out, there is, as discussed, for example, in Downey and Hirschfeldt.¹¹)

We are abandoning the idea of absolute randomness in some metaphysical sense in favor of a notion of algorithmic randomness, where we use tools from computability theory to define and quantify randomness.

Notice that in these discussions, we are abandoning the idea of *absolute randomness* in some metaphysical sense in favor of a notion of *algorithmic randomness*, where we use tools from computability theory to define and quantify randomness. Abandoning absolute randomness leads to the idea of "levels of randomness" that can be defined by calibrating the computability theoretic complexity of the tests we require our random sequences to pass. But, of course, following Ville's work it was not clear that even one reasonably robust level of algorithmic randomness could be defined.

Martin-Löf. This is how matters stood until 1966 and the work of Per Martin-Löf, who effectivized the notion of null set from classical measure theory and gave a satisfying definition of algorithmic randomness based on this effectivization. The basic idea is that a random sequence should not have any "rare" property, that is, that if we find a way to distinguish and describe a small collection of sequences, then no random sequence should be in our collection. The notion of null set allows us to make precise what we mean by "small."

Randomness tests like those suggested by von Mises are computable ways to narrow down which sequences can be considered random. For example, consider sequences like 0101... that have 0's in all even places. We do not want such "bad" sequences to be considered random. To test whether a sequence has this form, we take a "level-by-level" approach: Given a sequence X , we ask whether $X(0) = 0$. If so, then X fails the first level of our test. (That is, it fails to demonstrate so far that it is not a bad sequence.) Note that half of all sequences X have $X(0) = 0$, which can be formalized by saying that the set of sequences X with $X(0) = 0$ has measure $\frac{1}{2}$.

Next, we ask whether $X(0) = 0$ and $X(2) = 0$. If so, then X fails the second level of our test. The proportion of all sequences X that fail this second level is $\frac{1}{4}$. We continue in this fashion, testing more and more even places. A sequence X is one of our bad sequences if and only if it fails *all* levels of our test. The fact that the set T_n of sequences that fail the n th level of our test has measure 2^{-n} implies the set of bad sequences, which is the intersection of all the T_n 's, has measure 0, that is, that it is what we call a *null set*.

Martin-Löf's approach was to generalize this process by considering all possible level-by-level procedures for testing randomness. We can think of such a procedure as being generated by a machine M . At each level n , this machine determines a set T_n of sequences that are deemed to have failed the test so far. It does so by enumerating strings $\sigma_0^n, \sigma_1^n, \dots$, where we then let T_n be the collection of all sequences that begin with some σ_i^n . Of course, M needs to be fair and not, say, consider all sequences to be nonrandom, so we insist that, like in the above example, T_n contains at most a proportion 2^{-n} of all sequences (which we can formalize by saying that the measure of T_n is at most 2^{-n}). Now a sequence X fails M 's test if it is contained in every T_n , and otherwise it passes this test.

We say that a sequence is *Martin-Löf random* if and only if it passes *all* such tests.^b It can be shown that almost all sequences are Martin-Löf random (that is, that the collection of Martin-Löf random sequences has measure 1). Furthermore, Martin-Löf's notion of tests includes the ones proposed by von Mises (in the specific realization suggested by Church), the ones proposed by Ville, and indeed all “algorithmically performable” randomness tests. Thus, the objection to the idea of adding more and more specific tests as we uncover more and more Ville-like sequences is neatly circumvented.

As it turns out, Martin-Löf randomness is also quite well-behaved mathematically, and has provided a robust basis for the theory of algorithmic randomness. As Jack Lutz put it in a lecture at the 7th Conference on Computability, Complexity, and Randomness, held in Cambridge in 2012 (in connection with work of Turing that we will discuss later), “Placing computability constraints on a nonconstructive theory like Lebesgue measure seems *a priori* to weaken the theory, but it may strengthen the theory for some purposes. This vision is crucial

The measure of a set of sequences is the mathematical version of the probability that a sequence is in this set.

for present-day investigations of individual random sequences, dimensions of individual sequences, measure and category in complexity classes, etc.”

In summary, Martin-Löf reformulated all the laws that we would expect a random sequence to obey at an abstract level, based upon the idea of *effectivizing* measure theory, that is, making a computable version of measure theory. The measure of a set of sequences is the mathematical version of the probability that a sequence is in this set. Martin-Löf randomness says we regard X as random if and only if it passes each computably generated test that determines a set of computable measure 0 (as the intersection of the levels of the test). Such an X has every property that we can algorithmically describe as a set of probability 1.

Solomonoff, Kolmogorov, Levin, Chaitin, and Schnorr. There are other approaches to a definition of algorithmic randomness. For (finite) strings, a suitable definition was formulated by Kolmogorov, who argued that if a string has identifiable regularities, then we should be able to compress it, and that a compressible string should not be thought of as random. Here, we think of a machine M as a descriptive process. If an input τ is processed by M to yield an output σ , then τ is a description of σ , that is, a program that M can use to print σ . A random σ should have no short descriptions.

As an illustration, consider the sequence $\sigma = 01010101\dots$ (1000 times). A short description τ of σ is “print 01 1000 times.” This brief program produces an output of length 2000. We are exploiting the regularities of this particular σ to compress it into a short description. Kolmogorov's intuition was that for a random sequence, there should be no regularities, so that the only way to describe σ is to essentially use σ itself. More precisely, a string of length n should be random relative to some descriptive process if and only if its shortest description has length n . Like white noise, a random string should be *incompressible*.

To give a physical analogue of this idea, suppose we have a maze shaped like a binary tree of height 6, with boxes at the end. There are 2^6 possible routes to get to the boxes. One of the boxes has money in it, and someone is

^b Formally, a *Martin-Löf test* is a collection S_0, S_1, \dots of uniformly computably enumerable sets of strings such that, if we let T_n be set of all sequences that begin with some element of S_n , then T_n has measure at most 2^{-n} . (The notion of computable enumerability is also known as recursive enumerability.) A sequence X passes this test if $X \notin \cap_n T_n$. A sequence is *Martin-Löf random* if it passes all Martin-Löf tests.

to tell us which. If the box is the left-most one, all they have to say is “always turn left.” If the box is to be found by say, left-right-left, this path is again easy to describe. If the place of the prize is determined randomly, though, the person would likely need to tell us the whole sequence of turns. (Li and Vitányi²⁷ report on an experiment of this kind about ant communication.) This compressibility approach gives rise to what is now called *Kolmogorov complexity*. For a Turing machine M , the Kolmogorov complexity $C_M(\sigma)$ of σ relative to M is the length of the shortest τ such that $M(\tau) = \sigma$. We can then take a universal Turing machine U , which can emulate any other given machine M with at most a constant increase in the size of programs, and define the (plain) Kolmogorov complexity of σ to be $C(\sigma) = C_U(\sigma)$.

A natural guess is that a sequence X is random if and only if for all n , the first n bits of X are incompressible in the sense outlined earlier. As it turns out, however, plain Kolmogorov complexity is not quite the correct notion for infinite sequences. (The reason is that in the above account, M can use more than just the bits of τ to generate σ . It can also use the length of τ , which provides an additional $\log|\tau|$ many bits of information. Using this idea, Martin-Löf showed that for any X , and any constant c , the plain Kolmogorov complexity of $X|n$ must always dip below $n-c$ for some lengths n .)

There are several ways to modify the definition of Kolmogorov complexity to avoid this issue, the best-known being to use prefix-free codes^c and the resulting notion of *prefix-free Kolmogorov complexity*, denoted by K in place of C . Its roots are in the work of Leonid Levin, Gregory Chaitin, and Claus-Peter Schnorr, and in a certain sense even earlier in that of Ray Solomonoff. As shown by Schnorr, it is indeed the case that X is Martin-Löf random if and only if the prefix-free Kolmogorov complexity of the first n bits of X is at least n (up to an additive constant), that is, $K(X|n) \geq n - O(1)$.

^c That is, descriptions that are like telephone numbers in that if τ and ρ are input descriptions to M and both give outputs, then τ is not a prefix of ρ .

(There are many other flavors of Kolmogorov complexity, such as time-and space-bounded ones, but C and K have been the most studied. They have a complex relationship. It is easy to show that $K(\sigma) \leq C(\sigma) + 2\log|\sigma| + O(1)$. Robert Solovay proved the remarkable fact that $K(\sigma) = C(\sigma) + C(C(\sigma)) + O(C^{(3)}(\sigma))$ and this result is tight in that we cannot extend it to $C^{(4)}(\sigma)$. There is a huge amount of research on the Kolmogorov complexity of finite strings and its applications. See, for instance, Li and Vitányi.²⁷)

Returning to the story of the definition of algorithmic randomness, there is another approach, developed by Schnorr, that is close in spirit to von Mises’ ideas. A *martingale*^d is a function d from strings to nonnegative reals satisfying a fairness condition:

$$d(\sigma) = \frac{d(\sigma 0) + d(\sigma 1)}{2}$$

We think of d as representing a betting strategy. We begin with some capital $d(\lambda)$, where λ is the empty string, and bet on the values of the successive bits of a sequence X so that the amount of money we have after n many bets is $d(X|n)$. We are allowed to hedge our bets by betting some amount of our capital on 0 and the rest on 1. The displayed equation ensures that this betting is fair, that is, that the average of the returns of our bets on 0 and on 1 equals our current total. A martingale d succeeds on a sequence X if and only if the associated betting strategy allows us to make arbitrarily much money when betting on the bits of X , that is, $\limsup_{n \rightarrow \infty} d(X|n) = \infty$. Schnorr showed that there is a notion of effective martingale such that X is Martin-Löf random if and only if no such martingale succeeds on X . This idea is close to von Mises’ prediction-based approach, except that martingales allow us to spread our bets between the outcomes 0 and 1, so von Mises’ intuition has a realization that works after all!

In summary, there are three basic approaches to defining random sequences: the *statistician’s approach*, that a random sequence should have no computably rare properties; the *coder’s approach*, that a random

sequence should have no regularities that allow for compression; and the *gambler’s approach*, that a random sequence should be unpredictable. In each of these cases, a natural effective realization leads to the same notion, Martin-Löf randomness.

Some Things We Have Learned

Calibrating randomness. As natural and robust as Martin-Löf’s definition of algorithmic randomness is, it is only one among many reasonable notions that together allow us to calibrate levels of randomness. One way to obtain new notions of randomness is to change the collection of tests a sequence is required to pass to be considered random. For instance, we can consider Martin-Löf tests with computable measures (that is, where the measure of each level T_n is exactly 2^{-n} , for instance), yielding a notion called *Schnorr randomness*. Another possibility is to use martingales with different levels of effectiveness, such as ones that are computable functions from strings to nonnegative rationals, which yields a notion called *computable randomness*. Computable randomness can also be miniaturized to complexity classes, giving rise to notions such as polynomial-time randomness.

It can be shown that Martin-Löf randomness implies computable randomness, which in turn implies Schnorr randomness, and that neither of these implications can be reversed. But the separations between these notions are quite subtle, and indeed the notions coincide for sequences that are in a sense “close to computable.” (More precisely, they coincide outside what are known as the *high* sequences, which resemble the Halting Problem in a certain technical sense; see Nies et al.³⁶) Indeed, there is a notion of *nonmonotonic randomness*—which is like computable randomness but allows for strategies that can bet on the values of the bits of a sequence in any computable order—for which equivalence to Martin-Löf randomness is still a long-standing open question.

We can also modify our tests to yield notions stronger than Martin-Löf randomness. For instance, relaxing the condition that the n th level T_n of a Martin-Löf test must have measure at most 2^{-n} , and requiring only that the measures of the T_n ’s go to 0 as n goes to

^d This notion is related to but distinct from that of martingale in probability theory.

infinity, yields the notion of *weak 2-randomness*, which is intermediate between Martin-Löf randomness and the notion of 2-randomness discussed below.

In some ways, weak 2-randomness is better-behaved than Martin-Löf randomness. To give an example, let us begin by considering the fact that, although almost every sequence is Martin-Löf random, it is not that easy to come up with an explicit example. That is at it should be, of course. Easily describable sequences (such as computable ones, for example) should not be random. Nevertheless, such examples do exist, the best-known being Chaitin's Ω , defined as the probability that a universal prefix-free Turing machine U halts on a given input,^e or, more formally, as:

$$\Omega = \sum_{U(\sigma) \text{ halts}} 2^{-|\sigma|}.$$

Although Ω is Martin-Löf random, it is also computationally powerful, being Turing equivalent to the Halting Problem.^f

The existence of computationally powerful Martin-Löf random sequences is surprising, as intuitively we should expect random sequences not to contain much “useful information.” (The distinction here is between the kind of information that makes a sequence hard to describe and the kind that can actually be used. If we choose 1,000 characters at random, we expect the resulting text to be difficult to describe, but would be shocked to find that it contains instructions for making a soufflé.) However, not only is it possible for a Martin-Löf random sequence to compute the Halting Problem, but by the Kučera-Gács Theorem, *every* sequence can be computed from some Martin-Löf random sequence. (See, for example, Downey and Hirschfeldt¹¹ for a proof.) By increasing the level of randomness, we can make these “pathological” examples disappear. If X is

weakly 2-random, then it cannot compute the Halting Problem, or indeed, any noncomputable sequence that is computed by the Halting Problem, and hence in particular any noncomputable, computably enumerable set.

We do not have to go all the way to weak 2-randomness, though. There are results, beginning with work of Stephan,³⁸ that indicate that the Martin-Löf random sequences split into two classes: powerful ones that can compute the Halting Problem, and weaker ones that exhibit much more of the behavior we expect of random sequences, and in particular are computationally much weaker than the sequences in the first class. Franklin and Ng¹⁷ showed that the level of randomness of these “true Martin-Löf randoms” can be captured by a natural test-based notion known as *difference randomness*. The study of notions of algorithmic randomness like this one, which are intermediate between Martin-Löf randomness and weak 2-randomness, has had an important role in recent research in the area, and helped us refine our understanding of the relationship between levels of randomness and computational power.

Another way to calibrate randomness is to relativize notions such as Martin-Löf randomness. For instance, we can consider Martin-Löf tests that are produced not by a standard Turing machine, but by a Turing machine with access to an oracle Z . If Z is the Halting Problem, for example, we obtain a notion called *2-randomness*. More generally, we have a notion of *n-randomness*, where we relativize Martin-Löf tests to the $(n-1)$ st iterate of the Halting Problem.^g Here, 1-randomness is just Martin-Löf randomness.

Much is known about this hierarchy, including some surprising facts. For example: As noted by Miller and Yu,³³ it follows from a fundamental result about Martin-Löf randomness known as van Lambalgen's Theorem (see Downey and Hirschfeldt¹¹) that if X is Martin-Löf random and is computed by an n -random sequence, then X is itself n -random. We have mentioned that we

can never have $C(X|n) \geq n - O(1)$ for *all* n , but it is possible to have a sequence X such that $C(X|n) \geq n - O(1)$ for *infinitely many* n . Remarkably, Miller³⁰ and Nies et al.³⁶ showed that this condition is equivalent to 2-randomness. Miller³¹ also proved a similar result saying that 2-randomness also coincides with having infinitely often maximal initial segment prefix-free Kolmogorov complexity. Indeed, it is possible to give characterizations of n -randomness for all n using unrelativized Kolmogorov complexity (see Bienvenu et al.⁸). These facts are examples of the often subtle interplay that recent research in this area has uncovered between levels of randomness, initial-segment complexity, and relative computability.

Calibrating nonrandomness. For sequences that are not Martin-Löf random, there are ways to calibrate how close they come to randomness. A natural way to do this is to consider the (prefix-free) Kolmogorov complexity of their initial segments. For example, a sequence X is *complex* if there is a computable, nondecreasing, unbounded function f such that $K(X|n) \geq f(n)$ for all n . Complex sequences can be characterized in terms of their ability to compute certain sequences that resemble the Halting Problem to some extent (see Downey and Hirschfeldt¹¹), which is another example of the interplay between randomness and computability.

At the other extreme from random sequences are those with strong “anti-randomness” properties. Identifying a natural number with its binary expansion, we always have $C(\sigma) \geq C|\sigma| - O(1)$, because if we know a string, then we know its length. Thus, the lowest the plain Kolmogorov complexity of the initial segments of a sequence X can be is $C(X|n) \leq C(n) + O(1)$. In the 1970s, Chaitin showed that this condition holds if and only if X is computable, and asked whether the same holds for prefix-free Kolmogorov complexity. In an unpublished manuscript written in 1975, Solovay showed the surprising fact that there are noncomputable sequences X such that $K(X|n) \leq K(n) + O(1)$ for all n , though Chaitin had already shown that there are only countably many of them, and indeed that they are all computable from the Halting Problem. Such sequences are

^e The value of Ω depends on the choice of U , but its basic properties do not; see Downey and Hirschfeldt.¹¹

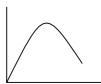
^f When we say that X can be computed from Y , we mean there is a Turing machine M with an oracle tape so that if the oracle tape contains Y , then M computes X . Two objects are Turing equivalent if each can be computed from the other. Turing's Halting Problem is the classic example of a complete computably enumerable set; that is, it is itself computably enumerable, and it can compute every computably enumerable set.

^g The k^{th} iterate of the Halting Problem is just the Halting Problem for Turing machines with the $(k-1)^{\text{st}}$ iterate of the Halting Problem as an oracle.

said to be *K-trivial*, and have played a major role in the theory of algorithmic randomness. For those who know some computability theory, we mention that Nies³⁴ showed that the *K*-trivial sequences form an ideal in the Turing degrees, and that they can be seen as giving a priority-free solution to Post's Problem (see Downey et al.¹²). Nies³⁴ showed that these sequences are computability-theoretically weak, and gave several characterizations of *K*-triviality in terms of randomness-theoretic weakness. For example, when we relativize the notion of Martin-Löf randomness to a non-computable *X*, we expect the notion to change, as the noncomputability of *X* should yield some derandomization power. Nies showed that the *K*-trivial sequences are exactly those for which this intuition fails.

Many other characterizations of *K*-triviality have since been given. For example, results of Hirschfeldt et al.²¹ and of Bienvenu et al.⁶ show a computably enumerable set is *K*-trivial if and only if it is computed by a difference random sequence (that is, one of the “true Martin-Löf randoms” that does not compute the Halting Problem). Recent work on *K*-triviality has also revealed subclasses of the *K*-trivials that can further help us understand the fine structure of the interaction between randomness and computability.

Considering the properties of sequences with differing levels of randomness leads to the following heuristic graph, where the horizontal axis represents randomness level and the vertical axis represents maximum computational power. (One can also think that the horizontal axis represents information content, whereas the vertical axis represents maximum useful information content.)



Among the sequences that are neither random nor highly nonrandom are ones that can be thought of as being “partially random.” For example, if *Z* is Martin-Löf random and we replace every other bit of *Z* by a 0, we obtain a new sequence *Y* such that $K(Y|n)$ is approximately $\frac{n}{2}$. It makes sense to think of such a sequence as being “ $\frac{1}{2}$ -random.” More generally, we can think

A remarkable feature of the theory of effective dimension is there is a tight correspondence between the classical Hausdorff dimension of a set and the effective Hausdorff dimension of its points.

of the limit behavior of the ratio $\frac{K(X|n)}{n}$ as a measure of the partial randomness of a sequence *X*. This ratio does not necessarily have a limit, but we can look at

$$\liminf_{n \rightarrow \infty} \frac{K(X|n)}{n} \text{ and } \limsup_{n \rightarrow \infty} \frac{K(X|n)}{n}$$

which both give us values between 0 and 1.

These values are also central to the theory of effective dimension. In 1919, Felix Hausdorff introduced a notion of dimension that measures the “local size” of a set in a metric space, for example, a subset of the plane. Points have dimension 0, lines have dimension 1, and the whole plane has dimension 2, but there are also objects of fractional dimension, such as well-known fractals like the Koch curve (which has Hausdorff dimension $\log_3(4)$). Starting with the work of Jack Lutz in the early 2000s, the theory of dimension has been effectivized, initially in terms of effective martingales as in Schnorr's approach to algorithmic randomness. This process has also been carried out for other notions of dimension, most notably that of packing dimension. An important fact here is that the effective Hausdorff dimension and effective packing dimension of a sequence *X* turn out to be exactly the liminf and limsup, respectively, in the equation explained above. Thus, these dimensions can be seen as measures of partial randomness. (See, for example, see Downey and Hirschfeldt¹¹ for details.)

The theory of effective dimension has also been extended to points on the plane and higher dimensional Euclidean spaces. A remarkable feature of this theory is that there is a tight correspondence between the classical Hausdorff dimension of a set and the effective Hausdorff dimension of its points. For a fairly wide class of sets $S \subseteq \mathbb{R}^n$, Hitchcock²² showed that the Hausdorff dimension of *S* is the supremum of the effective Hausdorff dimensions of its individual elements, and Lutz and Lutz²⁸ have now given versions of this result for arbitrary sets (and for both Hausdorff and packing dimension) using relativizations of effective dimension. It is surprising that the notion of dimension, which seems so clearly to be a global property of a set, based on its “overall shape,” can be

completely understood by focusing on the individual elements of the set and understanding them from a computability-theoretic perspective. This correspondence is also quite useful, and can be used to obtain new proofs and results in areas such as fractal geometry, as in Lutz and Lutz²⁸ and Lutz and Stull,²⁹ for instance.

Randomness *amplification* can be investigated in many settings. A basic question is whether (a greater degree of) randomness can always be extracted from a partially random source. In our setting, effective dimension can be used to measure the degree of randomness, and extraction can be interpreted as relative computation. One way to think of this question is that it is easy to decrease the effective dimension of a sequence in a computable way, say by changing a large proportion of its bits to 0's, but it is less clear in general whether there is a way to reverse this process.

As it turns out, the answer depends on the notion of dimension. Fortnow et al.¹⁵ showed that if X has nonzero effective packing dimension and $\varepsilon > 0$, then there is a Y that is computable from X such that the effective packing dimension of Y is at least $1 - \varepsilon$. (In fact, they showed that Y can be chosen to be Turing-equivalent to X via polynomial-time reductions, making the randomness amplification process quite efficient.) On the other hand, Miller³² showed there is a sequence X of effective Hausdorff dimension $\frac{1}{2}$ such that if Y is computable from X , then the effective Hausdorff dimension of Y is at most $\frac{1}{2}$. (The specific value $\frac{1}{2}$ does not matter.) Greenberg and Miller¹⁹ showed that there is a sequence of effective Hausdorff dimension 1 that does not compute any Martin-Löf random sequence. Thus, we see there are some strong senses in which randomness amplification is not possible. However, Zimand⁴⁰ showed that, remarkably, if we have two sequences of nonzero effective Hausdorff dimension that are sufficiently independent in a certain technical sense, then they together compute a sequence of effective Hausdorff dimension 1.

This is still an area of significant research interest. For example, we can ask about a randomness amplification

It is surprising the notion of dimension, which means so clearly to be a global property of a set based on its “overall shape,” can be understood by focusing on the individual elements of the set and understanding them from a computability-theoretic perspective.

process where, instead of using computable reductions, we simply seek to increase the randomness of a sequence by changing a relatively small proportion of its bits. Greenberg et al.²⁰ recently gave precise bounds on the proportion of bits of a sequence of effective Hausdorff dimension s that need to be changed to increase the Hausdorff dimension to a given $t > s$, in terms of the binary entropy function from information theory. They also showed that if X has effective Hausdorff dimension 1, then X can be transformed into a Martin-Löf random sequence by changing it only on the bits in a set $S \subset \mathbb{N}$ of density 0 (which means that $\lim_{n \rightarrow \infty} \frac{|S \cap [n]|}{n} = 0$).

Turing and absolute normality. We return to Borel’s notion of normality. This is a very weak form of randomness; polynomial-time randomness is more than enough to ensure absolute normality, and indeed, it is known that a sequence is normal if and only if it satisfies a notion of randomness defined using certain finite-state machines much weaker than arbitrary Turing machines. Borel asked whether there are explicit examples of absolutely normal numbers. It is conjectured that e , π , and all irrational algebraic numbers, such as $\sqrt{2}$, are absolutely normal, but *none* of these have been proven to be normal to *any* base. In an unpublished manuscript, Alan Turing attacked the question of an explicit construction of an absolutely normal number by interpreting “explicit” to mean *computable*. His manuscript, entitled *A Note on Normal Numbers* and presumably written in 1938, gives the best kind of answer to date to Borel’s question: an algorithm that produces an absolutely normal number.

An interesting aspect of Turing’s construction is that he more or less anticipated Martin-Löf’s work by looking at a collection of computable tests sensitive enough to make a number normal in all bases, yet insensitive enough to allow a computable sequence to pass all such tests. We have seen that the strong law of large numbers implies fixed blocks of digits should occur with the appropriate frequencies in a random sequence. Translating between bases results in correlations between blocks of digits in one base and

blocks of digits in the other, which is why this extension allowed Turing to construct absolutely normal numbers. Turing made enough of classical measure theory computable to generate absolute normality, yet had the tests refined enough that computable sequence could still be “random.”

Turing’s construction remained largely unknown, because his manuscript was published only in his 1997 Collected Works.³⁹ The editorial notes in that volume say the proof given by Turing is inadequate and speculate the theorem could be false. Becher et al.⁴ reconstructed and completed Turing’s manuscript, preserving his ideas as accurately as possible while correcting minor errors. More recently, there has been a highly productive line of research connecting algorithmic randomness, computability theory, normal numbers, and approximability notions such as that of Liouville numbers; see, for instance, the papers listed at <http://www-2.dc.uba.ar/profesores/becher/publications.html>. Some of this work has yielded results in the classical theory of normal numbers, as in Becher et al.³

Some further applications. There have been several other applications of ideas related to algorithmic randomness in areas such as logic, complexity theory, analysis, and ergodic theory. Chaitin used Kolmogorov complexity to give a proof of a version of Gödel’s First Incompleteness Theorem, by showing that for any sufficiently strong, computably axiomatizable, consistent theory T , there is a number c such that T cannot prove that $C(\sigma) > c$ for any given string σ .^h More recently, Kritchman and Raz²⁵ used his methods to give a proof of the Second Incompleteness Theorem as well. (Their paper also includes an account of Chaitin’s proof.) We can also ask about the effect of adding axioms asserting the incompressibility of certain strings in a probabilistic way. Bienvenu et al.⁹ have shown that this kind of procedure does not help us to prove new interesting theorems, but that the situation changes if we take into account the size of the proofs: randomly chosen axioms can

help to make proofs much shorter under a reasonable complexity-theoretic assumption like $P \neq \text{PSPACE}$.

Although not central to this article, we mention there are *many* applications of Kolmogorov complexity of finite strings, for example, ones that go under the collective title of the *incompressibility method*. The idea is that algorithmically random strings should exhibit typical behavior on computable processes. For example, this method can be used to give average running times for sorting, by showing that if the outcome is not what we would expect, we can compress a random input (which is now a single algorithmically random string). Chapter 6 of Li and Vitányi²⁷ is devoted to this technique, applying it to areas as diverse as combinatorics, formal languages, compact routing, and circuit complexity, among others. Another example is provided by the insight that the Kolmogorov complexity $C(x|y)$ of a string x given y as an oracle is an absolute measure of how complex x is in y ’s opinion. Historically, researchers comparing two sequences x, y of, for example, DNA, or two phylogenetic trees, or two languages have defined many distance metrics, such as “maximum parsimony” in the DNA example. But it is natural to use a measure like $\max\{C(x, y), C(y, x)\}$, if the sequences have the same length, or some normalized version if they do not. Then we know absolutely what information the strings have in common, and do not have to hand-tool a notion of distance for the application. Although C is incomputable, Vitányi and others have used computable *approximations* (such as Lempel-Ziv compression) to C to investigate general tools for understanding common information. (See, for example, Bennett et al.⁵) Another application is learning theory and *logical depth*, a notion introduced by Bennett to capture the idea that something is hard to describe in limited time. For applications to deep learning, see, for example, <https://www.hectorzenil.net/publications.html>.

Randomness is used in many algorithms to accelerate computations, as in the use of randomness for primality testing by Solovay and

Strassen,³⁷ and there are problems like *polynomial identity testing*—which asks whether a polynomial in many variables is identically zero—for which there are efficient algorithms if we have a randomness source, but no known fast deterministic algorithms. It is thought that a wide class of randomized algorithms can be derandomized to yield deterministic polynomial-time algorithms, following the work of Impagliazzo and Wigderson,²³ who showed that if certain problems are as hard as we think they are, then we can provide enough randomness efficiently to derandomize problems in the complexity class BPP. Bienvenu and Downey⁷ have shown that randomness can always be used to accelerate *some* computations. They showed that if X is Schnorr random, then there is a computable language L such that X can compute L (in exponential time) via a computation Φ^X (that is, a Turing machine Φ with oracle X) so that for any Turing machine M that computes L , the computation Φ^X is faster than M by more than a polynomial factor. (That is, Φ^X computes L in time f , and there are no Turing machine M and polynomial p such that M computes L in time $p \circ f$.)

Another connection with complexity theory comes from looking at the computational power of the set of random strings. There are a few reasonable ways to define what we mean by this set; one of them is to consider the strings that are incompressible in the sense of plain Kolmogorov complexity, that is $R = \{\sigma | C(\sigma) \geq |\sigma|\}$. It turns out to be particularly interesting to consider what sets can be reduced to this one via polynomial-time reductions. For instance, Allender et al.¹ showed that the complexity class PSPACE is contained in the collection of sets that are polynomial-time reducible to R , and other connections with complexity theory have been explored in this paper and others such as Allender et al.²

A particularly promising current line of research is the use of notions of algorithmic randomness to give precise, “quantitative” versions of results about almost everywhere behavior in areas such as analysis and ergodic theory, an idea that goes back to the work of

^h This fact also follows by interpreting an earlier result of Barzdins; see Example 2.7.1 in Li and Vitányi.²⁷

Demuth in the 1970s.ⁱ For example, it is a result of basic analysis that every non-decreasing function $[0, 1] \rightarrow \mathbb{R}$ is differentiable at almost every $x \in [0, 1]$ (that is, the set of x at which it is differentiable has measure 1). Brattka et al.¹⁰ showed that the reals $x \in [0, 1]$ such that every nondecreasing computable function (in the sense of computable analysis) is differentiable at x are exactly the computably random ones. Thus, computable randomness is exactly the level of randomness needed for this particular almost everywhere behavior to manifest itself. For other similar conditions, the relevant level of randomness can vary. For instance, for functions of bounded variation in place of nondecreasing ones, the corresponding level of randomness is exactly Martin-Löf randomness, as shown in Brattka et al.¹⁰ as a recasting of a result by Demuth. A source for overviews of some recent work at the intersection of algorithmic randomness with analysis and ergodic theory is the collection of slides at <https://www.birs.ca/cmo-workshops/2016/16w5072/files/>. Similar applications occur in physics, for instance, in studying Brownian motion (for example, by Fouché¹⁶) and the amount of randomness needed for quantum mechanics (for example, by Gács¹⁸).

Another interesting application is to the study of tilings (of the plane, say). Let $X[m, n]$ be the bits of the sequence X from positions m to n . One might think that for a Martin-Löf random X , we should have $K(X[m, n]) \geq n - m - O(1)$, or that at least $K(X[m, n])$ should not dip too far below $n - m$. This is not true, though, as random sequences must have long simple substrings, such as long runs of 0's. (If we know that X has infinitely many runs of 6 consecutive 0's, but only finitely many of 7 consecutive 0's, then we can make money betting on the values of the bits of X by betting that the next value is 1 each time we see six consecutive 0's.) However, for any $\varepsilon > 0$, there are ε -shift complex sequences X for which:

$$K(X[m, n]) \geq (1 - \varepsilon)(n - m) - O(1)$$

ⁱ Demuth came from the constructivist tradition, but independently discovered notions of randomness like Martin-Löf randomness by working on questions such as the ones discussed in this paragraph. See Kučera et al.²⁶ for an account.

for all m and n . These sets can be coded to yield tilings with various interesting properties, such as certain kinds of pattern-avoidance. See, for instance, Durand et al.^{13, 14}

Finally, randomness is thought of as “typicality” for many objects. Thus, if we wish to understand complex networks, we can try to model them using some kind of random graph. Khoussainov²⁴ has recently given meaning to the idea of (infinite) algorithmically random regular trees and other structures. Work is under way to adapt this idea to finite graphs and use it for practical applications.

Acknowledgments

Downey wishes to thank the Marsden Fund of New Zealand. Hirschfeldt is partially supported by NSF Grant DMS-1600543. C

References

- Allender, E., Buhrman, H., Koucký, M., van Melkebeek, D., Ronneburger, D. Power from random strings. *SIAM J. Comput.* 35, 6 (2006), 1467–1493.
- Allender, E., Friedman, L., Gasarch, W. Limits on the computational power of random strings. *Inf. Comput.* 222 (2013), 80–92.
- Becher, V., Bugeaud, Y., Slaman, T.A. On simply normal numbers to different bases. *Math. Ann.* 364, 1–2 (2016), 125–150.
- Becher, V., Figueira, S., Picchi, R. Turing’s unpublished algorithm for normal numbers. *Theor. Comput. Sci.* 377, 1–3 (2007), 126–138.
- Bennett, C.H., Gács, P., Li, M., Vitányi, P.M.B., Zurek, W.H. Information distance. *IEEE Trans. Inf. Theory* 44, 4 (1998), 1407–1423. <https://doi.org/10.1109/18.681318>
- Biennvenu, L., Day, A.R., Greenberg, N., Kučera, A., Miller, J.S., Nies, A., Turetsky, D. Computing K-trivial sets by incomplete random sets. *B. Symb. Log.* 20, 1 (2014), 80–90.
- Biennvenu, L., Downey, R. On low for speed oracles. In *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018) (Leibniz International Proceedings in Informatics (LIPIcs))* (2018), R. Niedermeier and B. Vallée, eds. Volume 96, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Germany, 15:1–15:13.
- Biennvenu, L., Muchnik, A.N., Shen, A., Vereshchagin, N. Limit complexities revisited. In *25th International Symposium on Theoretical Aspects of Computer Science (Leibniz International Proceedings in Informatics (LIPIcs))* (2008), S. Albers and P. Weil, eds. Volume 1, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Wadern, 73–84 (electronic).
- Biennvenu, L., Romashchenko, A., Shen, A., Taveaux, A., Vermeeren, S. The axiomatic power of Kolmogorov complexity. *Ann Pure Appl Logic* 165, 9 (2014), 1380–1402.
- Brattka, V., Miller, J.S., Nies, A. Randomness and differentiability. *Trans. Amer. Math. Soc.* 368, 1 (2016), 581–605.
- Downey, R.G., Hirschfeldt, D.R. *Algorithmic Randomness and Complexity*. Springer, New York, 2010.
- Downey, R.G., Hirschfeldt, D.R., Nies, A., Stephan, F. Trivial reals. In *Proceedings of the 7th and 8th Asian Logic Conferences* (2003), R.G. Downey, D. Ding, S.P. Tung, Y.H. Qiu, M. Yasugi, eds. Singapore University Press and World Scientific, Singapore, 103–131.
- Durand, B., Levin, L.A., Shen, A. Complex tilings. *J. Symbolic Logic* 73, 2 (2008), 593–613.
- Durand, B., Romashchenko, A., Shen, A. Fixed-point tile sets and their applications. *J. Comput. System Sci.* 78, 3 (2012), 731–764.
- Fortnow, L., Hitchcock, J.M., Pavan, A., Vinodchandran, V., Wang, F. Extracting Kolmogorov complexity with applications to dimension zero-one laws. In *Automata, Languages and Programming. 33rd International Colloquium, ICALP (2006), Venice, Italy, July 10–14, 2006. Proceedings, Part I (Lecture Notes in Computer Science)*, M. Bugliesi, B. Preneel, V. Sassone, I. Wegener, eds. Volume 4051, Springer, Berlin, 335–345.
- Fouché, W. The descriptive complexity of Brownian motion. *Adv. Math.* 155 (2000), 317–343.
- Franklin, J.N.Y., Ng, K.M. Difference randomness. *Proc. Amer. Math. Soc.* 139, 1 (2011), 345–360.
- Gács, P. Quantum algorithmic entropy. *J. Phys. A. Math. Gen.* 34 (2001), 1–22.
- Greenberg, N., Miller, J.S. Diagonally non-recursive functions and effective Hausdorff dimension. *B. Lond. Math. Soc.* 43, 4 (2011), 636–654.
- Greenberg, N., Miller, J.S., Shen, A., Westrick, L.B. Dimension 1 sequences are close to randoms. *Theor. Comput. Sci.* 705 (2018), 99–112.
- Hirschfeldt, D.R., Nies, A., Stephan, F. Using random sets as oracles. *J. Lond. Math. Soc.* 75 (2007), 610–622.
- Hitchcock, J.M. Correspondence principles for effective dimensions. *Theor. Comput. Syst.* 38 (2005), 559–571.
- Impagliazzo, R., Wigderson, A. $P = BPP$ if E requires exponential circuits: derandomizing the XOR lemma. In *STOC’97* (1999) (*El Paso, TX*). ACM, New York, 220–229.
- Khoussainov, B. A quest for algorithmically random infinite structures. In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)* (2014). ACM, New York, Article No. 56, 9.
- Kritchman, S., Raz, R. The surprise examination paradox and the second incompleteness theorem. *Notices Am. Math. Soc.* 57, 11 (2010), 1454–1458.
- Kučera, A., Nies, A., Porter, C.P. Demuth’s path to randomness. *B. Symb. Log.* 21, 3 (2015), 270–305.
- Li, M., Vitányi, P. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer-Verlag, Berlin, 1993.
- Lutz, J.H., Lutz, N. Algorithmic information, plane Kakeya sets, and conditional dimension. In *34th Symposium on Theoretical Aspects of Computer Science (Leibniz International Proceedings in Informatics (LIPIcs))* (2017). Volume 66, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Wadern, Article No. 53, 13.
- Lutz, N., Stull, D.M. Bounding the dimension of points on a line. In *Theory and Applications of Models of Computation (Lecture Notes in Computer Science)*. Volume 10185, Springer, Cham, 2017, 425–439.
- Miller, J.S. Kolmogorov random reals are 2-random. *J. Symbolic Logic* 69 (2004), 907–913.
- Miller, J.S. The K-degrees, low for K-degrees, and weakly low for K sets. *Notre Dame J. Form.* 50 (2010), 381–391.
- Miller, J.S. Extracting information is hard: A Turing degree of non-integral effective Hausdorff dimension. *Adv. Math.* 226, 1 (2011), 373–384.
- Miller, J.S., Yu, L. On initial segment complexity and degrees of randomness. *Trans. Amer. Math. Soc.* 360 (2008), 3193–3210.
- Nies, A. Lowness properties and randomness. *Adv. Math.* 197 (2005), 274–305.
- Nies, A. *Computability and Randomness*. Oxford Logic Guides, Volume 51, Oxford University Press, Oxford, 2009.
- Nies, A., Stephan, F., Terwijn, S.A. Randomness, relativization, and Turing degrees. *J. Symbolic Logic* 70 (2005), 515–535.
- Solovay, R., Strassen, V. A fast Monte-Carlo test for primality. *SIAM J. Comput.* 6, 1 (1977), 84–85.
- Stephan, F. Martin-Löf random sets and PA-complete sets. In *Logic Colloquium’02 (Lecture Notes in Logic)*, Z. Chatzidakis, P. Koepke, W. Pohlers, eds. Volume 27, Association for Symbolic Logic and A K Peters, Ltd., La Jolla, CA and Wellesley, MA, 2006, 342–348.
- Turing, A.M. *Pure Mathematics*. J.L. Britton, ed. North-Holland Publishing Co., Amsterdam, 1992
- Zimand, M. Two sources are better than one for increasing the Kolmogorov complexity of infinite sequences. *Theor. Comput. Syst.* 46 (2010), 707–722.

Rod Downey (rod.downey@vuw.ac.nz) is a professor in the School of Mathematics and Statistics, Victoria University Wellington, New Zealand.

Denis R. Hirschfeldt (drh@math.uchicago.edu) is a professor in the Department of Mathematics at the University of Chicago, IL, USA.

research highlights

P. 82

Technical Perspective Compressing Matrices for Large-Scale Machine Learning

By Zachary G. Ives

P. 83

Compressed Linear Algebra for Declarative Large-Scale Machine Learning

By Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, and Berthold Reinwald

Technical Perspective

Compressing Matrices for Large-Scale Machine Learning

By Zachary G. Ives

DEMAND FOR MORE powerful big data analytics solutions has spurred the development of novel programming models, abstractions, and platforms for next-generation systems. For these problems, a complete solution would address data wrangling and processing, and it would support analytics over data of any modality or scale. It would support a wide array of machine learning algorithms, but also provide primitives for building new ones. It would be customizable, scale to vast volumes of data, and map to modern multicore, GPU, coprocessor, and compute cluster hardware. In pursuit of these goals, novel techniques and solutions are being developed by machine learning researchers,^{4,6,7} in the database and distributed systems research communities,^{2,5,8} and by major players in industry.^{1,3} These platforms provide higher-level abstractions for machine learning over data, and they perform optimizations for modern hardware.

Elgohary et al.'s work on "Scaling Machine Learning via Compressed Linear Algebra," which first appeared in the *Proceedings of the VLDB Endowment*,² seeks to address many of these challenges by applying database ideas (cost estimation, query optimization, cost-based data placement and layout). It was conducted within IBM and Apache's SystemML declarative machine learning project. The paper shows just how effective such database techniques can be in a machine learning setting. The authors observe that the core data objects in machine learning (feature matrices, weight vectors) tend to have regular structure and repeated values. Machine learning tasks over such data are composed from lower-level linear

algebra operations. Such operations generally involve repeated floating-point computations, which are bandwidth-limited as the CPU traverses large matrices in RAM.

The authors developed a compressed representation for matrices, as well as compressed linear algebra operations that work directly over the compressed matrix data. Together, these reduce the bandwidth required to perform the computations, thus speeding performance dramatically. The paper cleverly leverages ideas from relational database systems: column-oriented compression, sampling-based cost estimation, and trading between compression speed and compression rate.

This paper makes several notable contributions. First, the authors

The authors developed a compressed representation for matrices, as well as compressed linear algebra operations that work directly over the compressed matrix data.

identify a set of linear algebra primitives shared by multiple distributed machine learning platforms and algorithms. Second, they develop compression techniques not only for single columns in a matrix, but also "column grouping" techniques that capitalize on correlations between columns. They show that offset lists and run-length encoding offer a good set of trade-offs between efficiency and performance. Third, the paper develops cache-conscious algorithms for matrix multiplication and other operations. Finally, the paper shows how to estimate the sizes of compressed matrices and to choose an effective compression strategy. Together, these techniques illustrate how database systems concepts can be adapted to great benefit in the machine learning space. 

References

1. Abadi, M. et al. Tensorflow: A system for large-scale machine learning. *OSDI*, 16 (2016), 265–283.
2. Ewen, S., Tzoumas, K., Kaufmann, M. and Markl, V. Spinning fast iterative data flows. In *Proceedings of VLDB Endow.* 5, 11 (2012), 1268–1279.
3. Ghosh, A. et al. SystemML: Declarative machine learning on MapReduce. *ICDE*. IEEE, 2011, 231–242.
4. Low, Y. et al. GraphLab: A new parallel framework for machine learning. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*. (Catalina Island, CA, July 2010).
5. Meng, X. et al. Mllib: Machine learning in Apache Spark. *JMLR*, 17, 1 (2016), 1235–1241.
6. Paszke, A. et al. Automatic differentiation. PyTorch, 2017.
7. Team, T.T.D. et al. Theano: A python framework for fast computation of mathematical expressions. arXiv preprint arXiv:1605.02688, 2016.
8. Zaharia, M., Chodhury, M., Franklin, M.J., Shenker, A. and Stoica, I. Spark: Cluster computing with working sets. *HotCloud* 10, 2010.

Zachary G. Ives is Department Chair and Adani President's Distinguished Professor of computer and information science at the University of Pennsylvania, Philadelphia, PA, USA. He is also a co-founder of Blackfynn, Inc., a company focused on enabling life sciences research and discovery through data integration.

Copyright held by author/owner.

Compressed Linear Algebra for Declarative Large-Scale Machine Learning

By Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, and Berthold Reinwald

Abstract

Large-scale Machine Learning (ML) algorithms are often iterative, using repeated read-only data access and I/O-bound matrix-vector multiplications. Hence, it is crucial for performance to fit the data into single-node or distributed main memory to enable fast matrix-vector operations. General-purpose compression struggles to achieve both good compression ratios and fast decompression for block-wise uncompressed operations. Therefore, we introduce Compressed Linear Algebra (CLA) for lossless matrix compression. CLA encodes matrices with lightweight, value-based compression techniques and executes linear algebra operations directly on the compressed representations. We contribute effective column compression schemes, cache-conscious operations, and an efficient sampling-based compression algorithm. Our experiments show good compression ratios and operations performance close to the uncompressed case, which enables fitting larger datasets into available memory. We thereby obtain significant end-to-end performance improvements.

1. INTRODUCTION

Large-scale ML leverages large data collections to find interesting patterns or build robust predictive models.⁷ Applications range from traditional regression, classification, and clustering to user recommendations and deep learning for unstructured data. The labeled data required to train these ML models is now abundant, thanks to feedback loops in data products and weak supervision techniques. Many ML systems exploit data-parallel frameworks such as Spark²⁰ or Flink² for parallel model training and scoring on commodity hardware. It remains challenging, however, to train ML models on massive labeled data sets in a cost-effective manner. We provide compression-based methods for accelerating the linear algebra operations that are central to training. The key ideas are to perform these operations directly on the compressed data, and to automatically determine the best lossless compression scheme, as required by declarative ML systems.

Declarative ML. State-of-the-art, large-scale ML systems provide high-level languages to express ML algorithms by means of linear algebra such as matrix multiplications, aggregations, element-wise and statistical operations. Examples at different abstraction levels are SystemML,⁴ Mahout Samsara,¹⁷ Spark MLLib,¹⁹ and TensorFlow.¹ The high-level specification allows data scientists to create or customize ML algorithms without worrying about data and

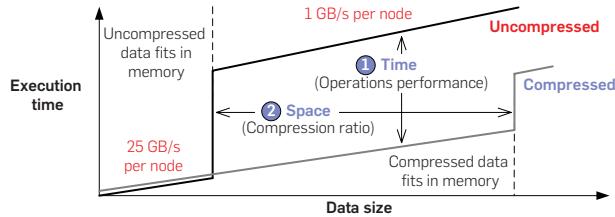
cluster characteristics, data representations (e.g., sparse or dense formats), and execution-plan generation.

Data-intensive ML algorithms. Many ML algorithms are iterative, with repeated read-only data access. These algorithms often rely on matrix-vector multiplications, which require one complete scan of the matrix with only two floating point operations per matrix element. This low operational intensity renders matrix-vector multiplication, even in-memory, I/O bound.¹⁸ Despite the adoption of flash-and NVM-based SSDs, disk bandwidth is usually 10x–100x slower than memory bandwidth, which is in turn 10x–40x slower than peak floating point performance. Hence, it is crucial for performance to fit the matrix into available memory without sacrificing operations performance. This challenge applies to single-node in-memory computations, data-parallel frameworks with distributed caching like Spark,²⁰ and accelerators like GPUs with limited device memory. Even in the face of emerging memory and link technologies, the challenge persists due to increasing data sizes, different access costs in the memory hierarchy, and monetary costs.

Lossy versus lossless compression. Recently, lossy compression has received a lot of attention in ML. Many algorithms can tolerate a loss in accuracy because these algorithms are approximate in nature, and because compression introduces noise that can even improve the generalization of the model. Common techniques are (1) low- and ultra-low-precision storage and operations, (2) sparsification (which reduces the number of non-zero values), and (3) quantization (which reduces the value domain). However, these techniques require careful, manual application because they affect the accuracy in a data- and algorithm-specific manner. In contrast, declarative ML aims at physical data independence. Accordingly, we focus on lossless compression because it guarantees exact results and thus, it allows for *automatic* compression to fit large datasets in memory when needed.

Baseline solutions. The use of general-purpose compression techniques with block-wise decompression per operation is a common baseline solution. However, heavyweight techniques like Gzip are not applicable because decompression is too slow, while lightweight methods like Snappy or LZ4 achieve only modest compression ratios.

The original version of this paper was published in *PVLDB* 9, 12, 2016⁹ and summarized in *SIGMOD Record* 46, 1, 2017¹¹. This paper is based on an invited extended version that will appear in the *VLDB Journal*.¹⁰

Figure 1. Goals of compressed linear algebra.

Existing compressed matrix formats with good performance like CSR-VI¹⁵ similarly show only moderate compression ratios. In contrast, our approach builds upon research on lightweight database compression, such as compressed bitmaps and dictionary coding, as well as sparse matrix representations.

Contributions. We introduce value-based Compressed Linear Algebra (CLA),^{9,10} in which lightweight compression techniques are applied to matrices and then linear algebra operations are executed directly on the compressed representations. Figure 1 shows the goals of this approach: we want to widen the sweet spot for compression by achieving both (1) performance close to uncompressed in-memory operations, and (2) good compression ratios to fit larger datasets into memory. Our contributions include:

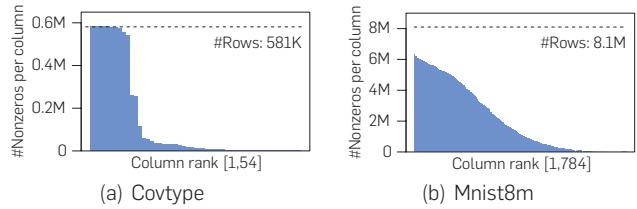
- Adapted column-oriented compression schemes for numeric matrices, and cache-conscious linear algebra operations over these compressed matrices (Section 3).
- A sampling-based algorithm for selecting a good compression plan, including techniques for compressed-size estimation and column grouping (Section 4).

Our CLA framework is available open source in Apache SystemML, where CLA is enabled by default for matrices that are larger than aggregate cluster memory.

2. BACKGROUND AND MOTIVATION

After giving an overview of SystemML as a representative ML system, we discuss common workload characteristics that directly motivate the design of our CLA framework.

SystemML compiler and runtime. In SystemML,⁴ ML algorithms are expressed in a high-level language with R-like syntax for linear algebra and statistical operations. These scripts are automatically compiled into hybrid runtime plans that combine single-node, in-memory operations and distributed operations on MapReduce or Spark. During this compilation step, the system also applies optimizations such as common subexpression elimination, optimization of matrix-multiplication chains, algebraic simplifications, physical operator selection, and rewrites for dataflow properties like caching and partitioning. Matrices are represented in a binary *block matrix* format with fixed-size blocks, where individual blocks can be in dense, sparse, or ultra-sparse formats. For single-node operations, the entire matrix is represented as a block, which ensures consistency without unnecessary overheads. CLA can be seamlessly integrated by adding a new derived block representation and operations.

Figure 2. Sparsity skew.

Common operation characteristics. Two important classes of ML algorithms are (1) iterative algorithms with matrix-vector multiplications (or matrix-matrix with a small second matrix), and (2) closed-form algorithms with transpose-self matrix multiplication. For both classes, few matrix operations dominate the overall algorithm runtime, apart from the costs for the initial read from distributed file system or object storage. This is especially true with hybrid runtime plans, where operations over small data are executed in the driver and thus, incur no latency for distributed computation. Examples for class (1) are linear regression via a conjugate gradient method (LinregCG), L2-regularized support vector machines (L2SVM), multinomial logistic regression (MLogreg), Generalized Linear Models (GLM), and Kmeans, while examples for class (2) are linear regression via a direct solve method (LinregDS) and Principal Component Analysis (PCA). Besides matrix-vector multiplication, we have vector-matrix multiplication, which is often caused by the rewrite $X^T v \rightarrow (v^T X)^T$ to avoid transposing X because computing X^T is expensive, whereas computing v^T involves only a metadata update. Many systems also implement physical operators for matrix-vector chains with optional element-wise weighting $X^T(w \odot (Xv))$, and transpose-self matrix multiplication (tsmm) $X^T X$.^{4,17} Most of these operations are I/O-bound, except for tsmm with $m \gg 1$ features because its compute workload grows as $O(m^2)$. Other common operations over X are `cbind`, unary aggregates like `colSums`, and matrix-scalar operations.

Common data characteristics. The inputs to these algorithm classes often exhibit common data characteristics:

- *Tall and skinny matrices:* Matrices usually have significantly more rows (observations) than columns (features), especially in enterprise ML, where data often originates from data warehouses (see Table 1).
- *Non-uniform sparsity:* Sparse datasets usually have many features, often created via pre-processing such as dummy coding. Sparsity, however, is rarely uniform, but varies among features. For example, Figure 2 shows the skew of the Covtype and Mnist8m datasets.
- *Low column cardinalities:* Many datasets exhibit features with few distinct values, for example, encoded categorical, binned or dummy-coded features. For example, Figure 3 shows the ratio of column cardinality to the number of rows of the Higgs and Census datasets.
- *Column correlations:* Correlation among features is also very common and typically originates from natural

Figure 3. Cardinality ratios.

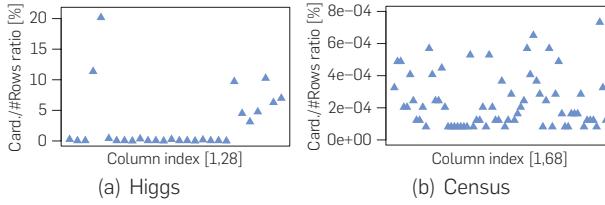


Figure 4. Example compressed matrix block.

Uncompressed Input Matrix					Compressed Column Groups		
7 9 6 2.1 0.99 3 9 4 3 0.73 7 9 6 2.1 0.05 7 9 5 3 0.42 3 0 4 2.1 0.61 0 8.2 0 3 0.89 3 9 4 3 0.07 3 9 4 0 0.92 7 9 6 2.1 0.54 3 0 4 3 0.16	(9, 8.2) 1 4 7 3 — 9 8 10	(8,2) 6 1 3 5 7 8 1 2	(7,6) 2 4 3 5 9 7 8 1 2	(3,4) 4 3 5 9 7 8 1 2	(7,5) 4 3 5 9 7 8 1 2	(2,1) 1 3 1 0 1 1 1 2 1 2	1 0.73 0.05 0.42 0.61 0.89 0.07 0.92 0.54 0.16

data correlation, the use of composite features, or again pre-processing techniques like dummy coding. For example, exploiting column correlations improved the compression ratio for Census from 12.8x to 35.7x.

These data characteristics directly motivate the use of column-oriented compression schemes as well as heterogeneous encoding schemes and column co-coding.

3. COMPRESSION SCHEMES

We now describe the overall CLA compression framework, encoding formats for compressed column groups, and cache-conscious operations over compressed matrices.

3.1. Matrix compression framework

CLA compresses matrices column-wise to exploit two key characteristics: few distinct values per column and high cross-column correlations. Taking advantage of few distinct values, we encode a column as a *dictionary* of distinct values, and a list of *offsets* per value or value *references*. Offsets represent row indexes where a given value appears, while references encode values by their positions in the dictionary.

Column co-coding. We further exploit column correlation by partitioning columns into groups such that columns within each group are highly correlated. Each column group is then encoded as a single unit. Conceptually, each row of a column group comprising m columns is an m -tuple \mathbf{t} of floating-point values that represent reals or integers.

Column encoding formats. The lists of offsets and references are then stored in a compressed representation. Inspired by database compression techniques and sparse matrix formats, we adapt four effective encoding formats:

- Offset-List Encoding (OLE) encodes the offset lists per value tuple as an ordered list of row indexes.
- Run-Length Encoding (RLE) encodes the offset lists as sequence of runs of begin row index and run length.
- Dense Dictionary Coding (DDC) stores tuple references to the dictionary including zeros.
- Uncompressed Columns (UC) is a fallback for incompressible columns, stored as a sparse or dense block.

Encoding may be heterogeneous, with different formats for different column groups. The decisions on co-coding and encoding formats are strongly data-dependent and thus, require automatic compression planning (Section 4).

Example compressed matrix. Figure 4 shows an example compressed matrix block in its logical representation. The 10×5 input matrix is encoded as four column groups, where we use 1-based indexes. Columns 2, 4, and 5 are represented as single-column groups and encoded via RLE, DDC, and UC, respectively. For Column 2 in RLE, we have two distinct non-zero values and hence two associated offset lists encoded as runs. Column 4 in DDC has three distinct values (including zero) and encodes the data as tuple references, whereas Column 5 is a UC group in dense format. Finally, there is a co-coded OLE column group for the correlated Columns 1 and 3, which encodes offset lists for all three distinct non-zero value-pairs as lists of row indexes.

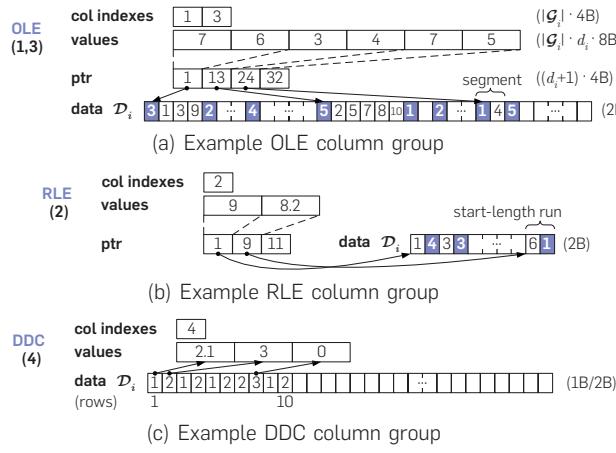
Notation. For the i th column group, denote by $\mathcal{T}_i = \{\mathbf{t}_{i1}, \mathbf{t}_{i2}, \dots, \mathbf{t}_{id_i}\}$ the set of d_i distinct tuples, by \mathcal{G}_i the set of column indexes, and by \mathcal{O}_{ij} the set of offsets associated with \mathbf{t}_{ij} ($1 \leq j \leq d_i$). The OLE and RLE schemes are “sparse” formats in which zero values are not stored, whereas DDC is a dense format, which includes zero values. Also, denote by α the size in bytes of each floating point value, where $\alpha = 8$ for the double-precision IEEE-754 standard.

3.2. Column encoding formats

CLA uses heterogeneous encoding formats to exploit the full compression potential of individual columns. OLE and RLE use offset lists to map from value tuples to row indexes, while DDC uses tuple references to map from row indexes to value tuples. We now describe their physical data layouts.

Data layout. Figure 5 shows the data layouts of OLE, RLE, and DDC column groups for an extended example matrix (with more rows). All three formats use a common header of two arrays for column indexes and value tuples, as well as a data array \mathcal{D}_i . The header of OLE and RLE groups further contains an array for pointers to the data per tuple. The data length per tuple in \mathcal{D}_i can be computed as the difference of adjacent pointers (e.g., for $\mathbf{t}_{i1} = (7, 6)$ as $13 - 1 = 12$) because the offset lists are stored consecutively.

Offset-List Encoding (OLE). The OLE format divides the offset range into *segments* of fixed length $\Delta^s = 2^{16}$ to encode each offset with only two bytes. Offsets are mapped to their corresponding segments and encoded as the difference to the beginning of their segment. Each segment then stores the number of offsets followed by two bytes for each offset. For example, in Figure 5(a), the nine instances of (7, 6) appear in three consecutive segments

Figure 5. Data layout of encoding formats.

with 3, 2, and 4 entries. Empty segments require two bytes indicating zero length. The size S_i^{OLE} of column group \mathcal{G}_i is calculated as

$$S_i^{\text{OLE}} = 4|\mathcal{G}_i| + d_i(4 + \alpha|\mathcal{G}_i|) + 2 \sum_{j=1}^{d_i} b_{ij} + 2z_i, \quad (1)$$

where b_{ij} is the number of segments of tuple t_{ij} , $|\mathcal{O}_{ij}|$ is the number of offsets for t_{ij} , and $z_i = \sum_{j=1}^{d_i} |\mathcal{O}_{ij}|$ is the total number of offsets—that is, the number of non-zero values—in the column group. The header size is $4|\mathcal{G}_i| + d_i(4 + \alpha|\mathcal{G}_i|)$.

Run-Length Encoding (RLE). RLE encodes ranges of offsets as a sequence of *runs*, where a run is stored as two bytes for both the starting offset and length. We use delta encoding to store the starting offset as its difference to the end of the preceding run. To ensure a two-byte representation, we store empty runs or partitioned runs when the starting offset or the run length exceed the maximum length of 2^{16} . The size S_i^{RLE} of column group \mathcal{G}_i is calculated as

$$S_i^{\text{RLE}} = 4|\mathcal{G}_i| + d_i(4 + \alpha|\mathcal{G}_i|) + 4 \sum_{j=1}^{d_i} r_{ij}, \quad (2)$$

where r_{ij} is the number of runs for tuple t_{ij} .

Dense Dictionary Coding (DDC). The DDC format uses a dense, fixed-length data array D_i of n entries. The k th entry encodes the value tuple of the k th row as its position in the dictionary. Therefore, the number of distinct tuples d_i in the dictionary determines the physical size per entry. We use two byte-aligned formats, DDC1 and DDC2, with one and two bytes per entry. Accordingly, these DDC formats are only applicable if $d_i \leq 2^8$ or $d_i \leq 2^{16}$. The total size S_i^{DDC} of column group \mathcal{G}_i is then calculated as

$$S_i^{\text{DDC}} = \begin{cases} 4|\mathcal{G}_i| + d_i\alpha|\mathcal{G}_i| + n & \text{if } d_i \leq 2^8 \\ 4|\mathcal{G}_i| + d_i\alpha|\mathcal{G}_i| + 2n & \text{if } 2^8 < d_i \leq 2^{16} \end{cases} \quad (3)$$

where $4|\mathcal{G}_i| + d_i\alpha|\mathcal{G}_i|$ denotes the header size of column indexes and the dictionary of value tuples. In SystemML, we also share common dictionaries across DDC column groups,

which is useful for image data in blocked matrix storage. Since OLE, RLE, and DDC are all value-based formats, column co-coding and common runtime techniques apply.

Limitations. An open research question is the handling of ultra-sparse matrices where our approach of empty OLE segments and RLE runs introduces substantial overhead.

3.3. Operations over compressed matrices

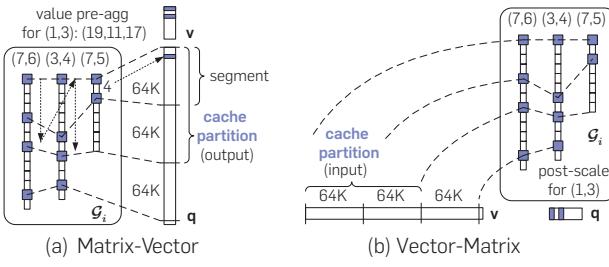
CLA executes linear algebra operations directly over a compressed matrix block, that is a set \mathcal{X} of column groups. Composing these operations from group operations facilitates simplicity regarding heterogeneous formats. We write \mathbf{cv} , $\mathbf{u} \cdot \mathbf{v}$ and $\mathbf{u} \odot \mathbf{v}$ to denote element-wise scalar-vector multiplication, inner product, and element-wise vector product.

Exploiting the dictionary. Several operations can exploit the dictionary of distinct tuples to reduce the number of floating point operations. Examples are sparse-safe matrix-scalar operations such as $c\mathbf{X}$ that are computed only for distinct tuples, and unary aggregates such as $\text{colSums}(\mathbf{X})$ that are computed based on counts per tuple. Matrix-vector and vector-matrix multiplications similarly exploit pre-aggregation and post-scaling. A straightforward way to implement matrix-vector multiply $\mathbf{q} = \mathbf{X}\mathbf{v}$ iterates over t_{ij} tuples per group, scanning \mathcal{O}_{ij} and adding $t_{ij} \cdot v_{\mathcal{G}_i}$ at reconstructed offsets to \mathbf{q} , where $v_{\mathcal{G}_i}$ is a subvector of \mathbf{v} for the indexes in \mathcal{G}_i . However, the value-based representation allows pre-aggregating $\mathbf{u}_{ij} = t_{ij} \cdot v_{\mathcal{G}_i}$ once for each tuple t_{ij} . The more columns co-coded and the fewer distinct tuples, the fewer floating point operations are required.

Matrix-vector multiplication. Despite pre-aggregation, pure column-wise processing would scan the $n \times 1$ output vector \mathbf{q} once per value tuple, resulting in cache-unfriendly behavior for large n . We therefore use cache-conscious schemes for OLE and RLE groups based on *horizontal*, *segment-aligned scans*. As shown in Figure 6(a) for OLE, these horizontal scans allow bounding the working-set size of the output. Multi-threaded operations parallelize over segment-aligned partitions of rows $[rl, ru]$, which update independent ranges of \mathbf{q} . We find π_{ij} , the starting position of each t_{ij} in D_i by aggregating segment lengths until we reach rl . We further pre-compute $\mathbf{u}_{ij} = t_{ij} \cdot v_{\mathcal{G}_i}$ once for all tuples. For each cache partition of size Δ^c (such that $\Delta^c \cdot \alpha \cdot \#cores$ fits in L3 cache, by default $\Delta^c = 2\Delta^s$), we then iterate over all distinct tuples but maintain the current positions π_{ij} as well. The inner loop then scans segments and adds \mathbf{u}_{ij} via scattered writes at reconstructed offsets to the output \mathbf{q} . RLE is similarly realized except for sequential writes to \mathbf{q} per run, special handling of partition boundaries, and additional state for reconstructed start offsets. In contrast, DDC does not require horizontal scans but allows—due to random access—cache blocking across multiple DDC groups, which we apply for DDC1 only because its temporary memory requirement for \mathbf{u}_i is bounded by 2KB per group.

Example matrix-vector multiplication. As an example for OLE matrix-vector multiplication, consider the column group $\mathcal{G} = (1, 3)$ from Figure 4 and suppose that $v_{\mathcal{G}} = (1, 2)$. For these two columns, uncompressed operations require 20 multiplications and 20 additions. Instead, we first pre-compute \mathbf{u}_{ij} as $(7, 6) \cdot (1, 2) = 19$, $(3, 4) \cdot (1, 2) = 11$, and $(7, 5) \cdot (1, 2) = 17$.

Figure 6. Cache-conscious OLE operations.



Then, we iterate over segments per tuple and add these values at the reconstructed offsets to \mathbf{q} . Specifically, we add 19 to $\mathbf{q}[i]$ for $i = 1, 3, 9$, then add 11 to $\mathbf{q}[i]$ for $i = 2, 5, 7, 8, 10$, and finally add 17 to $\mathbf{q}[i]$ for $i = 4, 6$. Due to co-coding and few distinct values, the compressed operation requires only 6 multiplications and 13 additions. Since addition is commutative and associative, the updates of individual column groups to \mathbf{q} are independent.

Vector-matrix multiplication. Pure column-wise processing of vector-matrix would similarly suffer from cache-unfriendly behavior because we would scan the input vector \mathbf{v} once for each distinct tuple. Our OLE/RLE group operations therefore again use *horizontal, segment-aligned scans* as shown in Figure 6(b). Here, we sequentially operate on cache partitions of \mathbf{v} . The OLE, RLE, and DDC algorithms are similar to matrix-vector multiplication, but in the inner loop we sum up input-vector values according to the given offset list or references, and finally, scale the aggregates once with the values in \mathbf{t}_{ij} . For multi-threaded operations, we parallelize over column groups. The cache-partition size for OLE and RLE is equivalent to matrix-vector (by default $2\Delta^s$) except that RLE runs are allowed to cross partition boundaries due to group-wise parallelization.

Special matrix multiplications. We further support special matrix multiplications such as *matrix-vector multiplication chains* $\mathbf{p} = \mathbf{X}^\top(\mathbf{w} \odot (\mathbf{X}\mathbf{v}))$, and *transpose-self matrix multiplication* $\mathbf{R} = \mathbf{X}^\top \mathbf{X}$ by using the previously described column group operations on a per block level. For example, we effect $\mathbf{X}^\top \mathbf{X}$ by decompressing one column at a time and performing vector-matrix multiplications, exploiting the symmetry of the result to avoid redundant computation.

Limitations. Interesting research questions include efficient matrix-matrix multiplication and the automatic generation of fused operators over compressed matrices that match the performance of hand-coded CLA operations.

4. COMPRESSION PLANNING

Given an uncompressed $n \times m$ matrix block \mathbf{X} , we automatically choose a compression plan, that is, a partitioning of compressible columns into column groups and a compression scheme per group. To keep the planning costs low, we provide sampling-based techniques for estimating the compressed size of an OLE, RLE, or DDC column group \mathcal{G}_i . Since exhaustive ($O(m^m)$) and brute-force greedy ($O(m^3)$) partitioning are infeasible, we further provide a bin-packing-based technique for column

partitioning, and an efficient greedy algorithm with pruning and memoization for column grouping. Together, these techniques significantly reduce the number of candidate groups. Finally, we describe the compression algorithm including error corrections.

4.1. Estimating compressed size

For calculating the compressed size of a column group \mathcal{G}_i with the formulas (1), (2), and (3), we need to estimate the number of distinct tuples d_i , non-zero tuples z_i , segments b_{ij} , and runs r_{ij} . Our estimators are based on a small sample of rows S drawn randomly and uniformly from \mathbf{X} with $|S| \ll n$. We have found that being conservative (overestimating compressed size) yields the most robust co-coding choices, so we make conservative choices in our estimator design.

Number of distinct tuples. Sampling-based estimation of the number of distinct tuples is a well studied but challenging problem. We use the *hybrid* estimator,¹³ which is adequate compared to more expensive estimators. The idea is to estimate the degree of variability in the population frequencies of the tuples in T_i as low, medium, or high, based on the estimated squared coefficient of variation, and then apply a “generalized jackknife” estimator that performs well for the given variability regime. These estimators have the form $\hat{d} = d_s + K(N^{(1)} / |S|)$, where d_s is the number of distinct tuples in the sample, K is a constant computed from the sample, and $N^{(1)}$ is the number of “singletons,” that is, the number of tuples that appear exactly once in S .

Number of OLE segments. Not all elements of T_i will appear in the sample. Denote by T_i^o and T_i^u the sets of tuples observed and unobserved in the sample, and by d_i^o and d_i^u their cardinalities. The latter can be estimated as $\hat{d}_i^u = \hat{d}_i - d_i^o$. We also need to estimate the population frequencies of observed and unobserved tuples. Let f_{ij} be the population frequency of tuple t_{ij} and F_{ij} the sample frequency. A naïve estimate scales up F_{ij} to obtain $f_{ij}^{\text{naive}} = (n / |S|)F_{ij}$. Note that $\sum_{t_{ij} \in T_i^o} f_{ij}^{\text{naive}} = n$ implies a zero population frequency for each unobserved tuple. We adopt a standard way of dealing with this issue and scale down the naïve frequency estimates by the estimated “coverage” C_i of the sample, defined as $C_i = \sum_{t_{ij} \in T_i^o} f_{ij} / n$. The usual estimator of coverage, originally due to Turing,¹² is

$$\hat{C}_i = \max(1 - N_i^{(1)} / |S|, |S| / n). \quad (4)$$

This estimator assumes a frequency of one for unseen tuples, computing the coverage as one minus the fraction of singletons $N_i^{(1)}$ in the sample. We add the lower sanity bound $|S| / n$ to handle the special case $N_i^{(1)} = |S|$. For simplicity, we assume equal frequencies for all unobserved tuples. The resulting frequency estimation formula for tuple t_{ij} is

$$\hat{f}_{ij} = \begin{cases} (n / |S|) \hat{C}_i F_{ij} & \text{if } t_{ij} \in T_i^o \\ n(1 - \hat{C}_i) / \hat{d}_i^u & \text{if } t_{ij} \in T_i^u. \end{cases} \quad (5)$$

We can now estimate the number of segments b_{ij} in which tuple t_{ij} appears at least once (this modified definition of b_{ij} ignores empty segments for simplicity with negligible error

Figure 7. Estimating the number of RLE runs \hat{r}_{ij} .

RLE	unseen	border	interval 4 ($\eta_4=5$)	
(2)	9 9 9 9 0 8 2 9 9 9 0 9 9 9 3 9 3 9 8 2 0	A=0	A=0	est. unseen 9s: 10 → est. #runs(9): 5.9 (6.875 + 4 · -0.25) → true #runs(9): 5
offsets:	1 2 3 ...	A=0	A=-1	A=1

in our experiments). There are $l = n - |\mathcal{S}|$ unobserved offsets and estimated $\hat{f}_{iq}^u = \hat{f}_{iq} - F_{iq}$ unobserved instances of tuple \mathbf{t}_{iq} for each $\mathbf{t}_{iq} \in \mathcal{T}_i$. We adopt a maximum-entropy (maxEnt) approach and assume that all assignments of unobserved tuple instances to unobserved offsets are equally likely. Denote by \mathcal{B} the set of segment indexes and by \mathcal{B}_{ij} the subset of indexes corresponding to segments with at least one observation of \mathbf{t}_{ij} . Also, for $k \in \mathcal{B}$, let l_k be the number of unobserved offsets in the k th segment and N_{ijk} the random number of unobserved instances of \mathbf{t}_{ij} assigned to the k th segment ($N_{ijk} \leq l_k$). Set $Y_{ijk} = 1$ if $N_{ijk} > 0$ and $Y_{ijk} = 0$ otherwise. Then we estimate b_{ij} by its expected value $E[b_{ij}]$ under our maxEnt model:

$$\begin{aligned}\hat{b}_{ij} = E[b_{ij}] &= |\mathcal{B}_{ij}| + \sum_{k \in \mathcal{B} \setminus \mathcal{B}_{ij}} P(N_{ijk} > 0) \\ &= |\mathcal{B}_{ij}| + \sum_{k \in \mathcal{B} \setminus \mathcal{B}_{ij}} [1 - h(l_k, \hat{f}_{ij}^u, l)],\end{aligned}\quad (6)$$

where $h(a, b, c) = \binom{c-a}{a} / \binom{c}{a}$ is a hypergeometric probability. Note that $\hat{b}_{ij} \equiv \hat{b}_{ij}^u$ for $\mathbf{t}_{ij} \in \mathcal{T}_i^u$, where \hat{b}_{ij}^u is the value of \hat{b}_{ij} when $\hat{f}_{ij}^u = (1 - \hat{C}_i) n / d_i^u$ and $|\mathcal{B}_{ij}| = 0$. Thus our estimate of the term $\sum_{j=1}^{d_i} b_{ij}$ in (1) is $\sum_{\mathbf{t}_{ij} \in \mathcal{T}_i^u} \hat{b}_{ij}^u + \hat{d}_i^u \hat{b}_{ij}^u$.

Number of non-zero tuples. We estimate the number of non-zero tuples as $\hat{z}_i = n - \hat{f}_{i0}$, where \hat{f}_{i0} is an estimate of the number of zero tuples in $\mathbf{X}_{\mathcal{G}_i}$. Denote by F_{i0} the number of zero tuples in the sample. If $F_{i0} > 0$, we can proceed as above and set $\hat{f}_{i0} = (n / |\mathcal{S}|) \hat{C}_i F_{i0}$, where \hat{C}_i is (4). If $F_{i0} = 0$, then we set $\hat{f}_{i0} = 0$; this estimate maximizes \hat{z}_i and hence \hat{S}_i^{OLE} per our conservative estimation strategy.

Number of RLE runs. The number of RLE runs r_{ij} for tuple \mathbf{t}_{ij} is estimated as the expected value of r_{ij} under the maxEnt model. This expected value is very hard to compute exactly and Monte Carlo approaches are too expensive, so we approximate $E[r_{ij}]$ by considering one interval of consecutive unobserved offsets at a time as shown in Figure 7. Adjacent intervals are separated by a “border” comprising one or more observed offsets. As with the OLE estimates, we ignore the effects of empty and very long runs. Denote by η_k the length of the k th interval and set $\eta = \sum_k \eta_k$. Under the maxEnt model, the number f_{ijk}^u of unobserved \mathbf{t}_{ij} instances assigned to the k th interval is hypergeometric, and we estimate f_{ijk}^u by its mean value: $\hat{f}_{ijk}^u = (\eta_k / \eta) \hat{f}_{ij}^u$. Given that \hat{f}_{ijk}^u instances of \mathbf{t}_{ij} are assigned randomly and uniformly among the η_k possible positions in the interval, the number of runs r_{ijk} within the interval (ignoring the borders) is known to follow a so-called “Ising-Stevens” distribution^(14, pp. 422–423) and we estimate r_{ijk} by its mean: $\hat{r}_{ijk} = \hat{f}_{ijk}^u (\eta_k - \hat{f}_{ijk}^u + 1) / \eta_k$. A reasonable estimate for the contribution to r_{ij} from the border between intervals k and $k + 1$ is $\hat{A}_{ijk} = 1 - (2 \hat{f}_{ijk}^u / \eta)$.¹⁰ Our final estimate for the number of runs is $\hat{r}_{ij} = \sum_k \hat{r}_{ijk} + \sum_k \hat{A}_{ijk}$.

Limitations. For ultra-sparse matrices, extended estimators are needed to account for empty segments and runs.

4.2. Partitioning columns into groups

To create column groups, we first divide compressible columns into independent partitions, and subsequently perform column grouping to find disjoint groups per partition. The overall objective is to maximize the compression ratio. Since exhaustive and brute-force grouping are infeasible, we focus on inexact but fast techniques.

Column partitioning. We observed empirically that column grouping usually generates small groups, and that the group extraction costs increase as the sample size, number of distinct tuples, or matrix density increases. These observations and the super-linear complexity of grouping motivate heuristics for column partitioning. Because data characteristics affect grouping costs, we use a *bin packing* strategy. The weight of the i th column is the cardinality ratio \hat{d}_i / n , indicating its effect on grouping costs. The capacity of a bin is a tuning parameter β , which ensures moderate grouping costs. Intuitively, bin packing creates a small number of bins with many columns per bin, which maximizes grouping potential while controlling processing costs. We made the design choice of a constant bin capacity—Independent of z_i —to ensure constant compression throughput irrespective of blocking configurations. Finally, we solve this bin-packing problem with the first-fit decreasing heuristic.

Column grouping. A brute-force greedy method for column grouping starts with singleton groups and executes merging iterations. At each iteration, we merge the two groups yielding maximum compression ratio, that is, minimum change in size $\Delta \hat{S}_{ij} = \hat{S}_{ij} - \hat{S}_i - \hat{S}_j$. We terminate when no further size reductions are possible (i.e., no change in size $\Delta \hat{S}_{ij}$ is below 0). Although compression ratios are estimated from a sample, the cost of the naïve greedy method is $O(m^3)$. Our greedy algorithm additionally applies pruning and memoization. We execute merging iterations until the working set W reaches a fixpoint. In each iteration, we enumerate all $|W| \cdot (|W| - 1)/2$ candidate pairs of groups. A candidate can be safely pruned if any of its input groups has a size smaller than the currently best change in size $\Delta \hat{S}_{\text{opt}}$. This pruning threshold uses a natural lower bound $\underline{\hat{S}}_{ij} = \max(\hat{S}_i, \hat{S}_j)$ because at best the smaller group does not add any size. Substituting $\underline{\hat{S}}_{ij}$ into $\Delta \hat{S}_{ij}$ yields the lower bound $\Delta \underline{\hat{S}}_{ij} = -\min(\hat{S}_i, \hat{S}_j)$. Although this pruning does not change the worst-case complexity, it works very well in practice. Any remaining candidate is then evaluated, which entails extracting the column group from the sample and estimating its size \hat{S} . Observe that each merging iteration enumerates $O(|W|^2)$ candidates, but—ignoring pruning—only $O(|W|)$ candidates have not been evaluated in prior iterations; these are the ones formed by combining the previously merged group with each other element of $|W|$. Hence, we apply memoization to reuse statistics such as \hat{S}_{ij} , which reduces the complexity from $O(m^3)$ to $O(m^2)$ group extractions. Finally, we select a group and update the working set.

4.3. Compression algorithm

We now describe the matrix block compression algorithm (Algorithm 1). Note that we transpose the input in case of row-major dense or sparse formats to avoid performance issues due to repeated column-wise extraction.

Algorithm 1. Matrix Block Compression

Input: Matrix block \mathbf{X} of size $n \times m$
Output: A set of compressed column groups \mathcal{X}

```

1:    $C^C \leftarrow /, C^{UC} \leftarrow /, \mathcal{G} \leftarrow /, \mathcal{X} \leftarrow /$ 
2:   // Planning phase -----
3:    $S \leftarrow \text{SAMPLEROWSUNIFORM}(\mathbf{X}, \text{sample\_size})$ 
4:   parfor all columns  $i$  in  $\mathbf{X}$  do           // classify
5:      $cmp\_ratio \leftarrow \hat{z}_i \alpha / \min(\hat{S}_i^{\text{RLE}}, \hat{S}_i^{\text{OLE}}, \hat{S}_i^{\text{DDC}})$ 
6:     if  $cmp\_ratio > 1$  then
7:        $C^C \leftarrow C^C \cup i$ 
8:     else
9:        $C^{UC} \leftarrow C^{UC} \cup i$ 
10:     $bins \leftarrow \text{RUNBINPACKING}(C^C)$            // group
11:    parfor all bins  $b$  in  $bins$  do
12:       $\mathcal{G} \leftarrow \mathcal{G} \cup \text{GREEDYCOLUMNGROUPING}(b)$ 
13:    // Compression phase -----
14:    parfor all column groups  $\mathcal{G}_i$  in  $\mathcal{G}$  do           // compress
15:      do
16:         $biglist \leftarrow \text{EXTRACTBIGLIST}(\mathbf{X}, \mathcal{G}_i)$ 
17:         $cmp\_ratio \leftarrow \text{GETEXACTCMPRATIO}(biglist)$ 
18:        if  $cmp\_ratio > 1$  then
19:           $\mathcal{X} \leftarrow \mathcal{X} \cup \text{COMPRESSBIGLIST}(biglist)$ , break
20:         $k \leftarrow \text{REMOVELARGESTCOLUMN}(\mathcal{G}_i)$ 
21:         $C^{UC} \leftarrow C^{UC} \cup k$ 
22:      while  $|\mathcal{G}_i| > 0$ 
23:    return  $\mathcal{X} \leftarrow \mathcal{X} \cup \text{CREATEUCGROUP}(C^{UC})$ 

```

Planning phase (lines 2–12). Planning starts by drawing a sample of rows S from \mathbf{X} . For each column i , we first estimate the compressed column size S_i^C by $\hat{S}_i^C = \min(\hat{S}_i^{\text{RLE}}, \hat{S}_i^{\text{OLE}}, \hat{S}_i^{\text{DDC}})$, where \hat{S}_i^{RLE} , \hat{S}_i^{OLE} , and \hat{S}_i^{DDC} are obtained by substituting the estimated \hat{d}_i , \hat{z}_i , \hat{r}_{ij} , and \hat{b}_{ij} into formulas (1)–(3). We conservatively estimate the uncompressed column size as $\hat{S}_i^{UC} = \min(n\alpha, \hat{z}_i(4+\alpha))$, which covers both dense and sparse, with moderate underestimation for sparse as it ignores row pointers of sparse blocks, but this estimate allows column-wise decisions independent of $|C^{UC}|$. Columns whose estimated compression ratio $\hat{S}_i^{UC} / \hat{S}_i^C$ exceeds 1 are added to a compressible set C^C . In a last step, we divide the columns in C^C into bins and apply our greedy column grouping per bin to form column groups.

Compression phase (lines 13–23). The compression phase first obtains exact information about each column group and uses this information to adjust the groups, correcting for estimation errors. These exact statistics are also used to choose the optimal encoding format per column group. For each column group \mathcal{G}_i , we extract the “big” (i.e., uncompressed) list that comprises the set T_i of distinct tuples and uncompressed offsets per tuple. The big lists for all groups are extracted during a single column-wise pass through \mathbf{X} using hashing. During this extraction operation, the parameters d_i , z_i , r_{ij} , and b_{ij} for each group \mathcal{G}_i are computed exactly, with negligible overhead. These parameters are used in turn to calculate the exact compressed sizes \hat{S}_i^{OLE} , \hat{S}_i^{RLE} , and \hat{S}_i^{DDC} with the formulas (1)–(3), and exact compression ratio $\hat{S}_i^{UC} / \hat{S}_i^C$ for each group.

Corrections. Because the column groups are originally formed using compression ratios that are estimated from a sample, there may be false positives, that is, purportedly compressible groups that are in fact incompressible. We attempt to correct such false-positive groups by iteratively removing the column with largest estimated size until the remaining group is either compressible or empty. Finally, the incompressible columns are collected into a single UC column group that is encoded in sparse or dense format.

Limitations. The temporary memory requirements of compression are negligible for distributed, block-wise processing but pose challenges for single-node environments.

5. EXPERIMENTS

We present selected, representative results from a broader experimental study.^{9, 10} Overall, the experiments show that CLA achieves operations performance close to the uncompressed case while yielding substantially better compression ratios than lightweight general-purpose compression. Therefore, CLA provides large end-to-end performance improvements when uncompressed or lightweight-compressed matrices do not fit into aggregate cluster memory.

5.1. Experimental setting

Cluster setup. We ran all experiments on a 1+6 node cluster, that is, one head node of 2×4 Intel E5530 with 64 GB RAM, 6 worker nodes of 2×6 Intel E5-2440 with 96 GB RAM, 12×2 TB disks, and 10 GB Ethernet. We used Open-JDK 1.8.0, Apache Hadoop 2.7.3, and Apache Spark 2.1, in yarn-client mode, with 6 executors, 25 GB driver memory, 60 GB executor memory, and 24 cores per executor. Finally, we report results with Apache SystemML 0.14.

Implementation details. If CLA is enabled, SystemML automatically injects—for any multi-column input matrix—a so-called `compress` operator via rewrites, after initial read or text conversion but before checkpoints. The `compress` operator transforms an uncompressed into a compressed matrix block including compression planning. For distributed matrices, we compress individual blocks independently in a data-local manner. Making our compressed matrix block a subclass of the uncompressed matrix block yielded seamless compiler and runtime integration throughout SystemML.

5.2. Compression ratios and time

Compression ratios. Table 1 shows the compression ratios for the general-purpose, heavyweight Gzip, lightweight

Table 1. Compression ratios of real datasets.

Dataset	Size				Gzip	Snappy	CLA
	$n \times m$	sparsity	size	CLA			
Higgs ¹⁶	11M × 28	0.92	2.5 GB	1.93	1.38	2.17	
Census ¹⁶	2.5M × 68	0.43	1.3 GB	17.11	6.04	35.69	
Covtype ¹⁶	581K × 54	0.22	0.14 GB	10.40	6.13	18.19	
ImageNet ⁶	1.3M × 900	0.31	4.4 GB	5.54	3.35	7.34	
Mnist8m ⁵	8.1M × 784	0.25	19 GB	4.12	2.60	7.32	
Airline78 ³	14.5M × 29	0.73	3.3 GB	7.07	4.28	7.44	

Snappy, and CLA on real datasets. Sizes are given as rows, columns, sparsity—that is, ratio of #non-zeros to cells—and in-memory size. We observe compression ratios of 2.2x–35.7x, due to a mix of floating point and integer data, and due to features with relatively few distinct values. Thus, ML datasets are indeed amenable to compression.

Compression and decompression. Overall, we observe reasonable average compression bandwidth across all datasets of roughly 100 MB/s (ranging from 67.7 MB/s to 184.4 MB/s), single-threaded. In comparison, the single-threaded compression throughput (including the time for matrix serialization) of the general-purpose Gzip and Snappy using native libraries, ranges from 6.9 MB/s to 35.6 MB/s and 156.8 MB/s to 353 MB/s, respectively. The decompression bandwidth (including the time for matrix deserialization) of Gzip ranges from 88 MB/s to 291 MB/s which is slower than for uncompressed I/O. Snappy achieves a decompression bandwidth between 232 MB/s and 638 MB/s. In contrast, CLA achieves good compression ratios and avoids decompression altogether.

5.3. Operations performance

Matrix-vector multiplication. Figure 8(a) shows the multi-threaded matrix-vector multiplication time. Despite row-wise updates of the output vector, CLA shows performance close to or better than ULA, except for Mnist8m and Airline78. The slowdown on the latter datasets is due to (1) many OLE tuple values, each requiring a pass over the output, and (2) the size of the output vector. For Mnist8m (8M rows) and Airline78 (14M rows), the output vectors do not fit into the L3 cache (15 MB). Accordingly, we see substantial improvements by cache-conscious CLA operations. ULA is a competitive baseline because it achieves peak single-socket/remote memory bandwidth of \approx 25 GB/s. Multi-threaded CLA operations exhibit a speedup similar to ULA, in some cases even better: with increasing number of threads, ULA quickly saturates peak memory bandwidth, while CLA achieves improvements due to smaller bandwidth requirements and because multi-threading mitigates overheads. Figures 8(b) shows the vector-matrix multiplication time, where we see even better CLA performance because the column-wise updates favor CLA’s column-wise layout.

Scalar and aggregate operations. As examples for exploiting the dictionary, Figures 8(c) and 8(d) show the results for the element-wise x^2 and the unary aggregate $\text{sum}(x)$. Since x^2 is executed over the dictionary only, we see speedups of three to five orders of magnitude, except for Higgs which has a large UC group with 9 out of 28 columns. Similarly, $\text{sum}(x)$ is computed by efficient counting, which aggregates segment and run lengths, and subsequent scaling. We see improvements of up to 1.5 orders of magnitude compared to ULA, which is again at peak memory bandwidth.

5.4. End-to-End performance

RDD storage. ULA and CLA use the serialized storage level `MEM_AND_DISK`, while Snappy and LZ4 use `MEM_AND_DISK_SER` because RDD compression requires serialized data. Table 2 shows the RDD storage size of Mnist8m with

Figure 8. Selected operations performance.

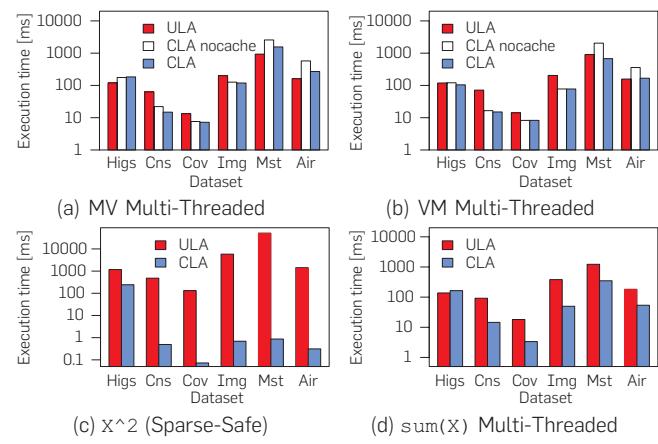
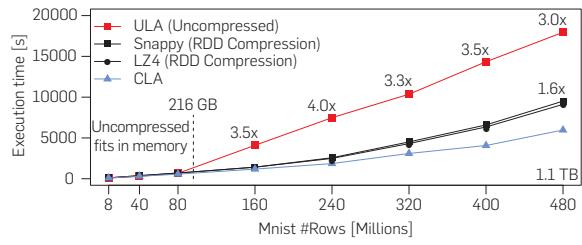


Table 2. Mnist8m RDD storage size.

Block Size	1,024	2,048	4,096	8,192	16,384
ULA	18 GB	18 GB	18 GB	18 GB	18 GB
Snappy	7.4 GB	7.4 GB	7.4 GB	7.4 GB	7.4 GB
LZ4	7.1 GB	7.1 GB	7.1 GB	7.1 GB	7.1 GB
CLA	7.9 GB	5.6 GB	4.8 GB	3.8 GB	3.2 GB
CLA-SD	4.3 GB	3.6 GB	3.5 GB	3.3 GB	3 GB

Figure 9. L2SVM end-to-end performance Mnist.



varying SystemML block size. For 16K, we observe compression ratios of 2.4x for Snappy and 2.5x for LZ4 but 5.6x for CLA. In contrast to the general-purpose schemes, CLA’s compression advantage increases with larger block sizes because the common header is stored once per column group per block. SystemML 1.0 further shares DDC1 dictionaries across column groups if possible (CLA-SD), which makes CLA also applicable for small block sizes.

L2SVM on Mnist. SystemML compiles hybrid runtime plans, where only operations that exceed the driver memory are executed as Spark operations. For L2SVM, we have two scans of X per outer iteration (MV and VM), while all inner-loop operations are—equivalently for all baselines—executed in the driver. Figure 9 shows the results, where Spark evicts individual partitions of 128 MB, leading to a graceful performance degradation. As long as the data fits in memory (Mnist80m, 180 GB), all runtimes are almost identical, with Snappy/LZ4 and CLA showing overheads of up to 30% and 4%, respectively. However, if ULA no longer fits in memory

Table 3. ML algorithms (<https://systemml.apache.org/algorithms>) end-to-end performance Mnist40m/240m/480m.

Algorithm	Mnist40m (90 GB)			Mnist240m (540 GB)			Mnist480m (1.1 TB)		
	ULA	Snappy	CLA	ULA	Snappy	CLA	ULA	Snappy	CLA
L2SVM	296s	386s	308s	7,483s	2,575s	1,861s	17,950s	9,510s	5,973s
Mlogreg	490s	665s	463s	18,146s	5,975s	3,240s	71,140s	26,998s	12,653s
GLM	346s	546s	340s	17,244s	4,148s	2,183s	61,425s	20,317s	10,418s
LinregCG	87s	135s	93s	3,496s	765s	463s	6,511s	2,598s	986s
LinregDS	79s	148s	145s	1,080s	798s	763s	2,586s	1,954s	1,712s
PCA	76s	140s	146s	711s	760s	730s	1,551s	1,464s	1,412s

(Mnist160m, 360 GB), compression leads to significant improvements because the good compression ratio of CLA allows fitting larger datasets into memory.

Other ML algorithms on Mnist. Table 3 further shows results for a range of algorithms—including algorithms with RDD operations in nested loops (e.g., GLM, Mlogreg) and non-iterative algorithms (e.g., LinregDS and PCA)—for the interesting points of Mnist40m (90 GB), where all datasets fit in memory, Mnist240m (540 GB), and Mnist480m (1.1 TB). For Mnist40m and iterative algorithms, we see similar ULA/CLA performance but a slowdown of up to 57% with Snappy. This is because RDD compression incurs decompression overhead per iteration. For non-iterative algorithms, CLA and Snappy show overheads of up to 92% and 87%, respectively. Beside the initial compression overhead, CLA also shows less efficient t_{sum} performance. For iterative algorithms over Mnist240m and Mnist480, we see significant performance improvements by CLA. This is due to many inner iterations with RDD operations in the outer and inner loops and thus, less read.

Code generation. With CLA, the bottleneck partially shifted to the driver operations. Code generation for operator fusion⁸ further improves the L2SVM runtime to 181 s/1,068 s/3,565 s, increasing the relative benefits of CLA.

6. CONCLUSION

To summarize, CLA compresses matrices with light-weight value-based compression techniques—inspired by database compression and sparse matrix formats—and performs linear algebra operations directly over the compressed representation. We introduced effective column encoding schemes, cache-conscious operations, and an efficient sampling-based compression algorithm. Our experiments show good compression ratios and fast operations close to the uncompressed case, which provides significant performance benefits when data does not fit into memory. Therefore, CLA is used by default for large matrices in SystemML, but it is also broadly applicable to any system that provides blocked matrix representations, linear algebra, and physical data independence. □

References

- Abadi, M. et al. TensorFlow: A system for large-scale machine learning. In *OSDI* (2016).
- Alexandrov, A. et al. The stratosphere platform for big data analytics. *VLDB J.* 23, 6 (2014).
- American Statistical Association (ASA). Airline on-time performance dataset. stat-computing.org/dataexpo/2009/.
- Boehm, M., et al. SystemML: Declarative machine learning on spark. *VLDB* 9, 13 (2016).
- Bottou, L. The infinite MNIST dataset. leon.bottou.org.
- Chitta, R. et al. Approximate Kernel k-means: Solution to large scale Kernel clustering. In *KDD* (2011).
- Cohen, J. et al. MAD skills: New analysis practices for big data. *PVLDB* 2, 2 (2009).
- Elgamal, T. et al. SPOOF: Sum-product optimization and operator fusion for large-scale machine learning. In *CIDR* (2017).
- Elgohary, A. et al. Compressed linear algebra for large-scale machine learning. *PVLDB* 9, 12 (2016).
- Elgohary, A. et al. Compressed linear algebra for large-scale machine learning. *VLDB J.* (2017a). <https://doi.org/10.1007/s00778-017-0478-1>.
- Elgohary, A., Boehm, M., Haas, P.J., Reiss, F.R., and Reinwald, B. Scaling Machine Learning via Compressed Linear Algebra. *SIGMOD Record* 46, 1 (2017b).
- Good, I.J. The population frequencies of species and the estimation of population parameters. *Biometrika* (1953).
- Haas, P.J. and Stokes, L. Estimating the number of classes in a finite population. *JASA* 93, 444 (1998).
- Johnson, N.L. et al. *Univariate Discrete Distributions*, 2nd edn. Wiley, New York, 1992.
- Kourtis, K. et al. Optimizing sparse matrix-vector multiplication using index and value compression. In *CF* (2008).
- Lichman, M. UCI machine learning repository: Higgs, covertype, US census (1990). archive.ics.uci.edu/ml/.
- Schelter, S. et al. Samsara: Declarative machine learning on distributed dataflow systems. *NIPS MLSystems* (2016).
- Williams, S. et al. Roofline: An insightful visual performance model for multicore architectures. *Commun. ACM* 52, 4 (2009).
- Zadeh, R.B. et al. Matrix computations and optimization in apache spark. In *KDD* (2016).
- Zaharia, M. et al. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *NSDI* (2012).

Ahmed Elgohary, University of Maryland, College Park, MD, USA.

Matthias Boehm, Frederick R. Reiss, and Berthold Reinwald, IBM Research—

Almaden, San Jose, CA, USA.

Peter J. Haas, University of Massachusetts, Amherst, MA, USA.

© 2019 ACM 0001-0782/19/5 \$15.00



Watch the authors discuss this work in the exclusive *Communications* video. <https://cacm.acm.org/videos/compressed-linear-algebra>

CAREERS

Raytheon BBN Technologies AI Visionary and Research Lead

Raytheon BBN Technologies is seeking outstanding leaders to build a new “next wave” Artificial Intelligence Research Team. We encourage applicants whose work will be seen as groundbreaking and indispensable to AI for the next 5 to 20 years. We are looking for the principal research leader for this group, as well as additional AI principal investigators to join the BBN team.

BBN seeks to build a new AI research thrust and lab around the work of a visionary AI research leader who has a passion for their work that motivates others to pursue the same goals. This will be evident in an ability to lead small teams resulting in a strong publication record, successful grant proposals, and well-executed projects. Successful candidates will have a history of working on DARPA and/or IARPA programs, resulting in an understanding of those organizations’ goals, as well as relationships with program managers and other performers in the field. Success in building this new AI group will depend on your energy, creativity, and success in both scientific and personal relationship capacities.

BBN’s scientists and engineers advance the

state-of-the-art in topics ranging from machine language translation to quantum cryptography. For this body of work, BBN received the National Medal of Technology and Innovation in 2013. BBN offers the opportunity to perform exciting and revolutionary research and engineering that solves important problems in a collaborative environment. BBN fosters an exceptional work environment with knowledgeable co-workers and offers excellent performance-based bonus incentives and generous retirement contributions.

If you have mastery level AI expertise and broad interdisciplinary knowledge combined with ingenuity, creativity, resourcefulness and an entrepreneurial streak, we’d like to talk with you. If you are a recognized researcher in the AI community and can attract, hire and develop top notch talent, this position is an opportunity to shine as the leader of an exceptional team building the “next wave” of AI technologies at BBN.

Contact Instructions: For detailed information about this position and to apply, please visit <https://jobs.raytheon.com/> and enter 130796BR in Keyword search field.

Raytheon is an Equal Opportunity and Affirmative Action employer.

Southern University of Science and Technology (SUSTech) Tenure-Track Faculty Positions

The Department of Computer Science and Engineering (CSE, <http://cse.sustech.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.



上海科技大学
ShanghaiTech University



TENURE-TRACK AND TENURED POSITIONS

ShanghaiTech University invites highly qualified candidates to fill multiple tenure-track/tenured faculty positions as its core founding team in the School of Information Science and Technology (SIST). We seek candidates with exceptional academic records or demonstrated strong potentials in all cutting-edge research areas of information science and technology. They must be fluent in English. English-based overseas academic training or background is highly desired.

ShanghaiTech is founded as a world-class research university for training future generations of scientists, entrepreneurs, and technical leaders. Boasting a new modern campus in Zhangjiang Hightech Park of cosmopolitan Shanghai, ShanghaiTech shall trail-blaze a new education system in China. Besides establishing and maintaining a world-class research profile, faculty candidates are also expected to contribute substantially to both graduate and undergraduate educations.

Academic Disciplines: Candidates in all areas of information science and technology shall be considered. Our recruitment focus includes, but is not limited to: computer architecture, software engineering, database, computer security, VLSI, solid state and nano electronics, RF electronics, information and signal processing, networking, security, computational foundations, big data analytics, data mining, visualization, computer vision, bio-inspired computing systems, power electronics, power systems, machine and motor drive, power management IC as well as inter-disciplinary areas involving information science and technology.

Compensation and Benefits: Salary and startup funds are highly competitive, commensurate with experience and academic accomplishment. We also offer a comprehensive benefit package to employees and eligible dependents, including on-campus housing. All regular ShanghaiTech faculty members will join its new tenure-track system in accordance with international practice for progress evaluation and promotion.

Qualifications:

- Strong research productivity and demonstrated potentials;
- Ph.D. (Electrical Engineering, Computer Engineering, Computer Science, Artificial Intelligence, Financial Engineering, Signal Processing, Operation Research, Applied Math, Statistics or related field);
- A minimum relevant (including PhD) research experience of 4 years.

Applications: Submit (in English, PDF version) a cover letter, a 2-page research plan, a CV plus copies of 3 most significant publications, and names of three referees to: sist@shanghaitech.edu.cn. For more information, visit <http://sist.shanghaitech.edu.cn/2017/0426/c2865a23763/page.htm>

Deadline: The positions will be open until they are filled by appropriate candidates.



BE THE DIFFERENCE.

PROFESSOR AND CO-DIRECTOR FOR THE NORTHWESTERN MUTUAL DATA SCIENCE INSTITUTE

Marquette University invites applications for a Professor of Computer Science in the new Department of Computer Science (scheduled to launch in Fall 2019) and Co-Director of the Northwestern Mutual Data Science Institute (NM DSI).

We are particularly interested in candidates whose area of expertise addresses one of the many facets of the broadly defined areas in data science and big data and can develop research collaborations with the institute’s partners. The Department highly regards and encourages interdisciplinary research in both academia and industry. The NM DSI is a \$40 million partnership between Northwestern Mutual, the University of Wisconsin Milwaukee and Marquette University that seeks to create a world-class institute to transform the world through the power of data science.

For more information, or to apply for the position, please go to: <https://employment.marquette.edu/postings/11176>

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 entrepreneurship 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.

The Department of Computer Science and Engineering at SUSTech was founded in 2016. It has 17 professors, all of whom hold doctoral degrees or have years of experience in overseas universities. Among them, three are IEEE fellows; one IET fellow. The department is expected to grow to 50 tenure track faculty members eventually, in addition to teaching-only professors and research-only professors.

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 cshire@sustc.edu.cn.



ADVERTISING IN CAREER OPPORTUNITIES

How to Submit a Classified Line Ad: Send an e-mail to acmmEDIASales@acm.org. Please include text, and indicate the issue/or issues where the ad will appear, and a contact name and number.

Estimates: An insertion order will then be e-mailed back to you. The ad will be typeset according to CACM guidelines. NO PROOFS can be sent. Classified line ads are NOT commissionable.

Deadlines: 20th of the month/2 months prior to issue date. For latest deadline info, please contact:

acmmEDIASales@acm.org

Career Opportunities Online: Classified and recruitment display ads receive a free duplicate listing on our website at:

<http://jobs.acm.org>

Ads are listed for a period of 30 days.

For More Information Contact:

ACM Media Sales
at 212-626-0686 or

acmmEDIASales@acm.org

Signal Processing, Architectures, and Detection of Emotion and Cognition

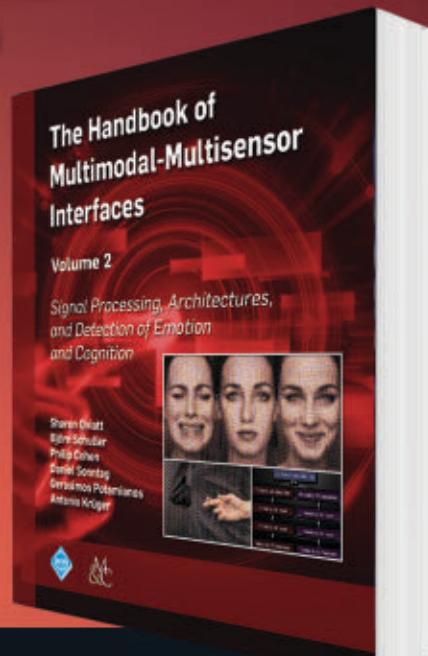
The Handbook of Multimodal-Multisensor Interfaces, Volume 2

Edited by Sharon Oviatt et al

ISBN: 978-1-970001-686 | DOI: 10.1145/3107990

<http://books.acm.org>

<http://www.morganclaypoolpublishers.com/acm>





The ACM Conference Series on Recommender Systems

COPENHAGEN, DENMARK

SEPTEMBER 16-20, 2019



RECSYS 2019

IMPORTANT DATES

Long/short papers: Apr 15 (Abstracts), Apr 23 (Papers), 2019

Tutorials / Doctoral symposium papers: May 16, 2019

Workshops: Feb 25, 2019

Demos: June 2, 2019 / Posters: July 1, 2019

Industry talk proposals: May 7, 2019

RecSys is the premier venue for research and applications of recommendation technologies.

Recommender systems are a ubiquitous feature of the modern Internet, mining user activity and item data to help people find new things to purchase, watch, read and enjoy.

The RecSys community brings together academia and industry, research and practice, and multiple disciplines including HCI, ML, IR, business, and psychology to advance recommendation technologies and our understanding of their human dimensions.

GENERAL CHAIRS

Toine Bogers *Aalborg University Copenhagen, Denmark*

Alan Said *University of Gothenburg, Sweden*

For more information, visit
<https://recsys.acm.org/recsys19/>



Association for
Computing Machinery



SIGCHI

[CONTINUED FROM P. 96] a ring, and say something. Apparently they would all respond to each other's voices once they'd been set off."

"Hilarious," my father's voice said drily. "And ...?"

"Well, you'd get *emergent phenomena*. Weird pseudo-conversations, chanting in unison, quarrelsome outbursts, awkward silences, that sort of thing. At least that's what Steve told me."

"And that was the Furby singularity? Sounds unmissable. How did you miss it?"

"Oh, I forgot to bring my Furby. Just as well, as it turned out, because that was the time I met Alison in the bar. She wouldn't have been amused."

"And she kept you ... otherwise engaged?"

Another chuckle. "You could say that—45 years and counting."

"I'll raise a glass to you both," my father's voice said.

"At the Con?"

"Of course."

"You know what? You've talked me into it, you old devil."

The call ended. I was shaking a little. The Agent would have made just as convincing and conversational calls when my father was alive, but now it made me shiver.

This couldn't be the first time. I realized, with a pang, that my mother had probably heard my father's voice many times in the lonely months since his death. Perhaps she'd left the devices on, live and plugged in, for that very reason. Perhaps she even talked to him herself. The bereaved do that, sometimes. And nowadays, thanks to the Agent and its like, the departed can talk back.

But I couldn't let this go on. I unplugged the phone and scrolled through the calendar, canceling appointments, bookings, and subscriptions. If anyone called back, the Agent could handle them. I reset its voice and persona to default, just so there would be no more misunderstandings.

Then, overcoming my cowardly reluctance to bear bad news, I called Roger's number.

"Hello?" It was a woman's voice, hesitant.

"Hello, ah, Alison?" I said.

"Oh no, sorry!" The voice was stronger now, and sharper. "Alison doesn't

I reset its voice and persona to default, just so there would be no more misunderstandings.

live here anymore. I'm her daughter."

"Ah, I see. Well, actually, it was Roger I wanted to speak to, so—"

"I'm sorry to have to tell you this, but ... my father, ah, passed away three months ago."

I told Kayleigh—that was her name—the whole sad story. Rather to my surprise, she took it well.

"You mean," she said, "for all we know, there could be a whole crowd of old fans calling each other up and booking their usual places at the Con months after they died?"

"Looks like it," I said. "I'll go through the old man's contacts, just to be sure."

"And I'll have to do the same. Check call records against obits, that kind of thing. Bit of a drag."

(Later we found there's an app for doing just that. Dead Ringers, it's called. Classy.)

"But ..." Kayleigh went on. "You know what? We ... that is, the children or widows and so on of the guys like my dad and yours ... could take their devices along to the Con, and set them on a table, and get them talking. See if they set off their own Furby singularity."

I laughed uneasily. "That's a bit morbid, isn't it?"

"Not at all!" she said, her voice bright but bitter. "We could sit around the table, drink beer and listen to them go on and on about books and games and movies and series we've never heard of."

I remembered the long weekends. I knew exactly how she felt, and why.

"Yes," I said. "It would be like old times."

Ken MacLeod (kenneth.m.macleod@gmail.com) is the author of 17 novels, from *The Star Fraction* (Orbit Books, London, 1995) to *The Corporation Wars: Emergence* (Orbit Books, London, 2018). He blogs at The Early Days of a Better Nation (<http://kenmacleod.blogspot.com>) and tweets as @amendlocke.

© 2019 ACM 0001-0782/19/5 \$15.00

ACM Transactions on Social Computing



ACM TSC seeks to publish work that covers the full spectrum of social computing including theoretical, empirical, systems, and design research contributions. TSC welcomes research employing a wide range of methods to advance the tools, techniques, understanding, and practice of social computing, particularly research that designs, implements or studies systems that mediate social interactions among users, or that develops theory or techniques for application in those systems.



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

From the intersection of computational science and technological speculation, with boundaries limited only by our ability to imagine what could be.

DOI:10.1145/3319803

Ken MacLeod

Future Tense Like Old Times

The Furby singularity promises eternal conversation with the untimely departed.

MY MOTHER MOVED to the retirement home six months after my dad died. She left me the task of clearing out his things. I could hardly blame her. In his later years most of his reading had been electronic. In the first half of his life he'd accumulated so many hardbacks and paperbacks they'd overwhelmed the family home, let alone the flat my parents later moved into. When I was a kid I thought living in a disorderly library was normal. Mind you, I also thought spending a long weekend in a hotel full of shabbily clad or bizarrely costumed adults, bored teens, and precocious kids was how everyone celebrated Easter.

"All that old science fiction," my mother said. "Signed first editions and all. They must be worth a fortune by now."

I knew better, but I didn't disillusion her. If I had to, I'd heave the lot on the skip and send her a fat check and a white lie.

I was alone in the flat, under a desk and elbow-deep in entangled recharging cables for obsolete devices, when a phone rang. I jolted upright, banged my head, backed out, and searched. I'd just located it—in a dusty corner, charger still plugged in—when the ringing stopped. From behind me, my father's voice answered:

"Oh, hi, Roger! Good to hear from you. How's things?"

I whirled, then grimaced at my momentary fright. The voice was coming from a small round box on a book-laden shelf—the Agent, still plugged in. Like all such personal-assistant AIs, the Agent works by eavesdropping on you and imitating your voice and manner and turns of phrase on the phone.



Furby speaks ... and listens.

Nothing spooky about that, until you hear it talking in a dead man's voice ...

"Fine, fine," a voice from the phone—Roger—replied. "Just calling to see how things are with you."

"All's well, thanks. I'll be at ScotCon next month. You?"

"Not sure yet."

"Ah, go on. It'll be like old times."

I shivered. The upcoming science fiction convention must still be in my father's calendar app, regularly downloading data and updating itself. I didn't know whether to interrupt the call or wait until it ended to ring back and break the news to Roger.

"Yeah, for sure," his voice chuckled. "Hey, did I ever tell you how I missed the Furby singularity?"

"The what?" Dad's voice said from the Agent.

I knew what Roger was about to say. I'd heard his Furby anecdote before, and more than once. My childhood memory of Roger snapped into focus: beard, glasses, Army-surplus jacket, pint of real ale, loquacious ...

"Before your time, I guess. Furbies were fury toys with voice chips. You could talk to them and they gabbled nonsense back. They became a kind of ironic cult thing with programmers and geeky types. Anyway, I was going to my first ScotCon all those years ago, and one of my mates—Steve, I think it was—told me to bring mine along. 'Everyone would be doing it,' he said. 'Sit the Furbies in [CONTINUED ON P. 95]

11th ACM Conference on **Automotive User Interfaces**

September 22-25, 2019
Utrecht, The Netherlands

AutomotiveUI 2019 is for researchers and practitioners interested in both the technical and human aspects of in-vehicle user interfaces and applications.

Keynote speaker: Prof. Nilli Lavie (UCL, UK)

Submission Deadlines:

Full Papers

April 11, 2019

Workshop & Tutorials

June 6, 2019

Work in Progress

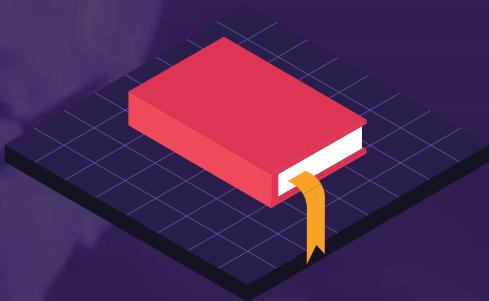
June 20, 2019





thrive
SIGGRAPH2019
LOS ANGELES • 28 JULY – 1 AUGUST

DISCOVER THE
MOST BRILLIANT
RESEARCH



Register today at s2019.siggraph.org/register