

COMMUNICATIONS

CACM.ACM.ORG

OF THE

ACM

07/2018 VOL.61 NO.07



Making Machine Learning Robust ...

Digital Nudging

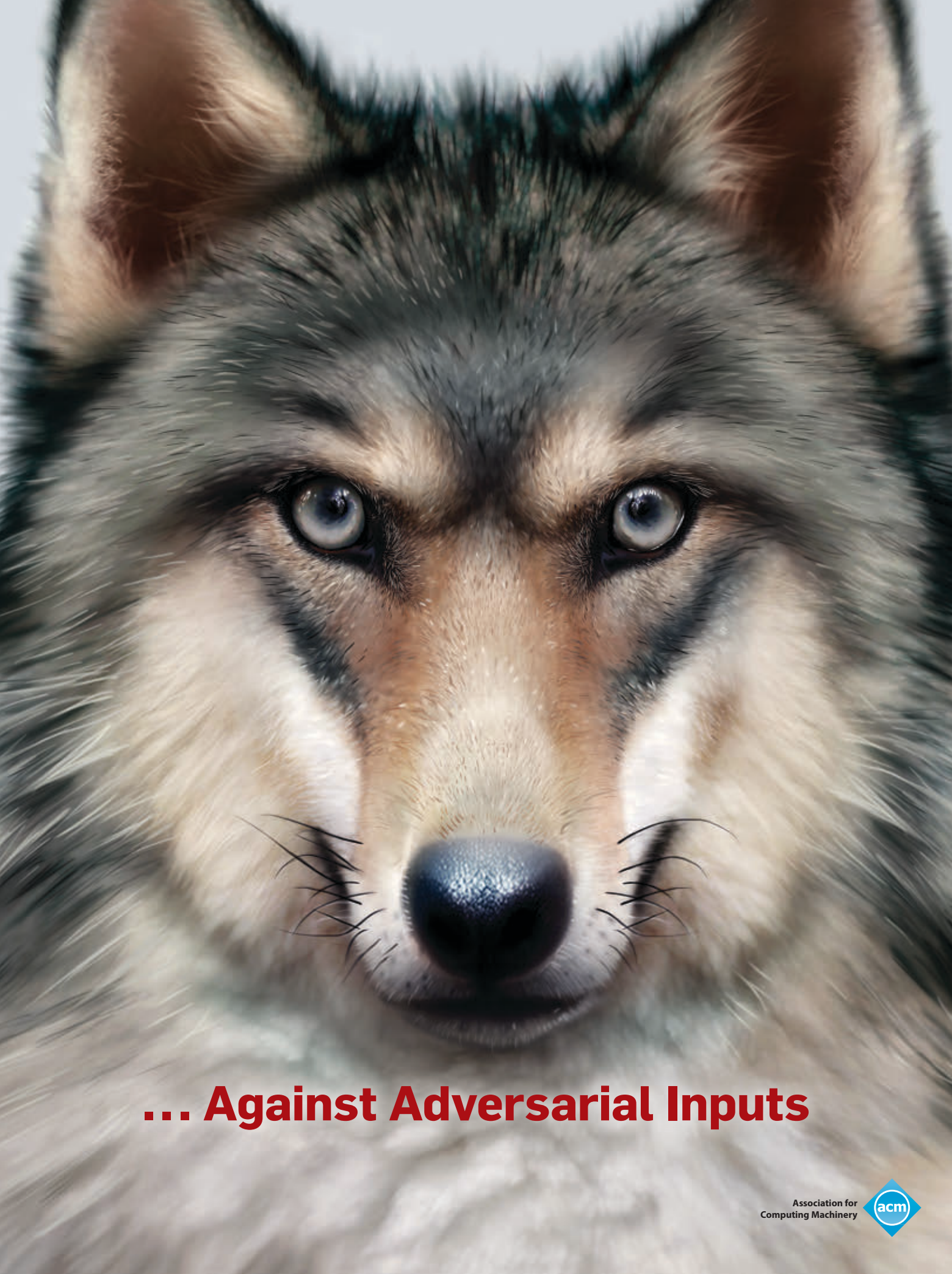
Smart Email Systems

Bitcoin Mining Is Vulnerable

How the Hippies Destroyed the Internet

Association for
Computing Machinery



A close-up, high-resolution photograph of a husky's face. The husky has light blue eyes, a black nose, and thick fur with grey, black, and tan markings. It is looking directly at the camera with a serious expression.

... Against Adversarial Inputs

Departments

- 5 **From the President**
Reflections on My Two Years
By Vicki L. Hanson
-
- 7 **Cerf's Up**
On Neural Networks
By Vinton G. Cerf
-
- 9 **Vardi's Insights**
How the Hippies Destroyed the Internet
By Moshe Y. Vardi
-
- 10 **Letters to the Editor**
Teach the Law (and the AI) 'Foreseeability'
-
- 12 **BLOG@CACM**
We Are Done with 'Hacking'
Today's programmers offer more valuable skills than simply being able to hack algorithms and make data structures, says Yegor Bugayenko.
-
- 29 **Calendar**
-
- 103 **Careers**

Last Byte

- 104 **Upstart Puzzles**
String Wars
By Dennis Shasha

News



- 15 **Why Cryptocurrencies Use So Much Energy—and What to Do About It**
The electricity consumption of mining for cryptocurrencies is becoming a real concern. Here's what to do about it.
By Logan Kugler
-
- 18 **You've Got Mail!**
And that's not all. Email is not what it used to be.
By Gary Anthes
-
- 20 **Bringing the Internet to the (Developing) World**
A growing number of low-cost (and free!) solutions aim to open the Internet to developing regions.
By Keith Kirkpatrick

Viewpoints

- 24 **Legally Speaking**
Copyright Blocks a News-Monitoring Technology
An evolving technological landscape has made application of copyright law increasingly difficult.
By Pamela Samuelson
-
- 27 **Economic and Business Dimensions**
Blockchain Revolution without the Blockchain?
Most of the suggested benefits of blockchain technologies do not come from elements unique to the blockchain.
By Hanna Halaburda
-
- 30 **Broadening Participation**
Beyond Diversity
Considering the confluence of research questions and sociopolitical dynamics.
By Alex Ahmed
-
- 33 **Viewpoint**
A New Perspective on Computational Thinking
Addressing its cognitive essence, universal value, and curricular practices.
By Osman Yaşar
-
- 40 **Viewpoint**
The Case for Disappearing Cyber Security
A proposal for keeping cyber security both out of sight and out of mind for end users.
By Josiah Dykstra and Eugene H. Spafford



Association for Computing Machinery
Advancing Computing as a Science & Profession

Practice



44

- 44 **C Is Not a Low-Level Language**
Your computer is not a fast PDP-11.
By David Chisnall

- 49 **How to Come Up with Great Ideas**
Think like an entrepreneur.
By Kate Matsudaira

- 52 **Research for Practice:
Toward a Network
of Connected Things**
Expert-curated guides to
the best of CS research.
By Deepak Vasisht

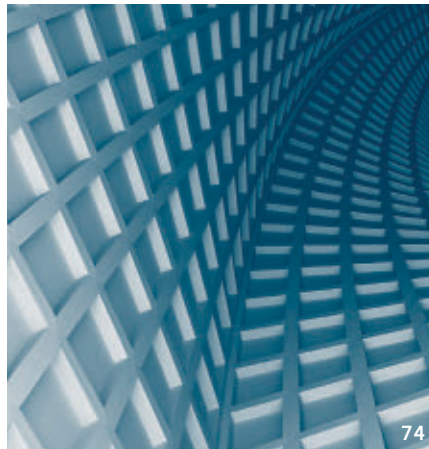


Articles' development led by **acmqueue**
queue.acm.org

**About the Cover:**

This month's cover story (p. 56) tells of the challenges of keeping machine learning strong amid adversarial attacks that have the power to, among other things, change a machine's perception of an image with a few pixel flips. Cover illustrations by Arn0.

Contributed Articles



74

- 56 **Making Machine Learning
Robust Against Adversarial Inputs**
Such inputs distort how machine-learning-based systems are able to function in the world as it is.
By Ian Goodfellow, Patrick McDaniel, and Nicolas Papernot



Watch the authors discuss their work in this exclusive *Communications* video.
<https://cacm.acm.org/videos/making-machine-learning-robust-against-adversarial-inputs>

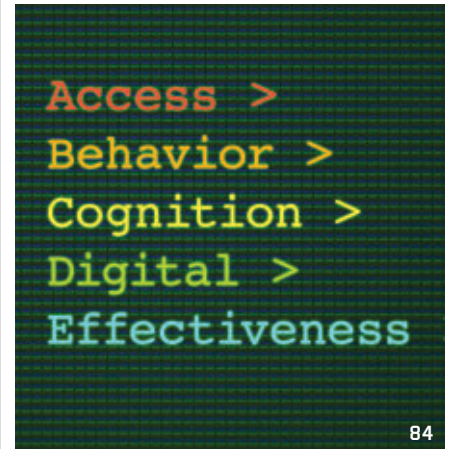
- 67 **Digital Nudging: Guiding
Online User Choices
through Interface Design**
Designers can create designs that nudge users toward the most desirable option.
By Christoph Schneider, Markus Weinmann, and Jan vom Brocke



Watch the authors discuss their work in this exclusive *Communications* video.
<https://cacm.acm.org/videos/digital-nudging>

- 74 **Always Measure One Level Deeper**
Performance measurements often go wrong, reporting surface-level results that are more marketing than science.
By John Ousterhout

Review Articles



84

- 84 **The ABCs of Effectiveness
in the Digital Society**
Digital effectiveness is not the same as mastering the ICTs, rather it is the art of using them in a purposeful, healthy way.
By Carlo Gabriel Porto Bellini

Research Highlights

- 94 **Technical Perspective
The Rewards of Selfish Mining**
By Sharon Goldberg and Ethan Heilman
- 95 **Majority Is Not Enough:
Bitcoin Mining Is Vulnerable**
By Ittay Eyal and Emin Gün Sirer



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.

Acting Executive Director
Deputy Executive Director and COO
 Patricia Ryan
Director, Office of Information Systems
 Wayne Graves
Director, Office of Financial Services
 Darren Ramdin
Director, Office of SIG Services
 Donna Cappel
Director, Office of Publications
 Scott E. Delman

ACM COUNCIL

President
 Cherri M. Pancake
Vice-President
 Elizabeth Churchill
Secretary/Treasurer
 Yannis Ioannidis
Past President
 Vicki L. Hanson
Chair, SGB Board
 Jeanna Matthews
Co-Chairs, Publications Board
 Jack Davidson and Joseph Konstan
Members-at-Large
 Gabriele Anderst-Kotis; Susan Dumais;
 Claudia Bauzer Medeiros; Elizabeth D. Mynatt;
 Pamela Samuelson; Theo Schlossnagle;
 Eugene H. Spafford
SGB Council Representatives
 Paul Beame; Jenna Neefe Matthews;
 Barbara Boucher Owens

BOARD CHAIRS

Education Board
 Mehran Sahami and Jane Chu Prey
Practitioners Board
 Terry Coatta and Stephen Ibaraki

REGIONAL COUNCIL CHAIRS

ACM Europe Council
 Chris Hankin
ACM India Council
 Madhavan Mukund
ACM China Council
 Yunhao Liu

PUBLICATIONS BOARD

Co-Chairs
 Jack Davidson; Joseph Konstan
Board Members
 Phoebe Ayers; Anne Condon; Nikil Dutt;
 Roch Guerrin; Chris Hankin;
 Yannis Ioannidis; Xiang-Yang Li;
 Sue Moon; Michael L. Nelson;
 Sharon Oviatt; Eugene H. Spafford;
 Stephen N. Spencer; Alex Wade;
 Julie R. Williamson

ACM U.S. Public Policy Office

Adam Eisgrau,
 Director of Global Policy and Public Affairs
 1701 Pennsylvania Ave NW, Suite 300,
 Washington, DC 20006 USA
 T (202) 659-9711; F (202) 667-1066

Computer Science Teachers Association
 Jake Baskin
 Executive Director

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

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

STAFF

DIRECTOR OF PUBLICATIONS

Scott E. Delman
 cacm-publisher@cacm.acm.org

Executive Editor

Diane Crawford

Managing Editor

Thomas E. Lambert

Senior Editor

Andrew Rosenbloom

Senior Editor/News

Lawrence M. Fisher

Web Editor

David Roman

Rights and Permissions

Deborah Cotton

Editorial Assistant

Jade Morris

Art Director

Andrij Borys

Associate Art Director

Margaret Gray

Assistant Art Director

Mia Angelica Balaquiot

Production Manager

Bernadette Shade

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 acmm mediasales@acm.org

Association for Computing Machinery (ACM)

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

EDITORIAL BOARD

EDITOR-IN-CHIEF

Andrew A. Chien
 eic@cacm.acm.org

Deputy to the Editor-in-Chief

Lihan Chen
 cacm.deputy.to.eic@gmail.com

SENIOR EDITOR

Moshe Y. Vardi

NEWS

Co-Chairs

William Pulleyblank and Marc Snir

Board Members

Monica Divitini; Mei Kobayashi;
 Michael Mitzenmacher; Rajeev Rastogi;
 François Sillion

VIEWPOINTS

Co-Chairs

Tim Finin; Susanne E. Hambrusch;
 John Leslie King; Paul Rosenbloom

Board Members

Stefan Bechtold; Michael L. Best; Judith Bishop;
 Andrew W. Cross; Mark Guzdial; Haym B. Hirsch;
 Richard Ladner; Carl Landwehr; Beng Chin Ooi;
 Francesca Rossi; 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;
 Terry Coatta; Stuart Feldman; Nicole Forsgren;
 Camille Fournier; Jessie Frazelle;
 Benjamin Fried; Tom Killalea; Tom Limoncelli;
 Kate Matsudaira; Marshall Kirk McKusick;
 Erik Meijer; George Neville-Neil;
 Jim Waldo; Meredith Whittaker

CONTRIBUTED ARTICLES

Co-Chairs

James Larus and Gail Murphy

Board Members

William Aiello; Robert Austin; Kim Bruce;
 Alan Bundy; Peter Buneman; Carl Gutwin;
 Yannis Ioannidis; Gal A. Kaminka;
 Ashish Kapoor; Kristin Lauter; Igor Markov;
 Bernhard Nebel; Lionel M. Ni; Adrian Perrig;
 Marie-Christine Rousset; Krishan Sabnani;
 m.c. schraefel; Ron Shamir; Alex Smola;
 Josep Torrellas; Sebastian Uchitel;
 Hannes Werthner; Reinhard Wilhelm

RESEARCH HIGHLIGHTS

Co-Chairs

Azer Bestavros and Shriram Krishnamurthi

Board Members

Martin Abadi; Amr El Abbadi; Sanjeev Arora;
 Michael Backes; Maria-Florina Balcan;
 Andrei Broder; David Brooks; Doug Burger;
 Stuart K. Card; Jeff Chase; Jon Crowcroft;
 Alexei Efros; Bryan Ford; Alon Halevy;
 Gernot Heiser; Takeo Igarashi; Sven Koenig;
 Steve Marschner; Greg Morrisett;
 Tim Roughgarden; Guy Steele, Jr.;
 Robert Williamson; Margaret H. Wright;
 Nikolai Zeldovich; Andreas Zeller

SPECIAL SECTIONS

Co-Chair

Sriram Rajamani

Board Members

Tao Xie; Kenjiro Taura; David Padua

ACM Copyright Notice

Copyright © 2018 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:10.1145/3226066

Vicki L. Hanson

Reflections on My Two Years

AS MY TERM as ACM President draws to an end, I have been reflecting on my time serving our organization. It is easy to get lost in the day-to-day demands of this position, losing sight of what we have been able to accomplish during the past two years. We have accomplished a lot.

A clear highlight for me and for many was the 50th Turing celebration last year. Honoring our Turing Laureates and bringing together so many who have profoundly shaped our field, the event featured panels on key topics in computing, all of which can be viewed at <https://www.acm.org/turing-award-50/video>. There are many special moments to enjoy; moments that still make me smile when I think back.

Another highlight for me was the greater prominence of our early-to-mid-career technical award, renamed the ACM Prize in Computing. Reflecting its stature, ACM Prize recipients now join our Turing Laureates, along with the Fields Medalists, Abel Laureates, and Nevanlinna Prize recipients at the annual Heidelberg Laureate Forum (<http://www.heidelberg-laureate-forum.org/>) for a week of mentoring and technical exchanges with 200 early-career researchers.

As many of you know, I have been particularly focused on the need to nurture and learn from the next generation of computing professionals—those who will emerge as the leaders of ACM and our profession in the years ahead. Just over a year ago, the ACM Future of Computing Academy was formed (<http://www.acm-fca.org/>), with the goal of empowering the next generation of computing researchers and practitioners. Contributions from this group have already made a mark on ACM, influencing our publication and education initiatives and shaping our

thinking on how we can play a leading role in solving today's and tomorrow's most pressing problems.

During my term we also have been buffeted by a number of unforeseen political, social, and technical policy challenges. These included proposed restrictions on free and open scientific exchange and seemingly unending revelations of powerful leaders—in media, politics, business, and, yes, even technology—behaving in completely unacceptable ways. ACM, like all professional organizations, had the choice to either run from these issues or take strong positions consistent with our character. I am proud that in response we have issued our Open Conference policy supporting global scientific exchange, our policy against harassment at ACM activities, and our U.S. and European policy guidelines on Algorithmic Transparency and Accountability.

My candidate statement two years ago outlined a pressing need for rapidly identifying and supporting emerging technical research areas. ACM's reputation for technical quality remains without peer, but that does not mean we can rest on our laurels. To get a handle on this we have surveyed our technical leaders and conducted targeted workshops around the world to identify emerging technical trends. This large body of input is currently being summarized but I note here that in addition to new technical directions in computing, more interdisciplinary work and a strong focus on the ethics of technology development and use were recurring themes.

Approximately half of our global members are practitioners. ACM has identified ways in which we already provide significant value along with ways in which we could do more. The ACM Practitioner Board has been doing a great job driving multiple activities, including a steady stream of high-quality and highly attended technical webi-

nars, additions to ACM's Distinguished Speaker series to reflect more practitioner interests, and using meetups and theme packs to provide information on topics such as blockchain and AI/ML.

Those who know me know my commitment to diversity and inclusion. My appointments to boards and committees have sought to reflect regional and gender diversity. The launch of a task force on Diversity and Inclusion will shed additional light on ways ACM can promote inclusive best practices in all we do. I have also worked to forge links with multiple national computing societies. ACM's sponsorship of the United Nations' initiative on *AI for Good* (<https://www.itu.int/en/ITU-T/AI/2018/Pages/default.aspx>) has expanded our influence within a multi-cultural, interdisciplinary community working to harness technology to address critical world problems.

As my term ends, I am pleased to say that ACM remains strong financially with talented volunteers and headquarters staff committed to serving the community. I have been continually impressed with the variety and depth of new initiatives created by our SIGs, board, and committees.

Finally, while it is gratifying to see the progress I mention, I am the first to admit that our work on these efforts is far from finished. I see ACM as a continual 'work in progress' as computing and our community evolves. In the future, I believe ACM will need to constantly reassess priorities and activities to ensure we remain vital to the global computing community. I encourage each of you to keep doing what you do so well, keeping ACM the premier global computing society. ■

Vicki L. Hanson (vlh@acm.org) is a Distinguished Professor in the B. Thomas Golisano College of Computing, Rochester Institute of Technology, Rochester, NY, USA.

Copyright held by author.



Google Cloud

A cloud made for L.A.



Latency goes down. Availability goes up.
Production can stay on schedule.
A new Google Cloud region is coming to L.A.

cloud.google.com/la



Vinton G. Cerf

DOI:10.1145/3224195

On Neural Networks

I am only a layman in the neural network space so the ideas and opinions in this column are sure to be refined by comments from more knowledgeable readers.

The recent successes of multilayer neural networks have made headlines. Much earlier work on what I imagine to be single-layer networks proved to have limitations. Indeed, the famous book, *Perceptrons*,^a by Turing laureate Marvin Minsky and his colleague Seymour Papert put the kibosh (that's a technical term) on further research in this space for some time. Among the most visible signs of advancement in this arena is the success of the DeepMind AlphaGo multilayer neural network that beat the international grand Go champion, Lee Sedol, four games out of five in March 2016 in Seoul.^b This was followed by a further success against Ke Jie winning three games of three in Wuzhen in May 2017.^c Further developments have led to AlphaGo Zero^d that learned to play championship Go in only 40 days with only the rules of Go and a reinforcement learning algorithm to generate the neural network weightings. AlphaGo Zero learned to play chess well enough to beat most (maybe all?) computer-based players in 24 hours.^e

These developments are dependent in part on vastly faster and cheaper computing engines such as graphical processing units and, at Google, tensor processing units that run multilayer ma-

chine learning algorithms swiftly and increasingly efficiently. Google used a version of its tensor processing system, TensorFlow, to reduce the cost of cooling a datacenter by 40%.^f Many other applications are surfacing for these neural network systems, including reliable identification of diseases (for example, diabetic retinopathy, cancerous cells), situational awareness for self-driving cars, and a raft of other hard recognition and optimization problems.

Speculation is rampant as to where these methods may take us in the future. Powerful decision-making tools are in development or already in operation, absorbing and applying large amounts of data to discrimination tasks normally reserved for human judgment. Therein lies an issue deserving of the attention of ACM members and all others engaged in fashioning these new tools. These systems are brittle in some respects. The training sets used to develop the neural network weights can be biased in ways not known to the trainers. The choices or decisions indicated by the networks can also exhibit chaotic effects in the sense that small changes in inputs can result in extreme changes in output. In so-called generative adversarial networks,^g two networks are pitted against each other (see Goodfellow et al. p. 56). One tries to fool the other into thinking that an image of a dog, for example, is actually a cat. Changes to small numbers of

pixels that are imperceptible to humans can cause a neural network to make major classification mistakes. In my layman's cartoon model of this phenomenon, I imagine each pixel in the input image is a distinct dimension in a high dimension space. Hyperplanes separate images of animals, for instance, from each other. Small changes in the values of some pixels may cause a vector in hyperspace to move across the hyperplane boundary and cause the system to misidentify an image of a dog as a cat or something else.

These hazards drive the need for serious thinking about the potential to depend too much on the output of such neural systems or to make autonomous decisions without human intervention. Attention to the training sets and assiduous testing against a wide range of inputs seems essential to limit serious side effects of decision making using these systems. That such neural networks can be extremely valuable is inarguable in the face of demonstrated results so far. That they can also be catastrophically wrong is equally evident, triggering a serious need for ethical considerations in their development and application. This is true in general for all software, but especially so for neural networks which functions are still somewhat mysterious and which successes are so spectacular that it is tempting to ignore the potential for unintended consequences of their use. 

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

Copyright held by author.

a <https://mitpress.mit.edu/books/perceptrons>

b https://en.wikipedia.org/wiki/AlphaGo_versus_Lee_Sedol

c https://en.wikipedia.org/wiki/AlphaGo_versus_Ke_Jie

d <https://deepmind.com/blog/alphago-zero-learning-scratch/>

e <https://en.chessbase.com/post/the-future-is-here-alphazero-learns-chess>

f <https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/>

g <https://deeplearning4j.org/generative-adversarial-network>



@ACMIUI



ACM IUI



SIGCHI



ACM IUI 2019

Intelligent User Interfaces

March 17-20 2019

Marriott Marina Del Rey
Los Angeles, CA

<http://iui.acm.org/2019/>

Where HCI meets A.I.

Important Dates

REGULAR PAPERS

Abstract Deadline Oct 1 2018

Papers Deadline Oct 8 2018

Notification Dec 7 2018

✉ program-iui2019@acm.org

WORKSHOPS

Proposals Due August 14 2018

Decisions Sent September 14 2018

✉ workshop-iui2019@acm.org

DEMOS&POSTERS

Due in December 2018

✉ poster-iui2019@acm.org

STUDENT CONSORTIUM

Due November 9 2018

✉ sc-iui2019@acm.org



CS@ILLINOIS
COMPUTER SCIENCE





Moshe Y. Vardi

DOI:10.1145/3226073

How the Hippies Destroyed the Internet

WHEN WE REFER to “the Internet” we refer not only to the global system of interconnected computer networks but also to the set of applications that utilize this network, including email, the Web, search engines, social media, and the like. To understand where this Internet comes from, we have to revisit the emergence of online communities in the early and mid-1980s. Consider, for example, the WELL, which began in 1985 as a dial-up bulletin board system, self-described as “a cherished watering hole for articulate and playful thinkers.” One of its founders was Stewart Brand, best known as editor of the *Whole Earth Catalog*, an American counterculture magazine and product catalog published periodically since the late 1960s. “Counterculture” refers to a late-1960s–early-1970s Western antiestablishment cultural movement, whose members were known as “hippies.” Today’s Internet, with its techno-utopian culture,^a connects with the 1960s counterculture movement.

In a 1984 Hackers’ Conference, Brand told Steve Wozniak, a founder of Apple Inc., “Information wants to be free, because the cost of getting it out is getting lower and lower all the time.” This phrase, “Information wants to be free,” came to mean people should be able to access information freely. It has become an ideology of many technology activists who criticize any restriction to open and free access to information. Completely forgotten today is the fact that this phrase is taken out of context; the preceding sentence by Brand was “Information wants to be expensive, because it’s so valuable.” Of course, infor-

mation does not want anything. It is people who want information to be free, but “Information wants to be free” meshed well with the antiestablishment character of the techno-utopianism. When the Internet and the World-Wide Web grew explosively in the early 1990s, information freedom became a mantra.

But information freedom meant that rather than creating an information market, information has become “commons,” an unregulated shared public resource, which, as popularized by Garrett Hardin in an influential 1968 article, is subject to “The Tragedy of the Commons.” This phrase refers to the phenomenon where individual users acting independently according to their own self-interest behave contrary to the common good. Of course, we all love free information. The question is whether information freedom is good for society.

But markets in which the prices for goods and services are determined by sellers and buyers are among the greatest inventions of human civilization, though arguments continue about the relative advantages and disadvantages of free markets, coordinated markets, regulated markets, and the like. Regardless of the details, markets provide us with a mechanism for finding the value of goods and services. Communism is the most famous 20th-century attempt to build market-free economies. It required coercion on a colossal scale, with an incalculable cost in human lives. The Internet is the second major attempt to build a market-free economy, limited to information. When search and social emerged as business in the late 1990s and mid-2000s, respectively, information freedom was already a hallowed Internet principle, and companies have adapted to it by making themselves into

advertising companies. Ethan Zuckerman, director of the MIT Center for Civic Media, called information freedom “The original sin of the Internet.”^b

Why is information freedom such a horrible mistake? To start with, information freedom is, of course, an illusion. Google and Facebook are stupendously profitable companies. Where do these profits come from? “Not from me,” you may say, “the advertisers pay to advertise.” But advertising is just the cost of doing business, and advertisers simply pass the cost of advertising to the price of the goods and services they provide. Thus, instead of having a transparent market in which posted prices lead to value discovery, we have an opaque market in which consumers support Internet companies via, essentially, an invisible tax.

But market opaqueness is just one problem. As we now know, Internet advertisers require data to ensure effective delivery of ads, so we not only pay for “free information” with invisible tax, we pay for it also with our personal information. The Internet has become a huge surveillance machine.

In a recent *New York Magazine* article, the “Internet apologized.”^c The article contains a breakdown of what went wrong with the Internet from the architects who built it. It is worth reading. But the real question is whether it is not too late to ditch the ad-based business model and build a better Internet. This is, I believe, one of the most important questions in computing right now! ☐

^b <https://theatlantic.com/2wU5Xsa>

^c <https://slet.al/2GZ6hGr>

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.

^a <https://bit.ly/2GyQ0XP>

Teach the Law (and the AI) ‘Foreseeability’

RYAN CALO’S “Law and Technology” Viewpoint “Is the Law Ready for Driverless Cars?” (May 2018) explored the implications, as Calo said, of “...genuinely unforeseeable categories of harm” in potential liability cases where death or injury is caused by a driverless car. He argued that common law would take care of most other legal issues involving artificial intelligence in driverless cars, apart from such “foreseeability.”

Calo also said the courts have worked out problems like AI before and seemed confident that AI foreseeability will eventually be accommodated. One can agree with this overall judgment but question the time horizon. AI may be quite different from anything the courts have seen or judged before for many reasons, as the technology is indeed designed to someday make its own decisions. After the fact, it may be impossible to ascertain the reasons for or logic behind its decisions.

AI is a sort of idiot savant that can be unpredictably, and potentially, dangerously literal. Calo gave an example of a driverless car instructed to maximize efficiency and decide that having a fully charged battery would be the best way to achieve it. The car kept its engine running in the garage of a house overnight and, in doing so, asphyxiated its human occupants. This is an example of the so-called “paper-clip problem,” whereby an AI is programmed with its sole objective to make paper clips. When it runs out of metal wire, it begins to make them out of anything else it can find. Recall how the HAL 9000 computer in Stanley Kubrick’s and Arthur C. Clarke’s *2001: A Space Odyssey* let nothing interfere with its mission objective, including, tragically, its human astronauts.

AI software designers are still so new at developing AI it will be difficult for them to predict what could happen as it is deployed in the real world. Manufacturers and designers using AI compete in an environment where market share and profitability almost always drive product development and release, more

than any study of potential outcomes. MIT physics professor Max Tegmark has insightfully explored such “bugs” in the application of current technology.¹

As liability cases are litigated, courts in different jurisdictions, following a similar set of facts and circumstances, may produce very different judgments. If the manufacturer claims particular AI software is proprietary, determining what led the software to make a particular decision might be futile.

AI is a field of information technology the average person, including owners of AI-equipped cars and members of a jury, can barely grasp, much less evaluate. Further study of foreseeability could only benefit the technology, as well as the law.

Reference

1. Tegmark, M. The near future: Breakthroughs, bugs, laws, weapons, and jobs. Chapter 3 in *Life 3.0: Being Human in the Age of Artificial Intelligence*. Alfred A. Knopf, New York, 2017, 93–110.

Evelyn McDonald, Fernandina Beach, FL, USA

Author Responds:

I appreciate this thoughtful response. The paper-clip problem has always fascinated me when offered as evidence of the supposed existential threat AI poses to humanity. The problem envisions a system so limited that it blindly follows a single objective function—making paper clips—but is simultaneously so powerful, intelligent, and versatile that it overcomes the sum of human resistance. Regardless, I completely agree with McDonald’s central takeaway that we cannot know how AI will be deployed in practice in the years to come.

Ryan Calo, Seattle, WA, USA

Political Correctness, Here, Too

I sympathize with Bob Toxen’s position, as outlined in his letter to the editor “Get ACM (and *Communications*) Out of Politics” (May 2018). Meanwhile, writing as if to lend additional support to Toxen’s critique, Moshe Y. Vardi wrote in his Vardi’s Insight column “How We Lost the Women in Computing” (also in May 2018) that women were being pushed

out of computing. I found this claim too harsh and also unjust. To me, it involves intention. And Vardi’s argumentation looks to me more political, or politically correct, than scientific. Of course we need more women in computing, and yes, one can be biased without recognizing one’s own bias. Still, my personal experience, on hiring committees at the University of Michigan and at Microsoft, is that computer scientists try very hard to bring women on board.

There are interesting parallels between U.S. and Soviet political correctness. In the late 1960s, I was the chair of the mathematics department in Sverdlovsk Institute for National Economy—a Soviet university—responsible for the entrance exams in mathematics. The rector of the university pressed for increasing the percentage of accepted students from the working classes, as opposed to the intelligentsia. In principle I liked the idea. My parents were laborers. The question was how to achieve the goal. I suggested a division of labor: We, the mathematicians, would grade the exams on merit, as usual, and the administration would accept whomever it deemed appropriate. I also suggested that we offer remedial courses for working-class high-school students to prepare them for the rigors of university-level mathematics. But the rector would have none of it. He wanted us to grade on merit and somehow simultaneously increase the percentage of working-class students. The pressure came from above. Higher authorities wanted to increase the percentage of working-class students. But even in the USSR, nobody accused us of pushing out the group of people in question.

The issue of this letter is bigger than gender equality or the Soviet experience. It is about political correctness. Responding to Toxen, Vardi wrote: “*Communications* is definitely not only about computers and programming.” I still like Toxen’s idea of taking *Communications* out of politics. But if we have to debate a political issue, it should be done constructively, without exaggerating or imputing intentions that people may not have.

Yuri Gurevich, Ann Arbor, MI, USA

Editor-In-Chief Responds:

It is good to see a debate has broken out in Communications. Issues concerning the health and inclusiveness of the computing professional community must be at the heart of what ACM does and is concerned about. I find the topic entirely appropriate and welcome thoughtful perspectives representing different points of view. In the larger scope, it is clear that computing is far too important to science, education, commerce, society, and government for Communications to take a narrow view. It must engage the multiplying issues of how computing is transforming the world, for the good, and, yes, sometimes for the worse, as well as how the world shapes computing.

Andrew A. Chien, Editor-In-Chief,
Communications of the ACM

Author Responds:

Gurevich argues that to justify the claim that women have been pushed out of computing I had to bring evidence of intention. But my claim was about action, not intention, and I did bring evidence for that, as much as possible within the confine of a one-page column. Gurevich also insinuates that women have less talent in computing than men. Factual evidence would show this insinuation to be false.

Moshe Y. Vardi, Houston, TX, USA

De-Identify My Patient Data or You Can't Have It

Computational de-identification techniques, many with near-perfect performance, increase patient privacy and reduce the potential for abuse of patient data, even in the most complex security situations (such as with hospital discharge summaries).² Although they perform well with patient data, applying them in real-life hospital and insurance systems remains a challenge. Samuel Greengard's news story "Finding a Healthier Approach to Managing Medical Data" (May 2018) emphasized the importance of maintaining patient privacy and deserves credit for bringing it to the attention of *Communications* readers, as well as to public-health researchers.

But could it be that the most advanced methods of protecting patient data are not actually as effective as Greengard seemed to assume? Imagine two repositories of patient data, as one would find in most hospital and insurance systems today. The first is the original and the

second a highly protected de-identified representation of the original. If a corrupt employee or malicious hacker wanted to access and redistribute or sell patient data, which source would they be more likely to target? The hacker would obviously go for the raw data of the first source, which is already fully accessible to a large number of hospital personnel, including nurses, doctors, and technical staff. Moreover, detailed patient medical claims for almost everyone are already available to insurance companies. Included are personal and demographic details, as well as clinical information like procedures, co-morbidities, and laboratories. A person with malicious intent could thus use the simplest technologies (such as a mobile hard drive or an email or FTP server) to transfer sensitive data of potentially hundreds of millions of patients from secure servers to unsecured laptops.

Public-health researchers and other data scientists often use de-identified patient data rather than the original patient data, as Greengard discussed, limiting their ability to produce credible studies; for instance, removing specific dates of death from a patient record would preclude their ability to study liver complications, as the standard outcome in this domain is the patient's short-term mortality.¹

As Greengard suggested regarding the benefit to public health from analyzing such data, lawmakers should look to impose significant penalties on anyone convicted of abusing data, rather than require medical professionals to de-identify it, as is currently the case. Data scientists would be better off if the data is left raw (legally) in its most unadulterated form rather than be de-identified, achieving improved accuracy of scientific findings that could directly improve patient care.

References

1. Kartoun, U., Corey, K., Simon, T., Zheng, H., Aggarwal, R., Ng, K., and Shaw, S. The MELD-Plus: A generalizable prediction risk score in cirrhosis. *PLOS ONE* 12, 10 (Oct. 2017). <http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0186301>
2. Uzuner, O., Luo, Y., and Szolovits, P. Evaluating the state of the art in automatic de-identification. *Journal of the American Medical Informatics Association* 14, 5 (June 2007), 550–563.

Uri Kartoun, Cambridge, MA, 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.

© 2018 ACM 0001-0782/18/7 \$15.00

Results of ACM's 2018 General Election

President:

Cherri M. Pancake
(July 1, 2018–June 30, 2020)

Vice President:

Elizabeth Churchill
(July 1, 2018–June 30, 2020)

Secretary/Treasurer:

Yannis Ioannidis
(July 1, 2018–June 30, 2020)

Members at Large:

Claudia Bauzer Mederios
(July 1, 2018–June 30, 2022)

Theo Schlossnagle
(July 1, 2018–June 30, 2022)

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:10.1145/3213760

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

We Are Done with 'Hacking'

Today's programmers offer more valuable skills than simply being able to hack algorithms and make data structures, says Yegor Bugayenko.



Yegor Bugayenko The Era of Hackers Is Over

<http://bit.ly/2JH8qrg>
April 23, 2018

In the 1970s, when Microsoft and Apple were founded, programming was an art only a limited group of dedicated enthusiasts actually knew how to perform properly. CPUs were rather slow, personal computers had a very limited amount of memory, and monitors were lo-res. To create something decent, a programmer had to fight against actual hardware limitations.

In order to win in this war, programmers had to be both trained and talented in computer science, a science that was at that time mostly about algorithms and data structures. The first three volumes of the famous book *The Art of Computer Programming* by Donald Knuth, a Stanford University professor and a Turing Award recipient, were published in 1968–1973. This programming bible earned a famous comment from Bill Gates: “It took incredible discipline, and several months, for me to read it.”

This comment demonstrates the reality that creating a simple piece of software was a complex engineering task, even though software engineering only “emerged as a discipline in its own right” later on in the 1980s, as Ian Sommerville said in his 1982 book *Software Engineering* (<https://dl.acm.org/citation.cfm?id=1841764>).

Most programmers were calling themselves “hackers,” even though in the early 1980s this word, according to Steven Levy’s book *Hackers: Heroes of the Computer Revolution*, “had acquired a specific and negative connotation.” Since the 1990s, this label has become “a shibboleth that identifies one as a member of the tribe,” as linguist Geoff Nunberg pointed out (<https://n.pr/2rg8g2v>).

Being able to hack both the software and the hardware was a virtue for a long time. However, the world of computer programming has changed dramatically in the last decade.

First of all, the cost of computing power gets cheaper every year. For example, one gigabyte of computer memory cost about \$1,000 in 2000; in 2018, it costs less than \$5. That

is 200 times cheaper in the span of only 18 years. The same is true for hard drives, monitors, CPUs, and all other hardware resources. As James Somers noticed in his analysis of industry problems in *The Coming Software Apocalypse* (<https://theatlantic.com/2HFazak>): “Computers [have] doubled in power every 18 months for the last 40 years.”

Second, the growth of open source is massive. The majority of software is available for free now along with its source code, including operating systems, graphics processors, compilers, editors, frameworks, cryptography tools, and whatever else we can imagine. Programmers do not need to write much code anymore; all they need to do in most cases is wire together already available components.

Third, despite an increasing population of programmers in the world, the field is still in a deficit. In some European countries, the demand for highly skilled IT personnel is twice as high as their market supply of talent. According to [Iamexpat.nl](http://iamexpat.nl), in the Netherlands “a whopping 76% of HR employees reported hav-

ing difficulty finding enough candidates with this qualification.”

Fourth, programmers now work remotely instead of in offices or cubicles. Thanks to the growth of high-speed Internet, conferencing software like Zoom and Skype, messaging tools like Slack and Telegram, and distributed repository managers like GitHub and Bitbucket, along with many other innovations, remote work has become more comfortable than the alternative of working in the traditional office setting.

Finally, programmer salaries have skyrocketed in the last few decades. In 2000, when one gigabyte of memory still cost \$1,000, the average senior programmer earned around \$80,000 in Silicon Valley. In 2018, they are currently making three times more, while RAM is 200 times cheaper.

Taking these five variables into account, it would appear that the skills required of professional and successful programmers are drastically different from the ones needed back in the 1990s. The profession now requires less mathematics and algorithms and instead emphasizes more skills under the umbrella term “sociotech.” Susan Long illustrates in her book *Socioanalytic Methods: Discovering the Hidden in Organizations and Social Systems* (<http://bit.ly/2w0sFhS>) that the term “sociotechnical systems” was coined by Eric Trist et al. in the World War II era based on their work with English coal miners at the Tavistock Institute in London. The term now seems more suitable to the new skills and techniques modern programmers need.

They need to know how to communicate with the open source community to find the needed components, to request features, and to learn bug fixes from their developers. Moreover, they have to be ready to contribute to open source software by submitting pull requests or even creating their own programs. Those who used to work only with commercial and private software will soon be far behind other programmers.

They have to know how to get help outside of an office or even a project team when working remotely and alone. Aside from Stack Overflow, which dominates the Q&A platform for

programming market, there are documentation and code repositories that a professional programmer must know how to navigate. Those who previously only relied on colleagues and friends will now lose to those who know how to learn from the entire Internet.

Programmers have to know how to write maintainable code that other programmers will be able to easily understand. Since hiring personnel grows more expensive every year, businesses emphasize the maintainability of their code bases over developing exceptionally complex code. It is easier for them to buy a larger server if the algorithm is not fast enough, rather than lose what previous programmers created when a new team or a replacement shows up and fails to understand how to modify the project. Because the cost of computers continues to grow cheaper and the cost for employing programmers continues to increase, maintainability continues to dominate the programming landscape as the primary virtue of almost any software. The end result is that these “Hackers” who spend their days writing complex, cryptic code will soon find themselves out of the market.

Edsger W. Dijkstra’s words—“Simplicity is a great virtue”—which he uttered in 1984, grow increasingly more valuable every year.

It seems that the future of programming rests less in math and more in sociotech relationships between people.

Comments

Not sure I agree with the viewpoints in this blog. “Hacker” is a casually understood term, it could be simply understood as someone who is extremely sophisticated in writing code, but not necessarily isolated from engaging in sociotech relationships. I am sure that “hackers” do run into mental blocks, and they, too, look for answers at “Stack Overflow” or happily provide some to “show off” their “hacking” skills. “Hacking” doesn’t necessarily make code more complex than necessary, it may do just the opposite — making code simpler than necessary that affects readability. “Hacking” might also mean the “code-and-fix” practice, which the article is right about. But the verdict is just as old

as the term “software engineering”—we need software craftsmanship, if not an engineering process, to build software. “Hackers” are always there regardless of how we define them, and I am sure that their productivity can be turned into good use.

Does the future of programming rest less in math? Coding more efficient algorithms (regardless of RAM price dropping) or writing more efficient code requires mathematical thinking. Had our developers known Z-Specification (a formal specification method based on Set Theory), they would have written much more reliable software than we have experienced. The future of programming may be dominated by data-analytics, machine-learning and deep-learning applications. Developers are not just to collect data and hit a button to invoke “shop-provided” models; rather, they are to tweak the models and, by all means, develop new models to be trained by datasets unique to the local business environment. To the opposite, the future of programming rests more in mathematics. After all, computing, by its very nature, is a mathematical discipline.

—Chenglie Hu

I, too, would have to disagree. The hardware of computers today has changed so drastically from those early days that to really write proficient/optimized code, a software engineer has to really understand the hardware and what it is doing ... so “Hackers” live, especially in the fields where speed and efficiency are very important. One has to understand memory, caches, cores and hyper-threading, throughput and latency, vector registers, and how to write code to properly take full advantage.

In order to get that mathematics you write of to run efficiently, one has to know how the computer handles the calculations, fetches, executes, and stores the results the best and fastest way.

Today more than ever, software engineers need to be “Hackers” in the manner that Steven Levy referred to them.

—Rod Haxton

Yegor Bugayenko is founder and CEO of software engineering and management platform Zerocracy.

© 2018 ACM 0001-0782/18/7 \$15.00



The 16th International Conference on **EMBEDDED WIRELESS SYSTEMS AND NETWORKS (EWSN)**

FEBRUARY 25-27, 2019

BEIJING, CHINA

EWSN is a highly selective single-track international conference focusing on the latest research results on embedded systems and wireless networking. This year, EWSN will go outside Europe for the first time, and be held in Beijing, China.

The conference proceedings will appear in the ACM Digital Library. Selected accepted papers will be fast-tracked to ACM Trans. on Sensor Networks (TOSN) or ACM Trans. on the Internet of Things (TIOT).

ORGANIZERS

GENERAL Co-Chairs

Yunhao Liu, *Tsinghua University, China*

Guoliang Xing, *Michigan State University, USA*

TPC Co-Chairs

Gian Pietro Picco, *University of Trento, Italy*

Yuan He, *Tsinghua University, China*

TOPICS OF INTEREST

- Communication and networking for wireless and embedded systems
- Sensing, actuation, and control
- Operating systems, middleware, and services
- Processing, storage, and management of data
- Programming paradigms, languages, and tools
- Emerging computing platforms, e.g., drones and robots
- Time and location management
- Dependability (real-time, reliability, availability, safety)
- Sustainability (low-power operation, energy management, energy harvesting)
- Privacy and security in applications and systems
- Human-machine interaction with wireless and embedded systems
- Design and implementation of real-world applications and systems

Contributions that reflect emerging technologies and applications related to industrial IoT are particularly welcome.

IMPORTANT DATES

Paper registration: September 7, 2018

Paper submission: September 14, 2018

Author rebuttal deadline: November 16, 2018

Notification: December 1, 2018

<http://ewsn2019.thss.tsinghua.edu.cn/>



Why Cryptocurrencies Use So Much Energy—and What to Do About It

The electricity consumption of mining for cryptocurrencies is becoming a real concern. Here's what to do about it.

IN RECENT MONTHS, bitcoin and other cryptocurrencies have plunged in value, yet the market capitalization for these digital currencies is still valued at hundreds of billions of dollars. That market cap has grown more than 20 times since last year, when the cryptocurrency boom began.

With bitcoin's booming popularity comes problems. Speculation in bitcoin and other cryptocurrencies is rampant. Scams abound, and plenty of initial coin offerings (ICOs) have overpromised or underdelivered spectacularly.

Through it all, the world has focused on how bitcoin and other cryptocurrencies could implode and go to zero—or make you rich—depending on who you ask. Yet another aspect to cryptocurrencies has not received as much attention.

A major issue that results from increased adoption has not been adequately addressed; they use a lot of energy. The “mining” process that creates bitcoin uses more energy than Serbia, says Digiconomist, a self-described “platform that provides in-depth analysis, opinions, and discussions with re-



gard to bitcoin and other cryptocurrencies ... on a voluntary, best-effort basis.”

According to Bitcoinist, another industry site, the average cost in electricity to mine a bitcoin in Serbia is about \$3,100, making it quite profitable to mine the coin there (among other countries with low-cost electricity).

As cryptocurrencies rise in price, the problem isn't going away. Right now, Digiconomist estimates that bitcoin mining, the process of generating bitcoins, accounts for 0.29% of the world's annual electricity consumption. The mining of a single bitcoin block—a block of transaction data on the bitcoin network—consumes enough energy to power more than 28 U.S. homes for a day.

Other cryptocurrencies that are structured similarly to bitcoin use energy for mining, too. Bitcoin is the most popular and best known cryptocurrency, but it is not unique in its energy needs.

Some people wonder if cryptocurrencies will disrupt the financial system, while others wonder if they will break the environment in the process.

Mining for Digital Gold

Many cryptocurrencies, including bitcoin, are “mined” into existence. Mining is when computers solve complex math problems to generate new bitcoins on the bitcoin network. The computers that solve each progressively more complex equation receive a reward in bitcoin.

According to site 99Bitcoins (a source of information on the crypto currency for the non-technical), a “constant amount” of bitcoins is created when a math problem is solved. The number of bitcoins awarded used to be 50 per problem solved, dispersed among all bitcoin miners; however, that number drops by half every 210,000 times an award is given out. In late 2017, that meant 12.5 bitcoins were awarded each time each progressively more difficult math problem was solved.

The bitcoin network, says the site, “is designed to produce a constant amount of bitcoin every 10 minutes.” That means every time a miner joins the network, it will become harder to solve the problem resulting in the reward of bitcoins. The difficulty scales up to ensure bitcoin is generated every 10 minutes, no matter how much pro-

cessing power you throw at it.

It is here that the energy problem arises. Bitcoin uses a “proof-of-work” (PoW) system to mine new bitcoins and verify transactions on the network. PoW means that computers “mining” bitcoin prove the data in each block of bitcoin being mined (the hard math problem to solve).

“The proof-of-work scheme requires guessing the solution to an equation (actually, an inequality),” says David Malone, a lecturer at Ireland's Maynooth University. “The guessing uses lots of computing power and, consequently, electricity.”

When PoW is completed, rewards are paid out in bitcoin. Depending on the price of bitcoin at any given time, you may spend less in electricity costs than you receive in bitcoin, potentially making the venture profitable.

For instance, 99Bitcoins calculates that mining bitcoin for one month using one advanced piece of computer hardware would use 1,375kW of electricity, which it estimates would cost the user \$118.

However, remember the part about the mining math problems getting harder over time? More and more computational firepower is required over time to mine at the same rate in order to keep your profitability stable, at least in terms of the number of bitcoins earned.

To cope, the bitcoin mining community often adopts ASICs, or application-specific integrated circuits. ASICs are circuits configured for a particular use case. Specialized ASICs are more powerful than regular computers at bitcoin mining, giving miners with these ASICs the ability to mine faster.

“Bitcoin's proof-of-work scheme

The mining of a single bitcoin block consumes enough electricity to power more than 28 U.S. homes for a full day.

has proven particularly easy to build custom hardware like ASICs for,” says Malone. This has led to the adoption of custom—and energy-intensive—hardware by those who would mine bitcoins for profit.

This isn't always the case with other cryptocurrencies.

“Some other proof-of-work schemes [known as being ASIC-resistant] are designed to be best calculated by regular computers, so people mining them can use regular computers instead of ASICs,” says Malone.

The result is a vicious cycle, with the potential to consume an increasing amount of electricity.

More and more computing power is needed to mine bitcoin, which requires more and more electricity. ASICs can be used to supercharge your mining, which uses even more electricity, and if bitcoin's price rises, it becomes even more profitable to mine, which causes more miners to jump into the game. The more miners, the more computing power needed to crack bitcoin's math problems.

And so the cycle begins anew.

“So, while the value of bitcoin is higher than the cost of electricity, we can only expect more people to jump in, increasing the overall energy demands,” says Malone.

How to Go Crypto-Green?

Bitcoin is the most popular cryptocurrency that uses PoW, but it's not the only one. Many cryptos run on various types of PoW schemes. Ethereum, one of the three most popular cryptos, uses a PoW scheme.

Bitcoin alone uses a lot of electricity, but should other PoW cryptos become popular, the problem could get much worse, much faster.

The good news is that the cryptocurrency community is aware of the problem, although possible solutions span the spectrum from theoretical to practical.

“Some systems use a semi-centralized model (like Ripple or Stellar) that are more green, but the trust assumptions are different than a fully decentralized system like bitcoin,” says Joseph Bonneau, a cryptographer and assistant professor of computer science at New York University who used to work at Google.

These systems may circumvent the energy consumption concerns that arise with bitcoin, but may offer something fundamentally different from the value propositions of existing PoW cryptos.

Though cryptos like Ethereum use PoW, says Bonneau, some people might argue for it being a greener option, since Ethereum mining is typically performed on general-purpose graphics processing units (GPUs) that you can find in everyday computers. These are, theoretically, “greener because the hardware could be repurposed for other things if the currency dies out,” Bonneau says. Bitcoin mining’s specialized ASICs, on the other hand, would have zero practical value if bitcoin disappeared tomorrow.

Yet the real problem is the mining process itself, no matter how green it gets. Bitcoin and other PoW mining schemes are incentivized to consume energy.

“Bitcoin is currently valuable, so people want to earn bitcoins,” says Malone. Miners use their computing power to add blocks of transaction data to the bitcoin blockchain; miners that do so are rewarded with more bitcoins.

“The way to earn bitcoins is to take part in adding blocks to the blockchain, as the bitcoin designers decided to reward this activity to incentivize people to maintain the blockchain,” says Bonneau.

He explains that cryptocurrency mining is “difficult by design to ensure that blocks are found at a certain rate, and money is created at a certain rate. If you designed a new chip that was twice as efficient, the puzzles would simply become twice as hard and there would be no benefit.”

Mining, at the end of the day, is the work that ends up consuming most of the energy on any given crypto network.

In the case of bitcoin, says Bonneau, the energy costs are related “almost entirely to mining; that is, to solving computational puzzles. There are other energy costs of the system, like maintaining the system history, broadcasting and verifying new transactions, but those energy costs are trivial compared to the mining.”

Some cryptocurrency developers have tried to circumvent the mining

No matter how green the mining process gets, cryptocurrencies based on proof-of-work schemes are incentivized to consume energy.

process entirely. They use a system called “proof-of-stake.” These cryptocurrencies, which include DASH and PIVX, don’t use PoW at all since it consumes too much energy, says Malone. Instead, users lock up quantities of cryptocurrency for periods of time, which secures the blockchain used by that currency. In return, they receive cryptocurrency rewards, as if they had mined cryptocurrency themselves.

The result is, potentially, a middle path: cryptocurrency projects can still incentivize people to secure their networks, without requiring the energy needs of a small country to do so. **C**

Further Reading

Beigel, O.

Is Bitcoin Mining Profitable in 2018?, *99Bitcoins*, Jan. 2, 2018, <https://99bitcoins.com/bitcoin-mining-profitable-beginners-explanation/>

Bitcoin Energy Consumption Index, *Digiconomist*, <https://digiconomist.net/bitcoin-energy-consumption>

Rodgers, A.

The Hard Math Behind Bitcoin’s Global Warming Problem, *WIRED*, Dec. 15, 2017, <https://www.wired.com/story/bitcoin-global-warming/>

Sompolinsky, Y. and Zohar, A. Bitcoin’s Underlying Incentives *Communications*, March 2018 <https://cacm.acm.org/magazines/2018/3/225472-bitcoins-underlying-incentives/fulltext>

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

© 2018 ACM 0001-0782/18/7 \$15.00

ACM Member News

COMBINING SIMULATIONS WITH REAL-TIME DATA



“My career has been focused on the problem of parallel and distributed execution of simulations,”

says Richard Fujimoto, Regents Professor at the School of Computational Science & Engineering of the Georgia Institute of Technology (Georgia Tech).

Fujimoto earned his master’s degree and doctorate in Computer Science and Electrical Engineering from the University of California, Berkeley. He received two separate bachelor degrees, one in computer science, the other in computer engineering, from the University of Illinois, Urbana-Champaign.

His Ph.D. training began with an emphasis on computer hardware and architecture. During his studies, Fujimoto became interested in creating simulations and modeling. He has been more focused on software methods ever since, particularly on executing event simulations on parallel computers.

Fujimoto worked at the University of Utah for several years before joining Georgia Tech in 1989.

Much of his work now concerns combining simulations with real-time data. He describes using live data streams of traffic conditions to drive simulation models, which then make predictions about future conditions.

One area in which Fujimoto is particularly interested is mobile computing devices, particularly with regard to the amount of energy consumed by simulation computation, which affects battery life. He anticipates putting considerable effort into research on the energy consumption properties of distributed simulation algorithms on mobile devices.

Fujimoto also is passionate about promoting the stature of modeling simulation as a field in its own right, instead of merely as an application area.

—John Delaney

You've Got Mail!

And that's not all. Email is not what it used to be.

THE FIRST NETWORKED electronic mail message was sent by Ray Tomlinson of Bolt Beranek and Newman in 1971. This year, according to market research firm Radicati Group, 3.8 billion email users worldwide will send 281 billion messages every day.

That's 3.2 million messages a second—hotel reservations, meeting notices, greetings from friends, product designs, receipts, flirtations, complaints, requests for help and, of course, spam. You may feel like a substantial number of them end up in your inbox.

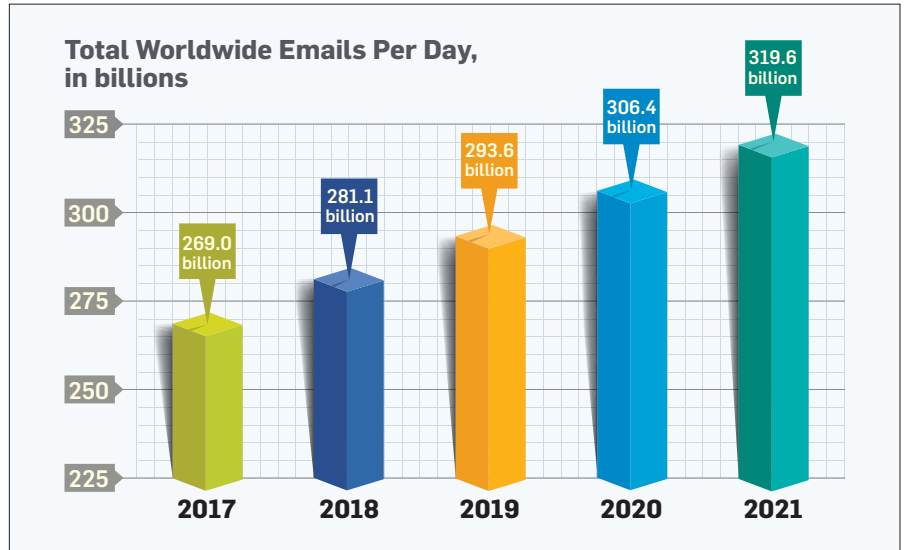
And yet, some observers say email is dying. It's so 'last century', they say, compared to social media messaging, texting, and powerful new collaboration tools. The youngest computer users don't use email much, as you know if you ever tried to email your teenager. When Alice texts Bob about lunch, she doesn't care that her message won't be filed in a "lunch" folder and stored in an archive; she wants something fast, easy, and informal.

Meanwhile, even the most mature users decry the manifold faults of email: its sometimes-cumbersome interfaces, slow response, lack of flexibility, security holes, and spam.

Finally, a host of software entrepreneurs want you to believe their replacements for old-fashioned email are better in every way.

On the other side of the issue stand email power users such as Craig Partridge, chief scientist at Raytheon BBN Technologies (which started out as Bolt Beranek and Newman). "Email is not dying," he says. "It has a core set of functions that no other service has effectively replaced. It gives people tremendous control over how their messages are handled. Users are not locked into a particular user interface, and they know their email will be around, and searchable, as long as they decide to keep it."

Partridge, who was voted into the Internet Hall of Fame last year for his



invention of mail-exchange records for email routing, praises email's use of vendor-independent open standards. "It prevents walled gardens—the inevitable attempt by someone to control your email flow. That's especially important in the corporate world."

Indeed, Radicati Group says reports of the death of email are greatly exaggerated, predicting email message traffic will grow 14% over the next three years, to 319.6 billion messages a day by 2021.

So, is email dying or not? It's a matter of definition. Just what is "email," anyway?

Beyond Messaging

For decades, basic email has been based on open standards within the TCP/IP suite of protocols. These include SMTP, for sending messages between servers, and POP and IMAP for reading or retrieving messages from servers.

Most applications users think of as email today go way beyond those basic message-handling functions. Popular email clients such as Gmail, Outlook, Yahoo, and Thunderbird include powerful and intuitive user interfaces, interfaces to other applications, spam filters, and tools for managing, organizing, archiving, and searching messages.

The newest offerings from start-

ups such as Superhuman, Edison, and Basecamp position their products less as messaging tools than as personal assistants, organizers of the threads of your digital life.

"It's hard to tell if email is on its way to becoming obsolete, or on its way to becoming even more central to how we do things," says Jon Kleinberg, Tisch University Professor of Computer Science at Cornell University and an expert on social and information networks.

As an example, Kleinberg cites TripIt, a smart organizer of travel-related email messages. TripIt will sift through your inbox, harvest travel-related items, update your calendar, and send you master itineraries. TripIt combines rules-based heuristics and artificial intelligence (machine learning and natural language processing) to recognize and make sense of travel information from hotels, car rental companies, travel agents, and the like.

Kleinberg applauds the emergence of such email-based organizing tools, yet he cautions, "With all these artificial assistants, there is this trade-off between the effort they save and the concerns that they will miss something."

Microsoft thinks of its Outlook email product as the basis for personal time management, and in recent years

it has added calendar- and task-management functions to Outlook's basic messaging. The company is taking a "suite approach" to its Office 365 applications, says Gaurav Sareen, Microsoft corporate vice president for Outlook, Yammer, and Groups. Outlook and other Microsoft applications, such as Word and Cortana, share fundamental capabilities and methods, many of them based on artificial intelligence.

For example, Sareen says, machine learning and natural language processing algorithms will soon be able to look at a draft email message and warn the user that a particular sentence is too formal, too informal, or culturally insensitive. This advice is based on three "signals"—knowledge of the user's established habits, awareness of what the user's organization does, and knowledge about the user population at large. In an application of artificial intelligence (AI) using these same signals, Outlook will be able to make shrewd guesses as to which incoming messages merit quick action by the user, a filtering especially useful on small, portable devices. Sareen says AI capabilities like this grow better over time as they increasingly learn about their users and their preferences.

So Many Choices ...

Kleinberg says users choose among a rich variety of messaging applications using two criteria, including the "ephemerality" of the intended message. With email, response times are often measured in days, and users expect that, "But if you get an answer to your text after two days, you might not even know what it's about anymore," he says. It is further assumed that an email message will survive for retrieval months later if needed, but in a couple of hours Bob and Alice won't care about their lunch deliberations.

The second basic discriminator among messaging applications is the strength of the social ties between communicators, Kleinberg says. Users of text messaging and social media-based messaging, for example, usually know each other pretty well, but email users very often do not.

Mapbox, an eight-year-old developer of open-source software tools for location-dependent applications, illustrates the breadth of messaging methods that

"It's hard to tell if email is on its way to becoming obsolete, or on its way to becoming even more central to how we do things."

companies have adopted in recent years. It uses Gmail and voice telephone for communicating with external parties, such as recruits and sales prospects. Internally, when employees want a quick, informal message path, they may use Slack, a tool designed for collaboration in project teams. Mapbox has 500 active, user-defined, subject-specific chat rooms, or channels, on Slack, and in a recent month its 300 employees sent 500,000 messages via Slack.

Julie Munro, manager of customer success at Mapbox, thinks of Slack as an instant-messaging tool for simple, routine communication. "If we feel a conversation in Slack is getting pretty big, and there are some crucial decisions that a couple of people should weigh in on, we'll ticket it and take it to GitHub," she says. GitHub is nominally for software version control and source code management, but it is increasingly used for all kinds of communication among developers.

Mapbox has more than 1,000 subject-specific "repositories" on GitHub, many of them containing code, many others containing message traffic about code, customers, and various subjects of ongoing importance to the company. Ticketed issues are visible to all employees and offer views of company activities and history not easily duplicated by an email system, Munro says. "It's a great way to have open, transparent decision-making," she adds.

... So Little Time

Partridge at BBN Technologies uses Outlook for business mail and Gmail for his personal messages. He processes 300 non-spam email messages a day, and as a touch typist he uses keyboard

shortcuts extensively. He says email interfaces have "gone downhill" for the power user over the past 10 years. "Searching has improved, but dispatching emails mostly requires clicking on things. It takes forever; you click on a message and it wants to pop up a separate window, and it takes a half-second to display. In this day and age, you should never wait for your computer once you have made a decision."

Cornell's Kleinberg, who handles between 100 and 200 messages a day, agrees. "Most of us deal with an extremely high rate of in-bound information," he says. "At some point, the bandwidth constraint becomes human and not technological. Even with all the artificial assistants in the world, there's a point at which it's cognitively hard to keep track of it all." 

Further Reading

Antoun, J.
Electronic Mail Communication Between Physicians and Patients: A Review of Challenges and Opportunities *Family Practice*, Volume 33, Issue 2, 1 April 2016, pages 121-126
<https://academic.oup.com/fampra/article/33/2/121/2404246>

Kleinberg, J.
The Small-World Phenomenon: An Algorithmic Perspective *Proceedings of the 32nd ACM Symposium on Theory of Computing*, 2000 <http://www.cs.cornell.edu/home/kleinber/swn.pdf>

Partridge, C.
The Technical Development of Internet Email *IEEE Annals of the History of Computing*, Volume 30, Number 2, April-June 2008
<https://muse.jhu.edu/article/240040>

Rhotan, J.
Programmer's Guide to Internet Mail: SMTP, POP, IMAP, and LDAP, 1st Edition Elsevier, Nov. 1999, eBook ISBN: 9780080515137
<https://www.elsevier.com/books/programmers-guide-to-internet-mail/rhoton/978-0-08-051513-7>

Takkinen, J. and Shahmehri, N.
Are You Busy, Cool, or Just Curious? – CAFE: A Model with Three Different States of Mind for a User to Manage Information in Electronic Mail *Human IT: Journal for Information Technology Studies as a Human Science*, vol. 2, no. 2, 1998
University of Borås (Sweden)
<https://humanit.hb.se/article/view/495>

Gary Anthes is a technology writer and editor based in Arlington, VA, USA.

© 2018 ACM 0001-0782/18/7 \$15.00

Bringing the Internet to the (Developing) World

A growing number of low-cost (and free!) solutions aim to open the Internet to developing regions.

FOR PEOPLE LIVING and working in North America, Europe, and other relatively prosperous regions of the world, access to the Internet is a given. Even if one cannot afford a private, subscription-based fixed or mobile account, Wi-Fi hotspots offering free Internet access are relatively ubiquitous in coffee shops, public libraries, and even in certain mass transit stations, which lets everyone with a smartphone, tablet, or laptop access the Internet.

Yet in the developing regions of the world, which include but are not limited to parts of Africa, Asia, and Latin America, there is relatively sparse infrastructure in place to allow citizens of these areas to access the Internet. Moreover, even when there are connections available, many people in those regions simply cannot afford either the devices required or the account access.

"Worldwide, I think a little bit over half the world's population does not have access to the Internet today," says Rick Wilmer, CEO of Mountain View, CA-based cloud-managed Wi-Fi service provider Mojo Networks. "One of the major reasons why is because of affordability. In a lot of developing countries, you obviously have lower levels of income and lower levels of disposable income, and the amount of money that the general population has available to spend on something like Internet connectivity is very limited."

Mojo Networks is one of several companies and organizations that are trying to bring connectivity to these underserved regions of the world. The company's largest effort is a venture with Reliance Jio, a wireless telecom provider based in India that has more than 100 million subscribers. Mojo Networks is supplying Reliance Jio with Wi-Fi network access points at cost, thereby helping the wireless car-

rier offer Wi-Fi hotspot access to consumers for free or at very low cost.

As of January 2018, Reliance Jio has deployed more than 100,000 base stations in India, with plans to double that number this year. Currently, non-Jio customers can access the Wi-Fi network for free for up to 30 minutes without getting disconnected. If a user is a Jio customer, they can remain connected to the Wi-Fi hotspot beyond that time limit, and any data use will be billed to the customer's data package. This package is still quite affordable in local terms, with the lowest-cost package starting at a cost of about 98 rupees (US \$1.53), which provides 28 days of access and a data cap of 2GB.

Clearly, there is more than altruism at work, when considering bringing Internet connectivity to places in the world where there is little discretionary income. Indeed, companies such as Mojo Networks are seeing a benefit to making access hardware affordable and shifting its business model to offer recurring revenue-based value-added services to local ISPs.

"I can't necessarily say that we've done this exclusively for solving the problem of affordability in the developing world," Wilmer says. "We're selling cloud services, so we really now look like a SaaS (software as a service) company. Our SaaS offering is a suite of applications that live in the cloud that allow a customer to manage their Wi-Fi network; that's how we make our money."

Other tech companies have also launched initiatives that are designed to provide access via hotspots. Facebook has partnered with India telecom giant Bharti Airtel to help it debut 20,000 Internet access points across the country, as part of the social networking company's Express Wi-Fi project to sell inexpensive Internet access in regions where web coverage is limited.

However, expanding the number of hotspots alone will not be enough. The greatest challenge to providing access to rural or underdeveloped regions is a lack of Internet infrastructure, simply because there has not been a concerted effort to bring big data pipes to those regions. Known as Internet backhaul services, these networks connect local access providers to the Internet itself, and usually are built using expensive solutions, such as undersea broadband cables or a network of satellites, in order to handle the huge traffic demands placed on the network. Without a guaranteed network of revenue-generating users, there is no incentive for a telecom company to lay out the capital required to lay cable or launch satellites.

One company that tried to address this issue is MainOne, a provider of telecom services and network solutions for businesses in West Africa, which started with the goal of building an undersea broadband cable stretching 4,350 miles from Portugal to Nigeria, with stops in Ghana. The company began in 2008 by raising money from Nigerian investors, and two years later, was selling access to local Internet service providers (ISPs).

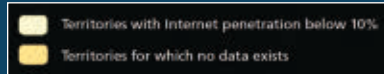
It appears efforts such as these are working. According to a 2018 report from advertising and marketing agency We Are Social and social media management firm Hootsuite, Internet users in Africa are up by more than 20%; the number of people accessing the Internet in Mali, for example, has increased by nearly six times in the past year. Similarly, the number of Internet users in Benin, Sierra Leone, Niger, and Mozambique have more than doubled over the same time frame.

Undersea cables are helpful, but bringing Internet access to landlocked countries or regions requires connections to be brought into these regions, and then supported by local ISPs. Companies such as Liquid Telecom, EveryLayer, and others are working with independent ISPs in cities and villages by helping to negotiate rates for these backhaul services so that they can provide affordable access to consumers.

Wilmer says providing backhaul services to these remote areas is a key challenge. "That's another problem that needs to be overcome."

Google has launched a project to

The Archipelago of Disconnection



use light beams to bring rural areas of the planet online, after it announced a planned rollout in India. Alphabet's X, the new name for the company formerly known as Google X, is working with telecom operator AP State FiberNet in India to utilize Free Space Optical Communications (FSOC), a technology that uses beams of light to deliver high-speed, high-capacity connectivity over long distances. The project will use 2,000 FSOC links to add capacity to its network, which requires connecting 2,000 boxes installed as much as 12 miles apart to create a new backbone to supply service to cellphone towers and Wi-Fi hotspots. Alphabet X's work in India is part of a government initiative to connect 12 million households to the Internet by 2019.

Google also has been trialing an initiative called Project Loon, which involves maintaining a fleet of balloons to provide Internet coverage to users on the ground in remote regions. The system was designed to provide high-speed Internet connectivity by transmitting signals from the ground to a network of balloons in the sky, routing the signal to adjacent balloons on the network and then back down again to users on the ground. However, "the balloons don't seem to have worked out very well," according to Darrell West, vice president of governance studies at the Brookings Institution. "Apparently, there have been problems with maintaining them at a steady height, because local airflows are pushing them up or down and disrupting communications."

West says the use of satellites to bring Internet coverage to wide geographic areas is a more reliable solution. Companies such as OneWeb, which is backed by Virgin founder Richard Branson, and SpaceX, Elon Musk's venture, are placing

satellites in low Earth orbit (100 to 1,250 miles overhead) to provide fast Internet access with low latency rates. Because these lower-latency satellites are not positioned as high in the sky as traditional satellites, more are required to provide adequate coverage of a region, West says.

Quika is one company actively working to bring a free satellite Internet broadband service to underserved geographic regions. Scheduled to launch in the second quarter of this year, the Quika service promised speedy, low-latency Ka-band data connectivity in developing countries, where income inequality and a lack of infrastructure (especially in rural areas) make conventional Internet access impractical for most. Service was slated to begin in Afghanistan and Iraq, and as those services gain traction, additional countries will be supported via local agents in a variety of countries, according to Quika chairman Alan Afrasiab.

"We are aiming to provide citizens of developing countries and those in rural areas trapped by poor and/or expensive Internet infrastructure with a decent, quality connection to help bring about real change to empower communities," Afrasiab says. "Quika Free will target those on low incomes," and services will be available across a range of offerings, from advertising-supported free services, to volume-based services available on gigabyte-per-month plans. Afrasiab says Quika also will target larger commercial organizations. "Quika Plus, VNO, and Trunk will be offered to those who require significant amounts of data or dedicated access to our MPLS network."

It is not just technical or cost issues that can make providing Internet access difficult. Internet access providers face their own regulatory issues,

including the possibility of being required to censor content for local governments. That's why most low-cost Internet services are fairly basic in terms of content, West says. "You get access to job listings, to weather, to news, and entertainment sites," he says, adding that some providers are also offering messaging capabilities.

Overall, the goal of these Internet providers is to give people who have never used the Internet a taste of the features and benefits of Internet access, so that it becomes a "must-have" in their lives, rather than a "nice-to-have."

"The key barrier is the initial deployment," West says. "Once people start using the Internet, they generally want to use it more, and so it becomes a higher financial priority for them. So, once [access is provided], then the market is in a stronger position to take over from there." C

Further Reading

Mumbere, D.

Digital in 2018: Africa's Internet Users Increase by 20%

Africa News.com, Feb. 6, 2016,

<http://www.africanews.com/2016/02/06/digital-in-2018-africa-s-internet-users-increase-by-20-percent/>

Kemp, S.

Digital in 2018: World's Internet Users Pass The 4 Billion Mark

<https://wearesocial.com/blog/2018/01/global-digital-report-2018>

Sokal, E.

Internet access in the developing world

March 15, 2017,

<https://www.youtube.com/watch?v=qN1lZgiHi9o>

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

© 2018 ACM 0001-0782/18/7 \$15.00

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 60 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,



Vicki L. Hanson
President
Association for Computing Machinery



Association for
Computing Machinery

Advancing Computing as a Science & Profession

SHAPE THE FUTURE OF COMPUTING.

JOIN ACM TODAY.

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

SELECT ONE MEMBERSHIP OPTION

ACM PROFESSIONAL MEMBERSHIP:

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

ACM STUDENT MEMBERSHIP:

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

- ☐ **Join ACM-W:** ACM-W supports, celebrates, and advocates internationally for the full engagement of women in computing. Membership in ACM-W is open to all ACM members and is free of charge.

Priority Code: CAPP

Payment Information

Name _____

ACM Member # _____

Mailing Address _____

City/State/Province _____

ZIP/Postal Code/Country _____

Email _____

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

☐ AMEX ☐ VISA/MasterCard ☐ Check/money order

Total Amount Due _____

Credit Card # _____

Exp. Date _____

Signature _____

Purposes of ACM

ACM is dedicated to:

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

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

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

Satisfaction Guaranteed!

BE CREATIVE. STAY CONNECTED. KEEP INVENTING.



Association for
Computing Machinery

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

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

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



DOI:10.1145/3225222

Pamela Samuelson

Legally Speaking Copyright Blocks a News-Monitoring Technology

An evolving technological landscape has made application of copyright law increasingly difficult.

FOR THOSE WHO want comprehensive access to recent televised news on any topic—be it bombings in Syria, protests in Turkey, tornados in the Midwest, or indictments of Trump campaign officials—TVEyes has been the “go to” news-monitoring service. Its system stores programming from 1,400 broadcast outlets on a 24/7 basis for 32-day periods, transcribes their contents, and indexes the transcripts to enable keyword searching. In response to customers’ search queries, TVEyes’ system generated lists of relevant video clips in reverse chronological order, which when clicked on, would play program segments containing the keywords, starting 14 seconds before the keywords to provide context and lasting no more than 10 minutes. Among TVEyes’ 2,200 subscribers have been the White House, the U.S. Department of Defense, 100 members of Congress, Goldman Sachs, Bloomberg, Reuters, and two major broadcast networks.

In response to Fox News Networks’ copyright infringement lawsuit, TVEyes raised a fair use defense. Although a trial court upheld this defense as to the system’s most salient features, the Second Circuit Court of Appeals in February 2018 ruled Fox was right: the challenged uses were unfair. The court’s opinion has significant implications for developers of technology-intensive services intended to offer similar features.

Search- Versus Watch-Related Functions

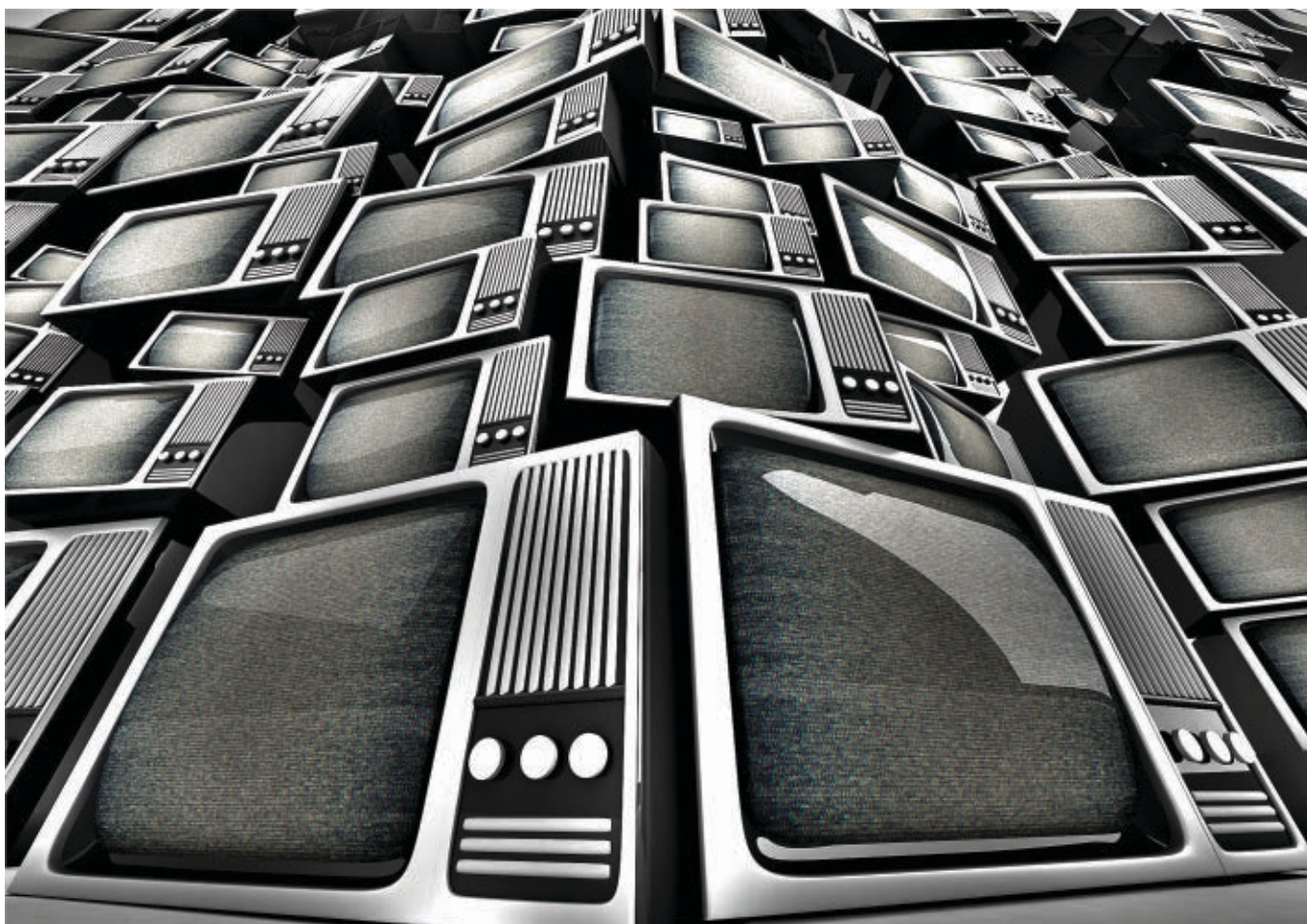
Fox did not challenge the extensive copying TVEyes did of its programs in creating its full-text searchable database. This is probably because this copying was so similar to Google’s scanning of 20 million books to index their contents for a full-text searchable database that the Second Circuit had ruled was fair use in its *Authors Guild v. Google* decision. The court also regarded Google’s serving up “snippets”

of book contents in response to user search queries as fair use.

Fox’s lawsuit focused on numerous watch-related services. TVEyes not only copied news programs in responding to search queries, but also allowed customers to set up “watch lists” for future relevant news, to archive video clips on TVEyes’ servers, to share video clips with third parties, and even to download clips and transcripts to customers’ computers. (To bolster its fair use argument, TVEyes contractually required its clients to limit their uses of the clips and transcripts to internal purposes.)

TVEyes’ Fair Use Argument

When assessing whether a challenged use is fair or infringing, U.S. copyright law directs courts to take into account four factors: the purpose of the defendant’s use, the nature of the copyrighted work, the amount and substantiality of the taking, and harm to actual or potential markets for the protected works.



TVEyes argued that its uses of Fox news programs were “transformative” because they were for a different purpose than Fox’s uses. (That is, TVEyes was using Fox contents for information-locating purposes, whereas Fox was providing news for its customers to consume.) It pointed to the Second Circuit’s *Google* decision holding that copying 20 million books for Google’s Book Search project was transformative because it had a different purpose. Transformative uses are more likely than non-transformative (that is, consumptive) uses to be fair. Moreover, TVEyes was enabling its customers to do research and news analysis, both of which are favored uses under the fair use provision of U.S. copyright law.

TVEyes argued that the factual nature of Fox news programs tipped the nature-of-the-use factor in favor of fair use. After all, numerous cases have found that fact-intensive works enjoy a narrower scope of copyright protection and a broader scope of fair use.

While the amount of copying in *TV-Eyes* was unquestionably extensive, so was the quantity of copying in the

Google case. Yet, the Second Circuit ruled that Google’s copying of 20 million books to index their contents was reasonable in light of its transformative purpose, and so TVEyes could credibly say its use was similarly reasonable.

TVEyes argued that Fox had not suffered harm because the video clips its customers made (85% of which lasted less than one minute) were not substitutes for Fox programs. Second Circuit cases have held that copyright owners are not entitled to control all transformative use markets. Only if the challenged use would supplant demand for the original should a use be deemed unfair. People have not stopped watching Fox news due to TVEyes’ service.

Fox’s Arguments Against Fair Use

Fox disputed TVEyes’ assertion that its purpose in using Fox programs was transformative. The Fox video clips TVEyes was serving up to its customers had exactly the same content as when the programs were initially televised. The non-transformative nature of the use cut against fair use, as did the com-

mercial nature of TVEyes’ enterprise.

Fox also challenged TVEyes’ nature-of-the-work theory because its system was copying and displaying creative expression from the Fox programs. It was not just extracting facts or assertions about facts from the programs.

The amount of TVEyes’ copying for its watch-related functions was, moreover, more extensive than Google’s. To avoid undercutting copyright markets, Google showed no more than three small snippets of text from books in response to search requests. It had, moreover, technologically restricted access to expressive contents beyond what was necessary to assess the relevance of information. TVEyes, by contrast, allowed customers to watch as many video clips as they wanted and to watch up to 10 minutes per clip.

Fox’s main harm argument was that TVEyes was usurping a valuable licensing market opportunity for Fox. To show this was not just a hypothetical market, Fox offered evidence of revenues it had derived from licensing of other video-clipping services.

Amicus Curiae Interest

The high-profile nature of the *TVEyes* case was evident from the 13 amicus curiae (friend of the court) briefs filed with the Second Circuit. Seven were in support of TVEyes, and six in support of Fox. (Amicus briefs are typically filed by firms or individuals whose arguments supplement or complement arguments made by the litigants, often explaining the amici's perspective on the policy implications or consequences of the court's decision.)

Among the TVEyes-side amicus briefs were those filed by Google, the Computer & Communications Industry Association, the Electronic Frontier Foundation, and several library associations. The Fox-side amicus briefs included ones by journalist and photographer organizations, the National Association of Broadcasters, CNN, and the National Cable & Telecommunications Association.

Second Circuit Ruling

Rather than doing a fair use analysis on a feature-by-feature basis, as the trial court had done, the Second Circuit divided the TVEyes uses of Fox contents into two categories: the search-related features, which Fox had not challenged, and the watch-related features, to which Fox objected. The court's opinion was measured in its analysis of each fair use factor, but ultimately concluded TVEyes was not a fair user.

Although one judge on the three-judge panel agreed with Fox that TVEyes' use was non-transformative, the majority decided that the watch-related features were "modestly transformative" because TVEyes' service had a different purpose than Fox's broadcasts. TVEyes' copying of Fox contents was, as in *Google*, transformative "insofar as it enables users to isolate, from an ocean of programming, material that is responsive to their interests and needs to access that material with targeted precision." Without a service such as TVEyes, that information would be "irretrievable or else retrievable only through prohibitively inconvenient or inefficient means."

The most interesting part of the *TVEyes* opinion was its characterization of improved efficiency in content delivery as indicative of the firm's transformative purpose. It likened this to

The court's opinion was measured in its analysis of each fair use factor, but ultimately concluded TVEyes was not a fair user.

the time-shift copying at issue in the Supreme Court's 1984 decision in *Sony v. Universal*, which challenged Sony's sale of videotape recording equipment to enable customers to make copies of television programs. Like Sony, TVEyes enabled its clients to find and watch specific content of interest to them without having to watch it at the time when it was initially broadcast. Yet, the commercial nature of TVEyes' service somewhat undercut the transformativeness argument.

As in many fair use cases, the nature-of-the-work factor was given very little attention and played an insignificant role in the *TVEyes* fair use ruling.

The amount-used factor favored Fox "because TVEyes makes available virtually the entirety of the Fox programming that TVEyes users want to see and hear." TVEyes enabled its users to make far greater use of Fox's copyrighted content than the three snippets per book that Google was serving up in response to user search queries. Insofar as TVEyes allowed users to watch and make copies of up to 10 minutes of Fox news, that would often convey "the entirety of the message conveyed by Fox to [its] authorized users." News segments are often shorter than 10 minutes.

What seems to have undermined the fair-use defense more than anything else was the perceived harm to an existent licensing market for video clips. The court thought Fox had a legitimate interest in controlling that market. Moreover, "[t]he success of the TVEyes business model demonstrates that deep-pocketed consumers are willing to pay well for a service that allows them to search for and view se-

lected television clips, and this market is worth millions of dollars in the aggregate." Because TVEyes clearly valued Fox content and was charging its customers substantial sums for access to it, it ought to be willing to license the content instead of getting it for free.

With the amount-used and harm factors cutting against fair use, the nature-of-the-work factor being neutral, and the purpose factor weighing only slightly in favor of TVEyes, the court concluded that Fox should prevail on summary judgment (that is, without having to go to trial). It directed the lower court to issue an injunction against the watch-related functions of TVEyes' system.

Conclusion

TVEyes has discontinued use of Fox programs for now; it remains to be seen whether it can reach a licensing agreement with Fox that would allow it to continue to offer comprehensive coverage of breaking news. If the litigants cannot reach a deal, one key value of the TVEyes system—its comprehensiveness—will be undermined. After Fox's win, other broadcasters may insist that TVEyes get a license from them as well. In order to maintain comprehensiveness, TVEyes will have to overcome holdup problems with broadcasters who realize they have this news-monitoring service over a barrel, so to speak. One possibility, which TVEyes is probably exploring, is whether allowing clients to watch shorter and/or fewer segments might get it back under the fair use roof.

The concurring opinion raised an alarm about the likely development of other technology services that, like TVEyes, would aim to improve the efficiency of delivery of copyrighted content. The judge objected to calling those types of services "transformative" because this might encourage technology developers to create scaled-down TVEyes-like services. Although his skeptical view did not prevail in the *TVEyes* case, it signals some caution for technology developers who aim to achieve a similar purpose to the TVEyes system. **C**

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

Copyright held by author.

Economic and Business Dimensions Blockchain Revolution without the Blockchain?

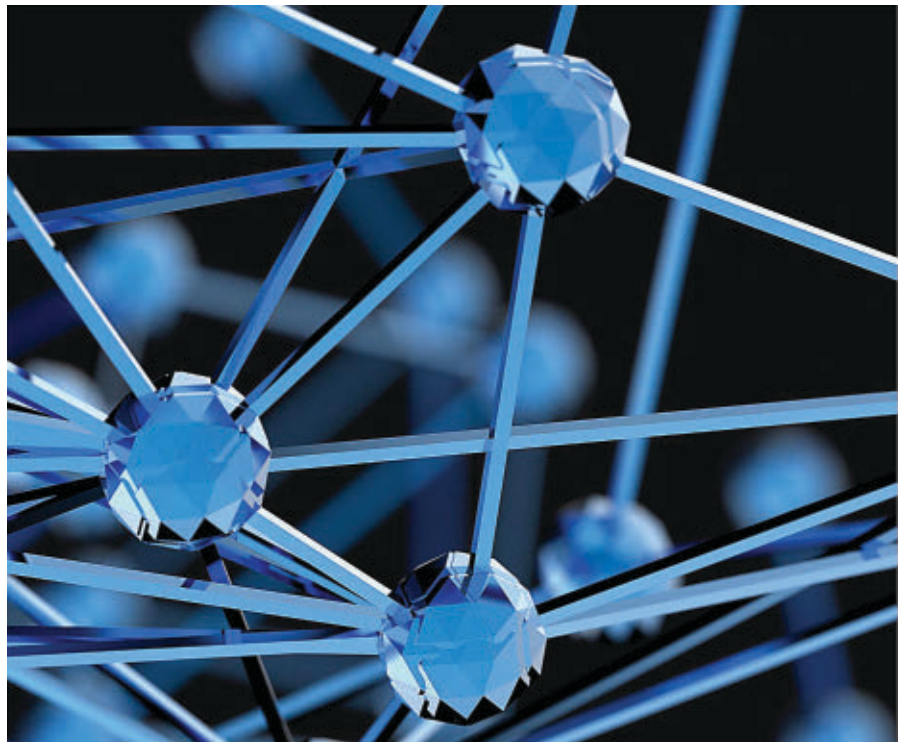
Most of the suggested benefits of blockchain technologies do not come from elements unique to the blockchain.

BLOCKCHAIN HAS ATTRACTED a lot of attention. Many are excited about this new technology based on a public, permissionless, distributed ledger that cryptographically assures immutability without a need for a trusted third party and allows for smart contracts. Large and small companies want to get on board, as they expect this technology will lower their costs by making transactions quicker, safer, transparent, and decentralized. However, the technology behind the blockchain is for the most part not well understood—there is no consensus on what benefits it may really bring, or on how it may fail.

A more careful look into the technology reveals that most of the proposed benefits of “blockchain technologies” do not really come from blockchain. Smart contracts, encryption, and distributed ledger are separate concepts. The three may be implemented together, but they do not need to be. Most of the proposed benefits come from encryption and smart contracts. But encryption and smart contracts do not need blockchain.

Confusion Around What Blockchain Actually Is

The growing excitement about blockchain technologies is perhaps best



summarized in the increasingly popular slogan “blockchain revolution.” The revolution is buoyed by a few forces, of which the most significant is the expectation of substantial cost savings.

The main sources of savings are supposed to come from increased security, faster transactions, and a shared ledger. However, the statements about the benefits of blockchain seem to confuse

three different concepts: encryption; automated execution of transactions (“smart contracts”); and distributed ledger, a type of a distributed database. The three may be applied together, but they are separate tools, and not all of them are necessary in a blockchain system.

So, what is “blockchain”? While there is no one standard definition of blockchain, the most parsimonious

and commonly used is “distributed ledger of transactions.”^a This is why the term “blockchain technologies” is often used interchangeably with the phrase “distributed ledger technologies.”

Where Is the Confusion Coming From?

The source of confusion around blockchain can be traced to the origin of the term. “Blockchain” was introduced as a shorthand term for “chain of blocks of transactions,” which was part of the Bitcoin system.⁴ Later, “blockchain” became an independent term in media discussions of whether there are other uses for distributed ledgers of transactions beyond Bitcoin.

Bitcoin’s system—a system operating without a trusted third party—has been quite successful since it started in 2009, in the sense there has been no fraud on its blockchain. For this reason, it is often said to be secure. Bitcoin’s blockchain is also public (all transactions are visible), and permissionless (any computer may participate in validating transactions and adding them to the ledger).

Some pundits erroneously extrapolate that any blockchain will have these properties: distributed, secure, public, permissionless, and will operate without the need for a trusted third party. This extrapolation may come from a misconception that the Bitcoin’s blockchain properties come solely from technology, while in reality they come from a combination of technology and an incentive system that accounts for the behavior of human participants. Yes, the Bitcoin system uses cryptographic tools. But the reason why the system is virtually immutable is because it is too costly to “rewrite the history.”

Note that smart contracts are not a core property of the Bitcoin blockchain. The Bitcoin system has a rudimentary capability to create code that would allow for some transactions to be automatically executed. Ethereum expanded on this feature, introducing a blockchain with a main

Current applications of blockchain have gathered only limited appeal.

purpose to facilitate smart contracts (see <http://www.ethereum.org>.) Since the term “smart contracts” entered the mainstream media in the context of blockchain, this may have created a perception that smart contracts are native to blockchains. However, a code automatically executing a transaction can be implemented by a wide range of entities.⁵

Therefore, smart contracts, encryption, and distributed ledger are separate concepts. They may be implemented together, but do not need to be. The term “blockchain” should not be used as a catch-all aggregation of these different terms.

Why Is It Important to Consider Smart Contracts, Encryption, and Distributed Ledger Separately?

The distinction matters for estimating costs and benefits, or even predicting the best uses of blockchain technologies. For example, smart contracts are computer programs that automatically implement the terms of an agreement between parties. One typically given example is that of a car lease: upon a missed payment, the car automatically locks and returns the control to the lender. Since execution of a smart contract does not involve a decision or an action of a human, it may increase speed as well as minimize the number of mistakes. Both would result in cost savings.

Some media outlets state that “through blockchain technology, smart contracts are now a reality.”³ However, smart contracts were a reality long before: an automated recurring payment that someone sets up with his or her bank or a limit order with a stock exchange are examples of smart contracts. Blockchain is not needed to gain the benefits from smart contracts, because smart con-

tracts can be set up just as effectively on a centralized system.

Other significant cost savings may come from improved encryption, which results in increased security of the system. Currently, encryption is underutilized in business practice. Bitcoin’s blockchain itself uses standard, well-established cryptography tools. But excitement about blockchain’s safety turned more attention to the new developments in cryptography.

What Are the Benefits of Blockchain?

What about the benefits of a distributed ledger—the blockchain itself? A distributed ledger allows multiple parties in the system to add transactions to a shared ledger in a way that the changes are reflected consistently across all copies.^b It brings benefits in places where reconciliation of contradictory ledgers is costly. At the same time, recording transactions on a shared ledger takes more time than on a centralized ledger, because of the reconciliation mechanisms (consensus mechanisms) that must be employed. Moreover, the need to store the ledger in multiple locations may significantly add to storage and computational costs. So far it has not been clearly demonstrated in which circumstances the benefits of employing a distributed ledger outweighs the cost of delays and duplicated storage.

Distributed ledgers are a special case of distributed databases. They have been known, and used, for three decades. But proponents of blockchain technologies expect more from the new technology than just distributed ledger. They expect that adopting blockchain could result in further cost savings due to disintermediation, as it does not require a trusted third party to be virtually immutable. Indeed, the core of Bitcoin’s computer-scientific innovation was the security of a permissionless distributed ledger, so that there is no need for a trusted third party anywhere in the system.

^a Note that “ledger of transactions” is different from “ledger of balances.” The former keeps the history of transactions, as in the “chain of blocks of transactions.” Using this definition, “Ledger of balances” would not be a blockchain.

^b Technically, distributed databases also have other desirable properties, but this one seems to be the focus in the context of blockchain technologies and fintech.

However, these benefits may be difficult to realize in a blockchain without Bitcoin. It has proven to be a challenge to create a decentralized, permissionless, and safe blockchain to transfer assets other than the native cryptocurrency (for example, bitcoins).

The first major issue is the gateway problem: The information about the underlying assets must enter the blockchain in the first place. The second major challenge is ensuring immutability of the ledger without a native currency. In most of the currently proposed applications, both these issues have been addressed by creating closed, permissioned blockchains, which require some involvement of a trusted third party. This is because blockchain without bitcoins is no longer virtually immutable without a trusted third party. In many cases, the permissioned blockchains are the right tools for their purpose, but more often a centralized system would be more efficient and reliable.

Current applications of blockchain have gathered only limited appeal. Bitcoin's blockchain is the most successful, but even after a decade Bitcoin has been adopted as a payment method only for specific niches. Mainstream users often indicate existing payment systems, such as credit and debit cards, not only satisfy their needs, but also provide services above what Bitcoin delivers.²

There are ideas for other, non-currency applications of blockchain, such as real-estate ownership records, voting information, or identity verification. However, a careful look into these areas shows the problems there do not arise from the need for a distributed ledger of transactions.

Consider an example of the pilot program administered by the Cook County real-estate office.¹ When someone acquires property, they usually need to purchase title insurance in case someone else claims the ownership property over the seller. The Cook County office was wondering whether putting the real-estate ownership on a blockchain would resolve this uncertainty. However, the major cause of the title uncertainty is that when a property is sold, there is no obligation to report it to the county office (or elsewhere). It is enough to have a written sales contract as a proof. Moreover,

the sales reported to the county office are manually entered into the system, which results in typing errors. Neither of these problems is solved by implementing a blockchain ...

The Future of the Blockchain Revolution

I expect blockchain technologies will have a big impact on many industries, and that it will not be limited to finance. However, it may not happen in the way it is currently envisioned. Both the entrants and the incumbents are looking with interest at the properties of Bitcoin's blockchain and smart contracts. But as they realize the benefits of different elements of the system, it may turn out that while new encryption tools and automated execution of transactions (smart contracts) have large and clear benefits, distributed databases may have a more limited appeal. Most of all, we need to realize that outside of Bitcoin (or other cryptocurrencies) we do not have a technology that offers "permissionless distributed ledgers that cryptographically assure immutability without a need for trusted third parties."

The blockchain revolution may give us new tools and change the landscape of some industries. But since the benefits of encryption and smart contracts can be realized without a distributed ledger, the world after the blockchain revolution may well be a world without the blockchain. C

References

1. Cook County Recorder of Deeds. Blockchain Pilot Program. Final Report. May 30, 2017; <https://bit.ly/2TeWduZ>
2. Henry, C., Huynh, K., and Nicholls, G. Bitcoin awareness and usage in Canada. BoC Staff Working Paper 2017-56. (Dec. 2017); <https://bit.ly/2IjblMR>
3. Lielacher, A. A cost-benefit analysis of using smart contracts in banking. BTCManager.com (Apr. 14, 2017); <https://bit.ly/2IIQJel>
4. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System, 2008; <https://bit.ly/2IgFLx8>
5. Szabo, N. Formalizing and securing relationship on public networks. *First Monday* (Sept. 1997); <https://bit.ly/2IgFLx8>

For more extensive discussion on the topic see "Blockchain Revolution without the Blockchain," BoC Staff Analytical Note 2018-5; <https://bit.ly/2GhXhva>

Hanna Halaburda (hhalaburda@gmail.com) is a Visiting Professor at NYU-Stern and a Senior Economist at the Bank of Canada.

The views expressed in this column are those of the author. No responsibility for them should be attributed to the Bank of Canada.

Copyright held by author.

Calendar of Events

July 2-4

ITiCSE '18: Innovation and Technology in Computer Science Education, Larnaka, Cyprus, Sponsored: ACM/SIG, Contact: Janet Read, Email: jcread@uclan.ac.uk

July 7-8

CI 2018: Collective Intelligence 2018, Zurich, Switzerland, Sponsored: ACM/SIG, Contact: Abraham Bernstein, Email: bernstein@ifi.uzh.ch

July 8-11

UMAP '18: 26th Conference on User Modeling, Adaptation and Personalization, Singapore, Co-Sponsored: ACM/SIG, Contact: Jie Zhang, Email: zhangj@ntu.edu.sg

July 8-12

SIGIR '18: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, Ann Arbor, MI, Sponsored: ACM/SIG, Contact: Kevyn Collins-Thompson, Email: kevynct@umich.edu

July 9-12

HT '18: 29th Conference on Hypertext and Social Media, Baltimore, MD, Sponsored: ACM/SIG, Contact: Dongwon Lee, Email: dongwon@psu.edu

July 15-19

GECCO '18: Genetic and Evolutionary Computation Conference, Kyoto, Japan, Sponsored: ACM/SIG, Contact: Keiki Takadama, Email: keiki@inf.ucc.ac.jp

July 16-18

SPAA '18: 30th ACM Symposium on Parallelism in Algorithms and Architectures, Vienna, Austria, Co-Sponsored: ACM/SIG, Contact: Christian Scheidele, Email: scheidel@upb.de

► Richard Ladner, Column Editor

Broadening Participation Beyond Diversity

Considering the confluence of research questions and sociopolitical dynamics.

I WAS NERVOUS when I took the stage. Despite assurances from everyone I had shown my speech, I was nervous because I was about to tell the 400 attendees at the 2017 ACM CHI conference's Diversity and Inclusivity Lunch that “diversity” and “inclusivity” are not enough.

There is a rapidly growing appreciation that diversity is a Problem That Must Be Solved in computing; as an example, despite women earning 57% of undergraduate degrees in the U.S., we earn only 18% of degrees in computer and information sciences (see <https://bit.ly/1W7j2Re>). But it is not just about women: academics facing multiple oppressions—homophobia, transphobia, ableism, racism, anti-Blackness, and intersections of all these^a—have much to say about the often-entrenched views of the “old guard” and the institutions they control. During my time at CHI 2017, I found this firsthand. But what I also found inspired me: young academics (mostly graduate students, many of them queer) are turning a critical eye to established research practices and transforming computing research.

^a Throughout this column I will refer to LGBTQ people (lesbian, gay, bisexual, transgender, queer). Transgender people are individuals whose gender does not match the one they were assigned at birth. I will also use first-person plural pronouns (we, our) variably to refer to these groups and identities I occupy. Finally, as a U.S.-based graduate student, this column features only my own limited perspective.



Gender-identity and sexual-orientation symbols on a Rubik's-cube-themed display in Paris, France, 2017.

Essential to this transformation is the understanding that powerful institutions—academia among them—often embrace “diversity and inclusion” but stop short of structural change.⁷ Who gets to decide who is “the right kind of diverse”? What are people being included in? Who is doing the including, and for what reasons? The answers to these questions hinge upon the social, economic, and political power structures that form the fabric of our society. In my speech, I argued that tokenization and privilege pervade the self-indulgent initiatives that often mark “diversity work.”² I reflected on

my experiences being tokenized: highlighted for my “diverse” personhood, and yet being unable to control my own narrative or to bring about meaningful, positive changes in my institutional environment. I wondered aloud whether I would be standing at that podium if I were raised in poverty, if I were a Black or indigenous person, or if I were disabled. Despite being both queer and a trans woman of color, my social position affords me privilege, safety, and a platform to speak. Critically, however, it does not matter how many “diverse” people are let into the room if we do not possess the power to change what hap-

pens inside it. Without that power, we can be ignored, silenced, or removed, and the status quo remains intact.

These issues impact not only the environments in which we do our work, but also the work itself. I have realized that asking certain research questions—such as how to build technology for women, disabled people, or trans people—requires critical engagement with sociopolitical problems. In my field of human-computer interaction (HCI), we have recently seen the development of important frameworks and approaches to support this: feminist HCI, anti-oppressive design, and social justice-oriented design. These are much more than research tools; as a graduate student slowly developing my mind-set and conceptual framing, they were impactful. And in a computer science department whose faculty, students, and administration largely look nothing like me, they are indispensable.

CHI 2017 featured papers that applied and built upon these frameworks and introduced new ones: they analyzed the full corpus of the conference's proceedings with an intersectional feminist lens; explored technological interventions for sex workers; and promoted an equitable participatory relationship between disabled people and assistive technology researchers. I made a point of meeting several like-minded authors throughout the conference, and as we laughed, shared our struggles and successes, and talked trash about the old guard, I began to believe we could change the future together. Or, at the very least, I wanted it.

One of many avenues that beg further exploration in HCI involves transgender people. As a group, we are systematically oppressed at both the institutional and individual levels: we face disproportionate violence due to hate crimes and punitive policing practices; barriers in access to both primary and transition-related health-care (and significant health disparities more broadly), denials of coverage by insurance providers and lack of provider knowledge, discrimination in public accommodations, housing, and employment; the list goes on.³ And, of course, these injustices can be ameliorated or intensified by race, class, gender, and disability. Schlesinger's analysis indicated that only three papers in

Who gets to decide who is “the right kind of diverse”?

CHI's history dealt directly with the experiences of transgender people.⁶

These papers, among others, tend to deal with the experiences and challenges of transgender users within existing technologies, such as Facebook and Tumblr. Such studies are critical in understanding how interactive systems fall short of providing trans people self-determination; they consistently find the assertion of identity, fear of reprisal and judgment, and the importance of collective belonging mark our interactions with technologies. From my own experiences, these findings extend to our daily lives, and the everyday struggles that arise from living in a society that oppresses us. It should come as no surprise, then, that most interactive systems we use today reflect and reinforce dominant and damaging cultural narratives. Technology designers—and researchers more broadly—therefore inhabit a privileged position; whether we perpetuate or subvert oppressive social structures is *our decision*. My goal, which I share with the inspiring researchers I met at CHI 2017, is to subvert narratives that silence and suppress, that obstruct our self-determination, and that strip us of power. I am committed to using my privilege to collaboratively design technologies with and for trans people.

As an example: in my work designing trans health technology, I interviewed several trans people about their experiences. One woman said that the pressure to come out “perfect”—as an embodiment of a stereotyped and objectified notion of womanhood—completely stopped her from transitioning and forced her to stay in the closet for decades, at the severe expense of her mental health. These narratives are reflected not just in media and culture, but also in the resources and health professionals that trans people frequently navigate throughout their lives. My dissertation work focuses on building a mobile application for voice train-

ing. Trans people may feel as though our voices do not represent us; this does not just hurt internally, but could also put us at risk for harassment or assault. When we present ourselves to the world (through our speech, dress, mannerisms, or otherwise), we might find ourselves walking a line between what we want to see in ourselves and what others expect of us. Another of my interview participants said she sometimes presents herself as more feminine than she feels in order to be gendered correctly by others. While a voice-training app could be useful, it could easily become prescriptive and propagate notions about how trans people should sound, act, and live. Without appreciating the complex personal and social contexts in which such technologies would be used, they will fail.

Conclusion

At CHI 2017, Ann Light presented her alt.chi submission (for which she won a Best Paper Award),⁴ which considered what HCI researchers should do in the face of existential crisis, a political and socioeconomic climate hostile to many, and the continuing decline of the environment and the associated uprising of the upper class. She was asked, “Are you hopeful?” To which she responded, “No, but I am determined to make change.”

Her words inspired me to fight for change in my institutional environments *and* through my research, and I am excited to join a community of researchers who share that goal. As we pursue work that appreciates the multifaceted nature of human identity, the injustices of stigma and oppression, and the shared responsibility of technology designers, engineers, and academics to work toward *true* social good (the health of our people and our planet),⁵ we are making a statement. And with every author, advisor, and collaborator behind that work, we are building collective power. We are beginning to move beyond diversity and inclusion—*are you joining us?*

Addendum

Because this column was first written following CHI 2017, I want to add some thoughts on this year's conference, which wrapped up at the end of April. If you made a word cloud from the SIG-

**Multiparty Privacy
in Social Media**

**Amdahl's Law
for Tail Latency**

**How to Teach
Computer Ethics
through Science Fiction**

**Point/Counterpoint
on E-Democracy**

**Accelerating GPU
Betweenness Centrality**

Consistently Eventual

**Research for Practice:
Prediction-Serving
Systems**

**Algorithms
Behind Modern
Storage Systems**

Plus the latest news about
how animals can teach robots,
3D stacking, and women in
computer science.

CHI administrators' speeches during the conference, "diversity" would likely be in large type. During the opening remarks, it was stated CHI was intended to be "the most diverse conference ever." Ironically, the keynote speaker was Christian Rudder (the founder of OkCupid). In his keynote speech, Rudder presented a woman's profile picture and commented on her lack of personality and showed graphs that gauged the preferences of the "hottest guys" and the "hottest girls" (see Figure 1, a tweet by Rosie Bellini, a graduate student at Newcastle University). When asked about the behaviors of non-heterosexual users on the platform, he said "guys will be guys and girls will be girls." Rudder's response to an ethical question was particularly disconcerting to the audience of HCI researchers, many of whom have made careers out of studying the ethics of technology development (see Figure 2).

Many attendees felt wronged by the poor choice of keynote speaker, but more so by the fact that we had no say in the process. I learned that next year the speakers will be chosen by committee, but this situation is symptomatic of a larger and more complex issue: that the structure of the ACM and SIGCHI are both very top-down. The average ACM member has little say in the day-to-day operations and policies of the organization, and conference speakers, venues, and other choices are not made democratically. As a result, we can point out various problems—such as the fact that the vast majority of CHI 2018 attendees were from North America and Europe, and less than 20 from Africa—but we have no concrete means to enact change under the current system. As I have written elsewhere,¹ this also leads to researchers facing strict content limitations when attempting to publish on "taboo" or "offensive" topics. When our work is altered or censored for this reason, we have no recourse.

We cannot talk about diversity without talking about politics. After Rudder's keynote, concerned members of the CHI community organized together to write an open letter and deliver a speech to the SIGCHI town hall meeting. Raising our collective voice is important and allows us to demand that large, powerful institutions work for *all* of us.

Figure 1. Twitter reaction to the CHI 2018 keynote speech; <https://bit.ly/2JfFhHh>

It sure is well crafted satire for CHI to boast on a record year for diversity and inclusion & then has an opening keynote that talks about 'ugly' and 'hot' women #CHI2018
10:57 PM · 23 Apr 18

Figure 2. Twitter reaction to the CHI 2018 keynote speech; <https://bit.ly/2smzUfB>

It's never a good sign when the speaker responds to a question by saying, "What do you mean 'ethics'? What do you mean by that?" #chi2018
11:16 PM · 23 Apr 18

We still need to ask and answer crucial questions about what it means to politically organize at an annual conference, and in the context of a global academic research institution. Academics, especially graduate students forming unions at high rates, are finding our political voice. And now is the time for us to use it. **C**

References

1. Ahmed, A.A. et al. What's at issue: Sex, stigma, and politics in ACM publishing. In *Proceedings of the 2018 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '18).
2. Ahmed, S. *On Being Included: Racism and Diversity in Institutional Life*. Duke University Press, 2012.
3. James, S. E. et al. *The Report of the 2015 US Transgender Survey*. National Center for Transgender Equality, Washington, D.C., 2016.
4. Light, A., Shklovski, I., and Powell, A. Design for existential crisis. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '17), (May 2017), 722–734; <http://doi.org/10.1145/3027063.3052760>
5. Pal, J. CHI4Good or Good4CHI. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '17), 2017, 709–721; <http://doi.org/10.1145/3027063.3052766>
6. Schlesinger, A., Edwards, W.K., and Grinter, R.E. Intersectional HCI. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (CHI '17), ACM Press, New York, NY, USA, 2017, 5412–5427; <http://doi.org/10.1145/3025453.3025766>
7. Stewart, D.-L. Language of appeasement. *Inside Higher Ed.* (Mar. 30, 2017); <https://bit.ly/2nnkEKM>

Alex Ahmed (ahmed.al@husky.neu.edu) is a fourth-year doctoral student in Personal Health Informatics at Northeastern University, Boston, MA, USA. Her dissertation research focuses on the community-based design and development of technology for transgender health. She is an organizer with Graduate Employees of Northeastern University-United Auto Workers (GENU-UAW).

Copyright held by author.

Viewpoint

A New Perspective on Computational Thinking

Addressing its cognitive essence, universal value, and curricular practices.

THE IDEA OF adding computational thinking (CT) to a child's analytical ability goes back almost four decades,²⁰ yet its recent promotion²⁹ as an "attitude and skill set" for everyone has helped popularize it all over the world. While periodic reviews on the status of CT education^{6,11} indicate wide agreement on what comprises CT, there is a struggle in the field by teachers and educators on how to integrate CT practices and skills into K–12 education. Many researchers and educators who initially supported the idea of teaching CT skills to everyone are now wary of its promise. Some of the remaining trouble spots include definition, methods of measurement, cognitive aspects, and universal value of CT.⁶ This Viewpoint presents an alternative perspective on computational thinking, positioning CT as a link to cognitive competencies involved not only in science and engineering but also in everyday life.

A major source of current troubles with CT comes from linking it to electronic computing devices and equating it with thinking by computer scientists. Accordingly, many of currently recognized CT skills are associated with problem solving and use of electronic devices with a goal of preparing tomorrow's programmers.^{11,29} A decade of discourse and experimentation has yet

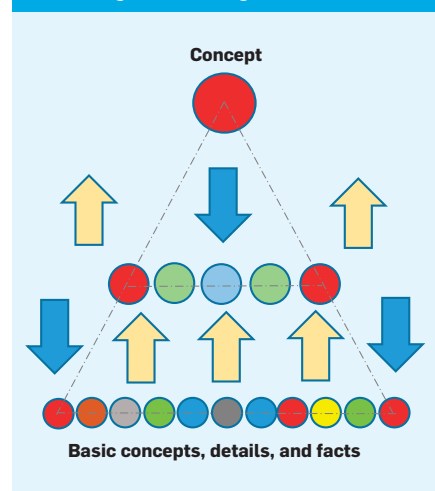
to produce ways to separate CT from programming and the use of electronic devices. And, the lack of such separation continues to preclude us from capturing the cognitive essence of CT.

Teaching experts' habits of mind to novices is inherently problematic because of prerequisite content knowledge and practice skills needed to engage in the same thinking processes, not to mention the cost of providing them a similar environment to conduct inquiry and design. This problem is not unique to CT education; it also applies to scientific thinking (ST) and engineering thinking (ET) education.^{7,26} A remedy that applies to all of

them is to link experts' habits of mind to fundamental cognitive processes so we can narrow the skillsets down to more basic competencies that can be taught to novices.³¹

Linking CT to cognition is not a new idea. In fact, it is what led to the design of electronic computing 80 years ago when Alan Turing²⁷ suggested that if thoughts (that is, information) can be broken up into simple constructs and algorithmic steps, then machines can add, subtract or rearrange them as our brains do. Electronic machines have since taken many complex and voluminous computations off our brains, further supporting the view of brain as a biological computational device.¹⁹ Unfortunately, understanding how biological computing generates cognition from electrical activities of neurons has been hindered by the fact that it involves a delicate, inaccessible, and complicated organ, the brain. The good news is that technology has recently broken some of these barriers. For example, neuroscientists now use imaging techniques to understand brain mechanisms that take part in receiving, storing, retrieving, and processing information. Cognitive psychologists use similar techniques to study where in the brain particular perceptual and cognitive processes occur. At the same time, cognitive and computer scientists form theories and

Figure 1. A universal mechanism by which all heterogeneous things form and evolve.³⁰



models of the mind to study how computation may be generating thinking.

Electronic computers have evolved to showcase many structural and functional similarities with the brain. So, we may have a chance to better understand how the brain works through easier access, use, and control of electronic devices. I suggest the similarities arise from quantifiable aspects of information constructs, as suggested by Alan Turing,²⁷ and the appearance of a universal mechanism (see Figure 1) by which quantifiable things form and evolve.³⁰ That is, like the granular matter, information constructs either unite associatively, as shown by the bottom-up arrows in Figure 1, to make bigger constructs or break down distributively, as shown by the top-down arrows, to smaller ones. Computing devices, be it electronic or biological, are likely to use similar ways to track and tally this invariant behavior of information. Another reason for similarities is the design, use, and control of electronic computing devices by biological computing agents.

Continuing the legacy of Turing to focus on device-independent processes (see Figure 2), we want to create more links between CT and cognition by identifying common patterns of information processing that are known to facilitate thinking. This may give us a framework to suggest a universal definition for CT—*thinking generated and facilitated by computation, regardless of the device that does the computation*—along with an electronic computing methodology to facilitate relevant cognitive processes. While the CS community is willing to modify its original definition along these lines,^{1,28} current curricular CT practices still deal only with teaching of electronic CT skills.

A clear distinction should be made between electronic and biological CT to more effectively integrate desired CT skills to the relevant grade-level curricula. Having dealt with many issues of CT education for three decades at both college and K–12 levels,^{30–34} I want to present an interdisciplinary perspective to address both cognitive and curricular aspects of CT by merging CS education research with concepts from epistemology, cognitive and neurosciences as briefly described here.

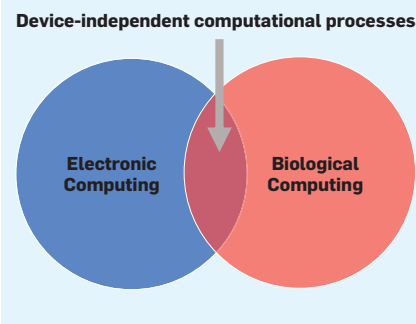
Neuroscientists see little or no distinction now between the acts of information storage/retrieval and the act of thinking.

Neuroscience's View of Information Storage, Retrieval, and Thinking

Contrary to the early compartmentalized and centralized design of electronic computers, the brain employs a distributed network of neurons to store, retrieve, and process information. Information gets stored into the memory in the form of a specific pattern of neurons placed on a pathway and fired together,¹⁴ as shown in Figure 3. Therefore, the number and strength of neural pathways are key to improving storage and retrieval of information.

Humans are born with ~100 billion neurons that get connected to each other in various ways as we grow older. Other key factors that affect our mental growth include the functionality that each neuron or groups of neurons assume, the size they grow into, and the placement in different parts of the brain that they migrate towards. More important is the number of neural connections, which could go up to 100 trillion. As we learn things, new connections are being made while the existing ones are strengthened, weakened, or even eliminated if not revisited often enough.

Figure 2. Information processing by electronic and biological computing devices include both device-independent and device-dependent processes.



The latest developments in neuroscience have contributed significantly to our understanding of learning in relation to information retrieval.⁴ Forgetting is now considered to be a good thing because it forces the learner to use effort to cognitively engage and recall or reconstruct newly acquired concepts through different neural pathways or links that exists and are retrievable. So, the more links to associated concepts, the higher the chances of recalling the newly acquired concept when needed later. Furthermore, cognitive retrieval practices attempted at different times, various settings, and contexts are good because every time the recall is attempted it establishes more links that will help the remembering and learning. Exposure to new concepts, then, through links to multiple views from different fields of study is an effective retrieval strategy recommended by cognitive psychologists.

Basically, retrieval sounds like an act of creative reimagination and what is retrieved is not the original pattern but one with some holes or extra bits. Consequently, neuroscientists see little or no distinction now between the acts of information storage/retrieval and the act of thinking. Such a consolidated view of storage, retrieval, and thinking is very much in tune with our model (Figure 1) of how information behaves naturally. Applying it to translate what neuroscientists say about storage and retrieval,⁴ we posit that a memory or a newly learned concept can be a combination or outcome of previously formed memories and concepts, each of which might also involve another level of vast network of concepts and details mapped onto the brain's neural network in a hierarchical way. When new information arrives, it lights up all related cues, neurons and pathways in a distributive process that is similar to the top-down action, where new concept is broken up into related pieces. By the same token, retrieving a memory is a reassembly of its original pattern of neurons and pathways in an associative process that is similar to the bottom-up action.

Accordingly, the brain attempts to analyze *deductively* every new concept and information that it encounters in terms of previously registered models—*objects, faces, scenarios, and so on*.

And, as our knowledge grows further, the relationships among registered information eventually lead to interplay of various combinations and scenarios of existing models that eventually end up *inductively* clustering related details into conclusions, generalizations, and more inclusive models of information.²⁵ As a result, the details our brain registers and stores and the hierarchical connections it establishes between them, along with these generalizations and conclusions, build over time a pyramid-like structure (see Figure 1) that we have come to call *mind*.¹⁹ Cognitive scientists often use a software analogy to distinguish it from the brain as noted here.

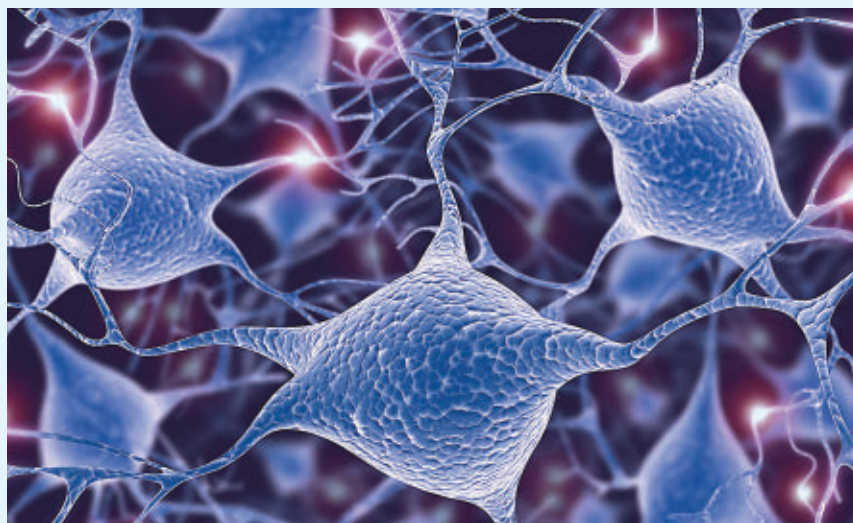
Cognitive View of Information Processing

While the distributed structure of neurons and their connections (hardware) influence cognitive processing (software), the relationship between software (mind) and hardware (brain) is not a one-to-one relationship. According to the biological computing view of mind,¹⁹ its processing of information consists of a hierarchy of many patterns and levels that may range from basic computations to more complex functions (sequence or structure of instructions) and models (mental representations) of perceived reality and imaginary scenarios.

While structural and functional similarities improve our understanding, I do not suggest the brain works exactly like electronic computers. Modeling the mind as a rational decision-making computational device has yet to fully capture mental representations and emotions.¹⁰ In fact, we may never be able to model the human brain unless we understand what intelligence is and how it is possible for the human brain to make decisions on as much electricity as consumed by a dim light. Many believe it does so by simplification and avoidance of exhaustive computations and evaluations of hypothetical scenarios surrounding an issue.¹³

To explain the root causes of the brain's efficient operation, neuropsychologists and evolutionary biologists point to some structural (hardware) interference by an autopilot limbic system to bypass, simplify, or reduce more elaborate cognitive functions of

Figure 3. Illustration of a distributed network of neurons firing and wiring together.



an evolved neocortex. In fact, it appears we are caught up between two competing brains²⁴ whose operations can be well understood by the flow of information processing in Figure 1. Typically, there is a cyclical tendency between simplifying things (bottom-up) and digging things deeper (top-down). One of these processes is fast, effortless, automatic, inflexible, nonconscious, and less demanding of working memory, while the other is slow, effortful, controlled, conscious, flexible, and more demanding of working memory.⁸

Cognitive scientist Read Montague¹⁹ points to some non-structural (software) tendencies to account for our brain's energy-efficient operation. He suggests that concern for efficiency, as part of our survival, leads to assigning value, priority, cost, and goals to our thoughts, decisions and action. To do this, the mind carries out computations, builds models, and conducts evaluative and hypothetical simulations of different scenarios. This may slow down and add imprecision to decision-making. However, because of bundling similar things together via a model, the overall process still ends up saving us from undertaking exhaustive and repetitive computations of various scenarios. According to Montague, the tendency to make trade-offs between simplicity and complexity and between details and generalizations is the root driver of our intelligence, and why we have pushed ourselves to be smarter over time.

Whether the causes are structural or non-structural, there is enough evidence about a duality in information storage, retrieval, processing, and reasoning that warrant further examination. In fact, both structural (hardware) and non-structural (software) drivers of our intelligence, reasoning, and thinking have a common mechanism that is consistent with Figure 1. For example, the act of modeling by our mind to assign value, cost, and goals to our thoughts before decision-making meshes well with the tendency for simplification (bottom-up flow of information in Figure 1). A clear advantage is that it makes it possible to work with approximate, abstract, or average representations, thereby bringing closure to an otherwise unending worry or inquiry about details. Accordingly, the human brain uses *modeling* not only for mental representation of external objects but also for wrapping up its own computations so it can compare their values and costs before deciding,¹³ a cognitive mechanism that epistemologists came up with two centuries ago, as noted here.

Epistemology of Knowledge Development

Epistemology is a branch of philosophy that studies *how we know what we know*. At its core are questions like '*what is true knowledge and its source?*' and '*how can we be sure of what we know?*' While scientists such as Galileo laid a strong foundation for building knowledge

through observations, experiments, and mathematics in the 16th century,¹⁸ philosophers debated for two more centuries whether a scientist's subjective view of the world can be considered as true knowledge.

One of the debated views (empiricism) argued that the mind is a blank slate and that it acquires knowledge through perception and *inductive reasoning*, which involves putting perceptions, experiences and related pieces of information in a synthetic (associative) way to arrive at generalizations and conclusions as depicted by the bottom-up flow (arrows) of information in Figure 1. Knowledge acquired this way is not warranted because new experiences may later change its validity. The other view (rationalism) argued that knowledge is initially acquired through innate concepts which then serve as the source of additional knowledge derived from them in a rational (analytic) way using *deductive reasoning*. In deductive reasoning, a concept generally applies to all members and situations that fall under its representation, as depicted by the top-down flow (arrows) of information in Figure 1. Since innate concepts were considered true, knowledge derived from them was considered to be warranted, not needing further examination.

By arguing against both views, Immanuel Kant created a bridge to lay the foundations of epistemology and today's scientific methodology of inquiry.¹⁵ He recognized what experience brings to mind as well as what mind itself brings to experience through structural representations. He considered that knowledge developed a posteriori through synthesis could become knowledge a priori later. And, a priori cognition of the scientist continues to evolve over the course of science's progress. Although the deductive and inductive cycle of scientific progress has historically been slow until recently,¹⁸ the growing knowledge and the number of researchers tackling a problem have all now shortened the timescale of progress. Concepts and theories once considered true and valid are now quickly being changed or eliminated.

Modeling: A Universal Process

Modeling and testing has been an important tool for scientific research for

Concepts and theories once considered true and valid are now quickly being changed or eliminated.

hundreds of years. In principle, it works exactly as articulated by Kant, and as illustrated in Figure 1. Scientists ideally start with a model of reality based on current research, facts, and information. They test the model's predictions against experiment. If results do not match, they then break down the model deductively into its parts (sub models) to identify what needs to be tweaked. They retest the revised model through what-if scenarios by changing relevant parameters and characteristics of the sub models. By putting together inductively new findings and relationships among sub models, the initial model gets revised. This cycle of modeling, testing, what-if scenarios, synthesis, decision-making, and re-modeling is repeated while resources permit until there is confidence in the revised model's validity. Electronic computers have recently accelerated this cycle because not only do they speed up the model building and testing via simulations but they also help conduct studies that are impossible experimentally due to size, access, and cost.

Modeling and simulation (M&S) appears to be a device-independent process of information that links computing and cognition. Its associative and distributive processes even describe computable actions of other quantifiable things besides information. For example, formation of physical objects or particles from smaller ones resembles the act of modeling because both seem to involve packing *parts* together associatively to form a *whole*. Furthermore, such act of modeling is often driven by external forces or by a collective "trial and error" process controllable by conditions and rules of engagement—much like a simulation.³⁰

Philosophers and psychologists have been studying the *parts-and-whole* dynamics since Plato⁹ to explain the nature and human behavior. Recently, with help from technology, cosmologists and cognitive scientists have also been searching for a universal process that may be guiding the growth of all networked systems, ranging from the tiny brain cells to atoms, to the Internet, and even the galaxies. The view that such a process may be described computationally, as in Figure 1, is now gaining traction because formation and evolution of an abstract idea or a computational model of information appears to be no different than that of a system of physical particles.³⁰ If so, then not only can we learn from an ongoing millennial argument of such a universal topic, but we can also put computing at the center of a discourse well beyond CT to understand the nature itself.

The Essence of Computational Thinking

Our brain's inclination to store, retrieve, and process information in an *associative/distributive* fashion may be a manifestation of a duality engrained in the fabric of matter and information. This inclination may just be an evolutionary response, shaped up over many years, to optimize the handling of sensory information whose quantifiable nature only resonates with distributive and associative operations. A similar evolution can be seen in electronic computing's structural change from a centralized hardware of the past to today's distributed network due to the growing need for faster processing and more storage to solve problems and improve our survival. As Montague suggests,¹⁹ our changing need for simplicity and generalization (via associative processing) as well as complexity and details (via distributive processing) of information has driven us to think harder and become smarter. At the same time, while our brain structure and cognitive processing offer all of us a chance for full utilization of an optimized response to a changing environment, the efficiency, intactness, and effortfulness with which we all use it depends on the individual.

At the core of our CT framework in Figure 4 lies a dichotomy both in the quantifiable nature of sensory infor-

mation and in the way information storage, retrieval, and processing is done by the brain hardware. Since cognitive researchers have demonstrated how information processing could lead to cognitive inferences via inductive reasoning,^{7,25-26} here we are not concerned about details of how information processing generates cognition but rather how duality in fundamental computation may lead to duality in higher-level reasoning. The invariant nature of information affects not only how similarly computing is done at the most fundamental level (that is, addition and subtraction at the core of our framework), but how this similarity would carry itself all the way to the high-level processing at the outer layers in Figure 4. Despite these similarities, however, high-level processing of information with different devices may still have device-dependent aspects, therefore requiring different skills to use each. Basically, the duality in information processing and *its cyclical and iterative use, as in Figure 1, is the very essence of computational thinking* that we all employ for learning, conceptual change, and problem solving. Anyone who wants to use electronic devices to further facilitate this process might need electronic CT skills on top of biological CT skills, as shown in the last layer of Figure 4.

Our framework's relation to habits of mind in other fields, such as ST and ET, has been reviewed favorably by relevant communities.³¹ In fact, ST's inductive and deductive processes are no different than those used in everyday thinking by non-scientists.^{7,26} We all use inductive reasoning to filter out details and place our focus on more general patterns, thereby assigning priority and importance to the newly acquired information. Deductive reasoning, on the other hand, helps us make decisions and draw conclusions from general concepts. Yet, using these CT skills in an iterative and cyclical fashion for inquiry and conceptual change varies for everyone, depending on the underlying brain structure and the quality and quantity of the environmental input it receives. A scientist is a good example of someone who does this in a frequent, consistent, and methodological way, leading eventually to a habit of mind that is often known as ST.

The two currently cited electronic CT skills that resonate with cognitive functions of a computational mind, as we defined here, are *abstraction* and *decomposition*. The rest can be considered as device-dependent skills that may or may not have any cognitive benefits other than being a routine or specific use of an electronic device. Abstraction is an *inductive* process that helps our cognition in important ways, especially at its developmental stages, by simplifying, categorizing, and registering key information for quicker retrieval and processing. Decomposition, on the other hand, is a *deductive* process that also helps us in many ways, including dealing with a complicated situation by distributing the complexity into smaller and simpler pieces in order to attack each one separately until a cumulative solution is found.

We all use abstraction and decomposition skills in our daily lives,³ but not everyone is equally aware of their importance, nor are we all practicing

and utilizing them fully and equally. In that sense, everyone, not just computer scientists, uses CT. But, since abstraction and decomposition skills are heavily used in programming and problem solving,² having students improve them has been a concern of educators. For example, abstraction is used to distribute the complexity of a code *vertically*, as shown in Figure 1, into seemingly independent layers and protocols in such a way to hide the details of how each layer does the requested service. Dijkstra, a pioneer in programming, regarded abstraction as the most vital activity of a competent programmer. In fact, a good programmer is expected to be able to oscillate between various levels of abstraction.² While being able to divorce one's thinking from low-level details and biases is key to finding solutions that can be transformed to different applications, most CS undergraduate students barely move beyond language and algorithm-specific details and biases. Similarly, decompo-

Figure 4. A cognitive framework on the essence of electronic CT skillset in terms of biological CT skills.³¹



sition is used in software engineering as well as in parallel computing to distribute the workload *horizontally*, as in Figure 1, among multiple processors. Unfortunately, automatic compilers are not here yet to help us write parallel codes, and teaching students parallel programming is still a challenge. There are no quick fixes but as mentioned in the next section M&S tools have been found to boost not only students' cognitive functions but also their motivation to learn programming and science content.

Measuring the Impact of CT Education

There are instruments with good psychometric properties to measure the impact of technological pedagogical content development¹⁷ tools on teaching and learning. M&S's interdisciplinary and changing technological nature require customization of its use in instruction and the assessment of its effectiveness in teaching of the content under consideration. Researchers may need to use not only quantitative methods to measure variables involved but also qualitative methods to initially identify those variables and to later understand and triangulate them for validity. The quantitative sources of data often include surveys to gather pre/post activity data, unit test scores, course passing rates, report cards, graduation rates, and achievement scores in standardized tests, while qualitative sources of data may be interviews, classroom observations, and computational artifacts.⁵

Education researchers have identified M&S as an exemplar of inquiry guided learning.^{21,23} These findings are also grounded in learning theories that recognize the role of abstract thinking and reflection in constructing knowledge and developing ideas and skills.³ However, because constructivist and unguided learning works only when learners have sufficiently high prior content knowledge to provide "internal" guidance,¹⁶ use of M&S in K-12 education has been slow. Technological changes in the past decade have given birth to new M&S tools that can shield the learner from high-level content knowledge in math (for example, differential equations), computing (such as programming), and science

(for example, laws of nature), thereby making them accessible to novices for constructive learning.

As noted in peer-reviewed articles,^{32,33} empirical data collected from hundreds of teachers and their students in 15 secondary schools for a period of seven years revealed statistically significant results to suggest that M&S inherently carries a mix of deductive and inductive pedagogies in the same setting. This is great news for educators who want to take advantage of both approaches of teaching. Basically, modeling provides a general simplistic framework from which instructors can *deductively* introduce a topic without details, and then move deeper gradually with more content after students gain a level of interest to help them endure the hardships of effortful and constructive learning. Simulation, on the other hand, provides a dynamic medium to test the model's predictions, break it into its constitutive parts to run various what-if scenarios, make changes to them if necessary, and put pieces of the puzzle together inductively to come up with a revised model. This kind of iterative and stepwise progression is consistent with psychology of optimal learning which suggests balancing skills and challenges.³ Anyone who learns in this iterative cycle of inductive and deductive reasoning would, in fact, be practicing the craft of scientists.

Measuring the impact of M&S on generating awareness of and appreciation for abstraction and decomposition skills, particularly in their relation to programming, needs further study. A question would be: Once learners

Anyone who learns in this iterative cycle of inductive and deductive reasoning would, in fact, be practicing the craft of scientists.

gain experience and fun creating artifacts (for example, models or videogames) with M&S, could this help them develop an interest to look for mathematical, computational and scientific principles under its hood? Some afterschool studies report encouraging results in teaching students textual programming in the process of creating videogames that connect to K-12 math and science learning outcomes.²² A quasi-experimental study of ours reports^{32,33} similar preliminary findings, as briefly explained next.

Annually, 50 teachers taught math and science topics using M&S tools during formal instruction. Teams of four students selected by each teacher received additional afterschool instruction from college faculty on mathematical principles of modeling (that is, new = old + change) as well as basic programming (in Excel and Python) to construct hands-on simulations. A panel of experts scored team projects and coded narratives to find common themes.⁵ According to these, hands-on modeling helped students realize the virtue of decomposition in problem solving, because finer decomposition led to more accurate answers. Other emerging themes included observations that textual programming provided better control of the decomposition (and desired accuracy) as well as easier coding (for example, via a simple loop). Finally, since computation of change in position, velocity, and acceleration necessitated a scientific formula to compute acting forces, this appeared to help students link computing and natural sciences. According to student interviews, it motivated them to plan on taking science and computing courses in later years. Follow-on quantitative data supported these anecdotal findings.³³ For example, while no physics courses were offered before in any of the 13 high schools of the urban school district, they became part of curricular offerings in two of them. The number of students taking general physics in the suburban high school increased by 50%. Also, the afterschool program led to design of a new computing course in one of the urban high schools, drawing high enrollment for three years until the teacher took a lucrative job in industry.

Conclusion

An interdisciplinary perspective on the cognitive essence of CT has been presented here based on the distributive and associative characteristics of information storage, retrieval, and processing by a network of neurons whose communication for searching, sorting, and analogies is driven by neural connectivity, richness of cues, a trade-off between simplification and elaboration, and a natural tendency to minimize energy usage. This broad approach might help clear some of the trouble spots with CT while putting it on a higher pedestal through a link to cognitive competencies involved in science and engineering.

Everyone cognitively benefits from CT by the virtue of having a computational mind. All we need is to help them use it in a more systematic way in their lives and professions. Since M&S facilitates an iterative and cyclical process of deductive and inductive reasoning, it could be used to teach novices not only critical CT skills (for example, abstraction and decomposition) but also ST and ET skills, including formation and change of hypothesis, concepts, designs, and models. While these are no different than cognitive processes of ordinary thinking,²⁶ not everyone uses them as consistently, frequently, and methodologically as computer scientists, natural scientists and engineers. The good news is they can be improved later through training and education.

CT's universal value is far beyond its relation to cognition. I argue that all heterogeneous stuff behaves computationally, regardless of what drives it. And, iterative and cyclical form of such behavior appears to be the essence of natural dynamism of all discrete forms. M&S is such a pattern, and putting computation in this fashion at the heart of natural sciences provides an opportunity to claim that computer science deals with natural phenomena, not artificial (digital). The computational revolution started by Turing may eventually be how our knowledge can come together to make more sense of our world.

One of the calls for action here for the CS community is to put more emphasis on M&S as a crucial part of student practice and education. This may help pave the way to teach

The computational revolution started by Turing may eventually be how our knowledge can come together to make more sense of our world.

computing principles to non-CS students.¹² Furthermore, while educational researchers have done a good job of measuring the impact of M&S on learning, a focus by the CS community can help generate interest among educational researchers to do similar research by measuring M&S's impact on conceptual change, abstraction, decomposition, and metacognitive skills, particularly in relation to CT and programming education. The second call is that prior to teaching students electronic CT skills, we need to teach them a habit of conceptual change through iterative and cyclical practices of inductive and deductive reasoning. Besides M&S tools, researchers should explore other modular and scalable design toys as well as reading and writing practices to offer similar CT practices. ■

References

1. Aho, A. Computation and computational thinking. *The Computer Journal* 55, 7 (Jul. 2012), 832–835.
2. Armoni, M. On teaching abstraction to computer science novices. *J. Comp in Math & Science Teaching* 32, 3 (Mar. 2013), 265–284.
3. Bransford, J., Brown, A., and Cocking, R. *How People Learn*. National Academy Press, Washington, D.C., 2000.
4. Brown, P., Roediger, H., and McDaniel, M. *Make it Stick*. Belknap Press of Harvard, 2014.
5. Creswell, J.W. *Educational Research*. 4th Edition. Pearson Education, Inc., 2012.
6. Denning, P. Remaining trouble spots with computational thinking. *Commun. ACM* 60, 6 (June 2017), 33–39.
7. Dunbar, K. and Klahr, D. Scientific thinking and reasoning. In K. Holyoak and R. Morrison, Eds., *The Oxford Handbook of Thinking and Reasoning*. Oxford University Press, London, 2012, 701–718.
8. Evans, J. and Frankish, K. *In Two Minds: Dual Processes and Beyond*. Oxford University Press, Oxford, 2009.
9. Findlay, S.D. and Thagard, P. How parts make up wholes. *Frontiers in Physiology* 3, 455 (2012).
10. Goleman, D. *Emotional Intelligence*. Bantam Dell, New York, 2006.
11. Grover, S. and Pea, R. Computational thinking:

A review of the state of the field. *Educational Researcher* 42, 1 (Jan. 2013), 38–43.

12. Guzdial, M. Paving the way for computational thinking. *Commun. ACM* 51, 8 (Aug. 2008), 25–27.
13. Hawkins, J. *On Intelligence*. Times Books, New York, 2004.
14. Hebb, D. *The Organization of Behavior*. Wiley, New York, 1949.
15. Kant, I. *The Critique of Pure Reason*. (J.M.D. Meiklejohn, Trans.). eBook@Adelaide, The University of Adelaide Library, Australia, 1787.
16. Kirschner, P.A., Sweller, J., and Clark, R.E. Why minimal guidance during instruction does not work. *Educational Psychologist* 41, 2 (Feb. 2006), 75–86.
17. Koehler, M., Shin, T., and Mishra, P. How do we measure TPAC? In R.N. Ronau, C.R. Rakes, and M.L. Niess, Eds., *Educational Technology, Teacher Knowledge, and Classroom Impact* IGI Global, Hershey, PA, 2012, 16–31.
18. Kuhn, T. *The Structure of Scientific Revolutions*. U. Chicago Press, Chicago, 1962.
19. Montague, R. *How We Make Decisions*. Plume Books, New York, 2006.
20. Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, New York, 1980.
21. Rutten, N., van Joolingen, W.R., and van der Veen, J.T. The learning effects of computer simulations in science education. *Computers & Education* 58, 1 (Jan. 2012), 136–153.
22. Schanzer, E., Fister, K. and Krishnamurthi, S. Bootstrapping beyond programming in after-school computer science. SPLASH Education Symposium, Claremont, CA., 2013.
23. Smetana, L.K. and Bell, R.L. Computer simulations to support science instruction and learning. *Int. J. Science Education* 34, 9 (Sept. 2012), 1337–1370.
24. Sun, R. *Duality of Mind*. Lawrence Erlbaum Associates, Mahwah, NJ, 2002.
25. Tenenbaum, J.B., Kemp, C., Griffiths, T.L., and Goodman, N.D. How to grow a mind: Statistics, structure, and abstraction. *Science* 331, (2011), 1279–1285.
26. Thagard, P. *The Cognitive Science of Science*. The MIT Press, Cambridge, MA, 2012.
27. Turing, A.M. *On Computable Numbers, with an Application to the Entscheidungs problem*. In *Proceedings of the London Mathematical Society* 2, 42 (1937), 230–265.
28. Wing, J.M. Computational thinking—What and why? *The Link Magazine* (Mar. 06, 2011).
29. Wing, J.M. Computational thinking. *Commun. ACM* 49, 3 (Mar. 2006), 33–35.
30. Yaşar, O. Modeling and simulation: How everything seems to form and grow. *Comp. in Sci. and Eng.* 19, 1 (Jan. 2017), 74–77.
31. Yaşar, O., Maliekal, J., Veronesi, P. and Little, L. The essence of scientific and engineering thinking and tools to promote it. In *Proceedings of the American Society of Engineering Education Annual Conference*, 2017.
32. Yaşar, O. and Maliekal, J. Computational pedagogy. *Comp. in Sci. and Eng.* 16, 3 (Mar. 2014), 78–88.
33. Yaşar, O., Maliekal, J., Veronesi, P., and Little, L. An interdisciplinary approach to professional development of math, science and technology teachers. *Comp. in Math & Sci. Teaching* 33, 3 (Mar. 2014), 349–374.
34. Yaşar, O. and Landau, R. Elements of computational science and engineering education. *SIAM Review* 45, 4 (2003), 787–805.

Osman Yaşar (oyasar@brockport.edu) is an Empire Innovation Professor at the State University of New York, The College at Brockport.

The author thanks Jose Maliekal, Peter Veronesi, and Leigh Little for their collaboration and comments; also grateful to Pinar Yaşar, who helped form the epistemological perspective expressed here.

Support was received from the National Science Foundation via grants EHR 0226962, DRL 0410509, DRL 0540824, DRL 0733864, DRL 1614847, and DUE 1136332. The author's views expressed in this Viewpoint are not necessarily those of his employer or the U.S. federal government.

Copyright held by author.

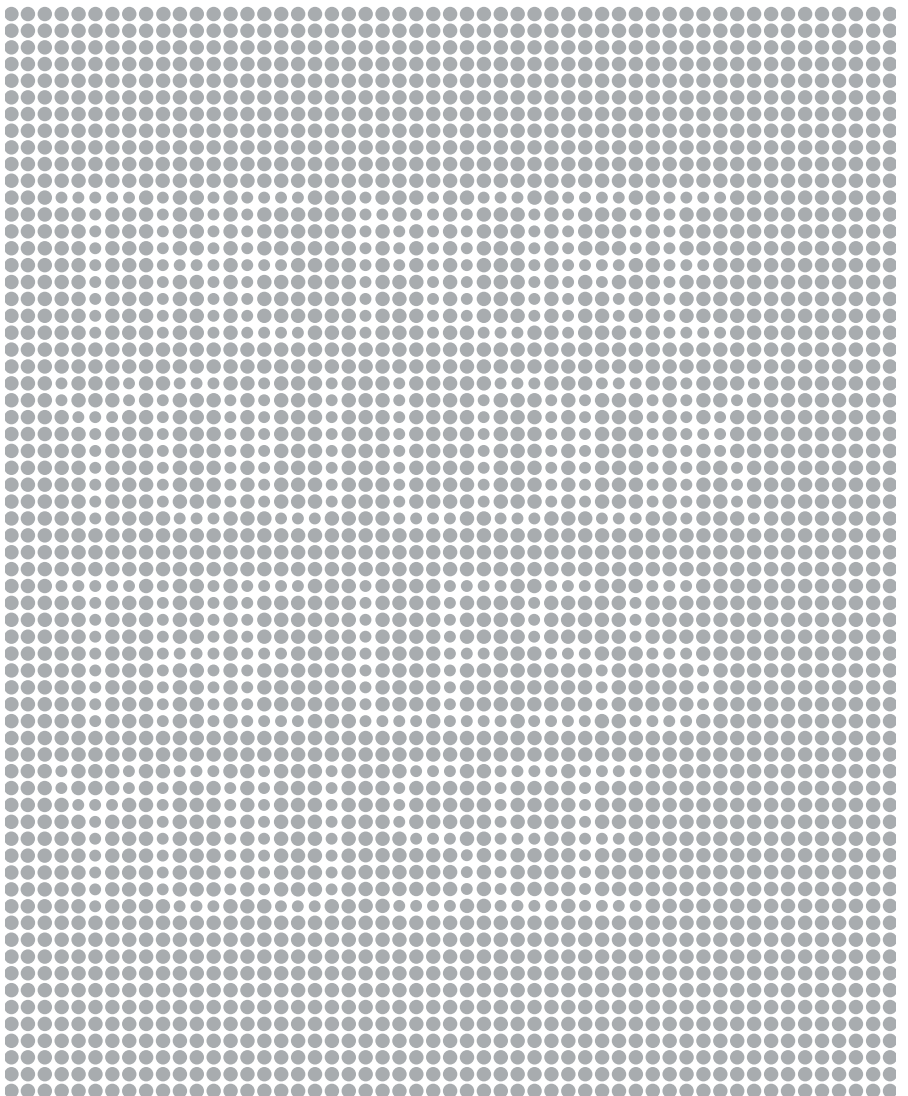
Viewpoint

The Case for Disappearing Cyber Security

A proposal for keeping cyber security both out of sight and out of mind for end users.

IN MAY 2017, WannaCry ransomware rapidly proliferated around the Internet, despite availability of a patch released by Microsoft in March. This is simply one of the most recent and notable attacks exploiting known flaws—there is a constant barrage of attacks, large and small. Although cyber security is more complicated than a simple failure to patch end systems, analysis of cyber security incidents has consistently shown that a failure to apply patches is one of the leading enablers of successful attacks.

We have reached a point in the evolution of cyber security where hands-off, behind-the-scenes cyber defense should be the norm. Clearly, the best solution would be to deploy less-vulnerable systems. This is a topic that has received great attention for approximately five decades, but developers continue to resist using tools and techniques that have been shown to be effective, such as code minimization, employing formal development methods, and using type-safe languages. Additionally, consumers are widely believed to be reluctant to accept the software limitations and increased costs that result from some of these more secure development practices. Those issues, coupled with the vast amount of legacy code in place and being reused, have meant that better security is often, at best, an “add-



on” rather than “built-in” function. Patching and configuration changes will be required indefinitely to keep the current infrastructure at least moderately secure.

Many end users today are not taking the deliberate actions required to protect themselves. For many decades, we have created increasingly effective tools and techniques for protecting users and systems, including limited automatic updates.³ Despite active and valiant work at adoption and usability, it is apparent many users cannot or will not avail themselves of appropriate cyber security options. Many of today’s systems were created with many low-level options by those who understood how they worked. Most end users today do not understand the underlying risks and results of their choices for those options. Those who are better informed may believe (sometimes, mistakenly) they understand all the issues and do not want to give up any options. In one study, even well-intentioned users with high engagement in security made choices leading to ineffective security.² In a 2008 *Communications* article, “The Psychology of Security,” Ryan West concluded that “The ideal security user experience for most users would be none at all,” but that users were, at that time, in the control loop by necessity.¹⁰ It is time to reconsider the amount of control we require of users across the spectrum of experience.

We suggest an answer for a large percentage of end users would be to make the security aspects and interactions of today disappear from their view. That is, we ought to continue the pursuit of secure system development, patching, configuration, and operation as required, but do so without any explicit or necessary action of the users. Security should become transparent and all but “disappear” from those users’ consciousness.

Users seem to be accepting one form of out-of-sight security already: automatic patch updates. Automatic software patches are now a common feature in desktop software, including Windows 10 and Google Chrome. It is certainly conceivable that other vendors and service providers could employ additional defenses needing updating without the express permission

It is time to reconsider the amount of control we require of users across the spectrum of experience.

and awareness of users. It is commonplace now for users to agree to (but not read) end user license agreements (EULAs), some of which include language about limited out-of-sight changes.

Consider phishing as another example different from patching. Social engineering attacks executed by email have existed since the deployment of MIME-enabled email in the 1990s and remain a significant threat to this day. For decades, educational campaigns have attempted to raise awareness, with mixed success. In an ideal and sufficiently advanced world, phishing messages would never reach the end user and require the user’s attention or judgment. Today, email providers such as Gmail employ machine learning and self-reporting in an attempt to flag suspicious messages and warn users. However, no matter how good those systems become there are false positives and users may still suffer consequences as they browse their “junk” folders looking for misclassified email messages. Incidents could be further reduced if the bad items known with extreme certainty were never even visible in the “junk” repositories of most users. This would make part of the protection “invisible” in the typical case.

An additional factor that supports more automated, invisible security is that online safety is closely tied to where users get their security information and advice. Too few users know where to obtain understandable, authoritative, and actionable advice. Furthermore, as threats (and systems) evolve, users need to update their security knowledge more often than they

do. A 2016 research study examined reasons why subjects chose not to take a secure action even after receiving information recommending the behavior.⁶ Among the 43% that rejected at least one action from the choices of installing antivirus, updating, and deploying two-factor authentication, the most common reasons were inconvenience and advice with “too much marketing material.” Not only do people reject advice they think is unduly biased by vendors, but we have also seen far too many incidents with people who think they understand the risks and mechanisms better than they do. In both cases, the end users fail to follow best practices and good advice for patching, protections, firewalls, authentication, backups, and so forth. Security for both groups could be improved if the appropriate actions were executed automatically, without the need for user intervention or awareness.

Complexity for end users is an enemy of good security. The base case of applying security updates without delay or modification is all that a majority of end users need. Out-of-sight security could be simpler for the user if it reduced or eliminated complex security configurations and user interactions. If the security evolved along with the systems, users would not need to master the accompanying new terms and concepts.

While home and enterprise users will benefit from out-of-sight security by default, enterprises may continue to require more granular control. In many large environments, automatic, forced patching and control are already the norm for systems administered by organizations. Centralized control results in a more uniform application of necessary updates, and obviates the need for user involvement. It also simplifies issues of recovery from failures, and, when necessary, forensic analysis. Here, the same concept of automated, invisible security is applied, but at an organizational level, consistent with local needs.

Users may look for ways around automated security mechanisms.⁷ A small population of technical users will be uncomfortable about surrendering control of their security options;

some of these users value theoretical considerations of control over practical security. Thus, it may be necessary to allow more control for those users with greater experience or special needs. The default start for these users would be the “invisible” security, with non-obvious options requiring explicit acknowledgment of risk, and perhaps a certain level of technical skill to access. This would be the cyber equivalent of “No user serviceable parts inside” warnings on many consumer electronics.

We acknowledge there are risks with invisible security that must be considered. Automated updates could interfere or break other software, or worse.^a We will need mechanisms to verify that the invisible security is enabled and working as it should. We also recognize there are circumstances where patches must be certified in some way—including having patched systems meet performance and safety standards, such as those present for industrial controls, medical uses, and national security. In these cases, exceptions may need to be made to delay patching and support necessary testing. (This begs the questions of why those critical applications are using commodity software that may be prone to serious errors, and why they are configured in such a way that their safe operation would necessitate such patches.) Generally, these special cases make up a minority of deployed systems, and exempting them from automated patching would not negate the benefits of quickly fixing problems in all the rest. We also note the serious issue of securing legacy, unsupported, and unlicensed systems will remain a challenge, but it is made neither better nor worse by behind-the-scenes security.

Research will be necessary to determine whether or not users feel more secure—and if they actually are more secure—when cyber security defenses are invisible. That requires understanding what is meant by “security” in different contexts and with different types of security controls (such as patching, anti-malware, anti-phishing). It has been repeatedly noted that

Complexity for end users is an enemy of good security.

adequate security is relative to current environments and threats. For many end users, good security simply means their privacy is protected, even if nothing else is. Thus, security is often seen as both a reality and a feeling. Bruce Schneier, in particular, has criticized “security theater” but he acknowledges that “a bit of well-placed security theater might be exactly what we need to both be and feel more secure.”⁸ Cyber security professionals can learn from examples in other domains of visible versus non-visible security implementations. For instance, visible policing is an approach to security that places uniformed police officers in public to deter crime and reassure citizens. Research shows mixed results, including cases of increased crime and fear of crime after increasingly visible police presence.⁴

Clearly, those of us involved with equipping the world with advanced computation have some ethical obligations to make that computation safe.¹ The security community should strive for default security without explicit user interaction. The challenge is one of balance: How do we continue to provide appropriate autonomy and freedom to computer users while also protecting them? What is an appropriate level of residual risk to allow? We believe these (and related) questions should be considered and discussed, now, to enable development of a new climate for cyber security (and thus, privacy protection, which usually depends on good security) rather than continue to apply patchwork protections as marketing opportunities. We suggest that part of the solution is to move away from solutions that default to settings for the users who need the most options and choices, and instead automate security as the new default.

The cyber security community has succeeded in substantially advancing the field of resilient and trustworthy systems. Furthermore, research and development in security usability have made better security available to more users. The continued state of poor security adoption and practice, interacting with basic human nature, requires us to consider the next step of offering automated, behind-the-scenes cyber security as widely as possible. Continued work is necessary to refine the balance of control between human and machine, similar to the conversations around machine learning and artificial intelligence. If anything, those fields will require good cyber security to achieve their full promise. We believe it is time to consider a new approach, as we have outlined in this Viewpoint. **C**

References

1. ACM Committee on Professional Ethics. *2018 ACM Code of Ethics and Professional Conduct: Draft 2*; <https://ethics.acm.org/2018-code-draft-2>
2. Forget, A. Pearman, S., Thomas, J. et al. Do or do not, there is no try: User engagement may not improve security outcomes. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*. (USENIX Association, Denver, CO), 2016, 97–111.
3. Frei, S., Duebendorfer, T. and Plattner, B. Firefox (in) security update dynamics exposed. *ACM SIGCOMM Comput. Commun. Rev.* 39, 1 (Jan. 2009), 16–22.
4. Millie, A. and Herrington, V. Bridging the gap: Understanding reassurance policing. *The Howard Journal* 44, 1 (Feb. 2005), 41–56.
5. Nachenberg, C. *The Florentine Deception*. Open Road Media Mystery & Thriller, 2015. <http://florentinedeception.weebly.com>
6. Redmiles, E., Malone, A. and Mazurek, M. I think they're trying to tell me something: Advice sources and selection for digital security. *IEEE Symposium on Security and Privacy*, 2016.
7. Sasse, M.A., Smith, M., Herley, C., Lipford, H., and Vaniea, K. Debunking security-usability tradeoff myths. *IEEE Security & Privacy* 14, 5 (May 2016), 33–39.
8. Schneier, B. The psychology of security. In *Proceedings of the Cryptology in Africa Ist International Conference on Progress in Cryptology (AFRICACRYPT'08)*, Serge Vaudenay, Ed. Springer-Verlag, Berlin, Heidelberg, 2008, 50–79.
9. Wash, R., Rader, E., Vaniea, K. et al. Out of the loop: How automated software updates cause unintended security consequences. In *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*. USENIX Association, Berkeley, CA, 2014, 89–104.
10. West, R. The psychology of security: Why do good users make bad decisions? *Commun. ACM* 51, 4 (Apr. 2008), 34–40.

Josiah Dykstra (josiahdykstra@acm.org) is a cyber security researcher with the U.S. Department of Defense in Baltimore, MD, USA.

Eugene H. Spafford (spaf@acm.org) is a professor of computer science at Purdue University, West Lafayette, IN, USA.

The views and opinions expressed in this Viewpoint are those of the authors and do not necessarily reflect those of the U.S. government, the U.S. Department of Defense, or Purdue University.

Copyright held by author.

a This topic is explored in a recent work of fiction by a senior security professional.⁵



Advertise with ACM!

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

Request a media kit with specifications and pricing:



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

A 21st-century Model for Teaching Computer Science

Reaching Over 1 Million Learners Worldwide

Textbooks



Studio-Produced Lectures



informit.com/sedgewick

 **Pearson**
Addison-Wesley

Free Online Content

intros.cs.princeton.edu
algs4.cs.princeton.edu

MOOCs





Computer Science: Algorithms, Theory, & Machines
coursera.org/learn/cs-algorithms-theory-machines



Algorithms, Part I
coursera.org/learn/algorithms-part1



Algorithms, Part II
coursera.org/learn/algorithms-part2



12 rising-stars from different subfields of multimedia research discuss the challenges and state-of-the-art developments of their prospective research areas in a general manner to the broad community.

The field of multimedia is unique in offering a rich and dynamic forum for researchers from “traditional” fields to collaborate and develop new solutions and knowledge that transcend the boundaries of individual disciplines. Despite the prolific research activities and outcomes, however, few efforts have been made to develop books that serve as an introduction to the rich spectrum of topics covered by this broad field. A few books are available that either focus on specific subfields or basic background in multimedia. Tutorial-style materials covering the active topics being pursued by the leading researchers at frontiers of the field are currently lacking...**UNTIL NOW.**



ISBN: 978-1-970001-044 DOI: 10.1145/3122865

<http://books.acm.org>

<http://www.morganclaypoolpublishers.com/chang>

Your computer is not a fast PDP-11.

BY DAVID CHISNALL

C Is Not a Low-Level Language

IN THE WAKE of the recent Meltdown and Spectre vulnerabilities, it is worth spending some time looking at root causes. Both of these vulnerabilities involved processors speculatively executing instructions past some kind of access check and allowing the attacker to observe the results via a side channel. The features that led to these vulnerabilities, along with several others, were added to let C programmers continue to believe they were programming in a low-level language, when this hasn't been the case for decades.

Processor vendors are not alone in this. Those of us working on C/C++ compilers have also participated.

Computer science pioneer Alan Perlis defined low-level languages this way:

“A programming language is low level when its programs require attention to the irrelevant.”⁵

While, yes, this definition applies to C, it does not capture what people desire in a low-level language. Various attributes cause people to regard a language as low level. Think of programming languages as belonging

on a continuum, with assembly at one end and the interface to the Starship *Enterprise's* computer at the other. Low-level languages are “close to the metal,” whereas high-level languages are closer to how humans think.

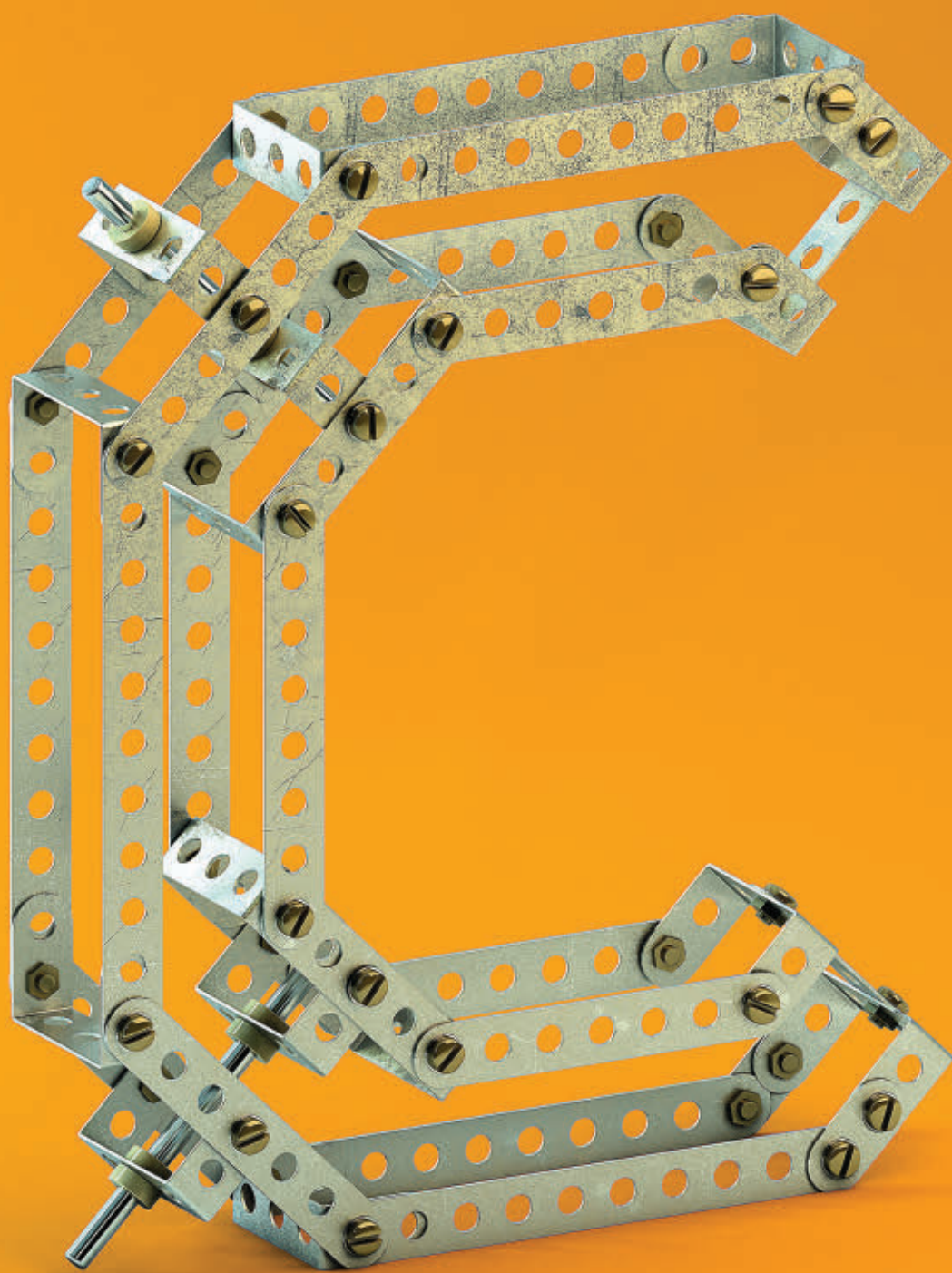
For a language to be “close to the metal,” it must provide an abstract machine that maps easily to the abstractions exposed by the target platform. It's easy to argue that C was a low-level language for the PDP-11. They both described a model in which programs executed sequentially, in which memory was a flat space, and even the pre- and post-increment operators cleanly lined up with the PDP-11 addressing modes.

Fast PDP-11 Emulators

The root cause of the Spectre and Meltdown vulnerabilities was that processor architects were trying to build not just fast processors, but fast processors that expose the same abstract machine as a PDP-11. This is essential because it allows C programmers to continue in the belief that their language is close to the underlying hardware.

C code provides a mostly serial abstract machine (until C11, an entirely serial machine if nonstandard vendor extensions were excluded). Creating a new thread is a library operation known to be expensive, so processors wishing to keep their execution units busy running C code rely on ILP (instruction-level parallelism). They inspect adjacent operations and issue independent ones in parallel. This adds a significant amount of complexity (and power consumption) to allow programmers to write mostly sequential code. In contrast, GPUs achieve very high performance without any of this logic, at the expense of requiring explicitly parallel programs.

The quest for high ILP was the direct cause of Spectre and Meltdown. A modern Intel processor has up to 180 instructions in flight at a time (in stark contrast to a sequential C abstract machine, which expects each operation to complete before the next one begins). A typical heuristic for C code



is that there is a branch, on average, every seven instructions. If you wish to keep such a pipeline full from a single thread, then you must guess the targets of the next 25 branches. Again, this adds complexity; it also means that an incorrect guess results in work being done and then discarded, which is not ideal for power consumption. This discarded work has visible side effects, which the Spectre and Meltdown attacks could exploit.

On a modern high-end core, the register rename engine is one of the largest consumers of die area and power. To make matters worse, it cannot be turned off or power gated while any instructions are running, which makes it inconvenient in a dark silicon era when transistors are cheap but powered transistors are an expensive resource. This unit is conspicuously absent on GPUs, where parallelism again comes from multiple threads rather than trying to extract instruction-level parallelism from intrinsically scalar code. If instructions do not have dependencies that must be reordered, then register renaming is not necessary.

Consider another core part of the C abstract machine's memory model: flat memory. This has not been true for more than two decades. A modern processor often has three levels of cache in between registers and main memory, which attempt to hide latency.

The cache is, as its name implies, hidden from the programmer and so is not visible to C. Efficient use of the cache is one of the most important ways of making code run quickly on a modern processor, yet this is completely hidden by the abstract machine, and programmers must rely on knowing implementation details of the cache (for example, two values that are 64-byte-aligned may end up in the same cache line) to write efficient code.

Optimizing C

One of the common attributes ascribed to low-level languages is that they are fast. In particular, they should be easy to translate into fast code without requiring a particularly complex compiler. The argument that a sufficiently smart compiler can make a language fast is one that C



The root cause of the Spectre and Meltdown vulnerabilities was that processor architects were trying to build not just fast processors, but fast processors that expose the same abstract machine as a PDP-11.



proponents often dismiss when talking about other languages.

Unfortunately, simple translation providing fast code is not true for C. In spite of the heroic efforts that processor architects invest in trying to design chips that can run C code fast, the levels of performance expected by C programmers are achieved only as a result of incredibly complex compiler transforms. The Clang compiler, including the relevant parts of LLVM, is around two million lines of code. Even just counting the analysis and transform passes required to make C run quickly adds up to almost 200,000 lines (excluding comments and blank lines).

For example, in C, processing a large amount of data means writing a loop that processes each element sequentially. To run this optimally on a modern CPU, the compiler must first determine that the loop iterations are independent. The C `restrict` keyword can help here. It guarantees that writes through one pointer do not interfere with reads via another (or if they do, that the programmer is happy for the program to give unexpected results). This information is far more limited than in a language such as Fortran, which is a big part of the reason that C has failed to displace Fortran in high-performance computing.

Once the compiler has determined that loop iterations are independent, then the next step is to attempt to vectorize the result, because modern processors get four to eight times the throughput in vector code than they achieve in scalar code. A low-level language for such processors would have native vector types of arbitrary lengths. LLVM IR (intermediate representation) has precisely this, because it is always easier to split a large vector operation into smaller ones than to construct larger vector operations.

Optimizers at this point must fight the C memory layout guarantees. C guarantees that structures with the same prefix can be used interchangeably, and it exposes the offset of structure fields into the language. This means that a compiler is not free to reorder fields or insert padding to improve vectorization (for example, transforming a structure of arrays into an

array of structures or vice versa). That is not necessarily a problem for a low-level language, where fine-grained control over data structure layout is a feature, but it does make it more difficult to make C fast.

C also requires padding at the end of a structure because it guarantees no padding in arrays. Padding is a particularly complex part of the C specification and interacts poorly with other parts of the language. For example, you must be able to compare two `structs` using a type-oblivious comparison (for example, `memcmp`), so a copy of a `struct` must retain its padding. In some experimentation, a noticeable amount of total runtime on some workloads was found spent in copying padding (which is often awkwardly sized and aligned).

Consider two of the core optimizations that a C compiler performs: SROA (scalar replacement of aggregates) and loop unswitching. SROA attempts to replace `structs` (and arrays with fixed lengths) with individual variables. This then allows the compiler to treat accesses as independent and elide operations entirely if it can prove that the results are never visible. This has the side effect of deleting padding in some cases but not others.

The second optimization, loop unswitching, transforms a loop containing a conditional into a conditional with a loop in both paths. This changes flow control, contradicting the idea that a programmer knows what code will execute when low-level language code runs. It can also cause significant problems with C's notions of unspecified values and undefined behavior.

In C, a read from an uninitialized variable is an unspecified value and is allowed to be any value each time it is read. This is important, because it allows behavior such as lazy recycling of pages: for example, on FreeBSD the `malloc` implementation informs the operating system that pages are currently unused, and the operating system uses the first write to a page as the hint that this is no longer true. A read to newly `malloced` memory may initially read the old value; then the operating system may reuse the underlying physical page; and then on the next write to a different location in

the page replace it with a newly zeroed page. The second read from the same location will then give a zero value.

If an unspecified value for flow control is used (for example, using it as the condition in an `if` statement), then the result is undefined behavior: anything is allowed to happen. Consider the loop-unswitching optimization, this time in the case where the loop ends up being executed zero times. In the original version, the entire body of the loop is dead code. In the unswitched version, there is now a branch on the variable, which may be uninitialized. Some dead code has been transformed into undefined behavior. This is just one of many optimizations that a close investigation of the C semantics shows to be unsound.

In summary, it is possible to make C code run quickly but only by spending thousands of person-years building a sufficiently smart compiler—and even then, only if you violate some of the language rules. Compiler writers let C programmers pretend that they are writing code that is “close to the metal” but must then generate machine code that has very different behavior if they want C programmers to keep believing they are using a fast language.

Understanding C

One of the key attributes of a low-level language is that programmers can easily understand how the language's abstract machine maps to the underlying physical machine. This was certainly true on the PDP-11, where each C expression mapped trivially to one or two instructions. Similarly, the compiler performed a straightforward lowering of local variables to stack slots and mapped primitive types to things that the PDP-11 could operate on natively.

Since then, implementations of C have had to become increasingly complex to maintain the illusion that C maps easily to the underlying hardware and gives fast code. A 2015 survey of C programmers, compiler writers, and standards committee members raised several issues about the comprehensibility of C.³ For example, C permits an implementation to insert padding into structures (but not into

arrays) to ensure all fields have a useful alignment for the target. If you zero a structure and then set some of the fields, will the padding bits all be zero? According to the results of the survey, 36% were sure that they would be, and 29% didn't know. Depending on the compiler (and optimization level), it may or may not be.

This is a fairly trivial example, yet a significant proportion of programmers either believes the wrong thing or is not sure. When you introduce pointers, the semantics of C become a lot more confusing. The BCPL model was fairly simple: values are words. Each word is either some data or the address of some data. Memory is a flat array of storage cells indexed by address.

The C model, in contrast, was intended to allow implementation on a variety of targets, including segmented architectures (where a pointer might be a segment ID and an offset) and even garbage-collected virtual machines. The C specification is careful to restrict valid operations on pointers to avoid problems for such systems. The response to Defect Report 260¹ included the notion of *pointer provenance* in the definition of pointer:

Implementations are permitted to track the origins of a bit pattern and treat those representing an indeterminate value as distinct from those representing a determined value. They may also treat pointers based on different origins as distinct even though they are bitwise identical.

Unfortunately, the word *provenance* does not appear in the C11 specification at all, so it is up to compiler writers to decide what it means. GNU Compiler Collection (GCC) and Clang, for example, differ on whether a pointer that is converted to an integer and back retains its provenance through the casts. Compilers are free to determine that two pointers to different `malloc` results or stack allocations always compare as not-equal, even when a bitwise comparison of the pointers may show them to describe the same address.

These misunderstandings are not purely academic in nature. For ex-

ample, security vulnerabilities have been observed from signed integer overflow (undefined behavior in C) and from code that dereferenced a pointer before a null check, indicating to the compiler that the pointer could not be null because dereferencing a null pointer is undefined behavior in C and therefore can be assumed not to happen (CVE-2009-1897).

In light of such issues, it is difficult to argue that a programmer can be expected to understand exactly how a C program will map to an underlying architecture.

Imagining a Non-C Processor

The proposed fixes for Spectre and Meltdown impose significant performance penalties, largely offsetting the advances in microarchitecture in the past decade. Perhaps it is time to stop trying to make C code fast and instead think about what programming models would look like on a processor designed to be fast.

We have a number of examples of designs that have not focused on traditional C code to provide some inspiration. For example, highly multi-threaded chips, such as Sun/Oracle's UltraSPARC Tx series, do not require as much cache to keep their execution units full. Research processors² have extended this concept to very large numbers of hardware-scheduled threads. The key idea behind these designs is that with enough high-level parallelism, you can suspend the threads that are waiting for data from memory and fill your execution units with instructions from others. The problem with such designs is that C programs tend to have few busy threads.

ARM's Scalar Vector Extensions (SVE)—and similar work from Berkeley⁴—provides another glimpse at a better interface between program and hardware. Conventional vector units expose fixed-sized vector operations and expect the compiler to try to map the algorithm to the available unit size. In contrast, the SVE interface expects the programmer to describe the degree of parallelism available and relies on the hardware to map it down to the available number of execution units. Using this from C is complex, because the autovector-

izer must infer the available parallelism from loop structures. Generating code for it from a functional-style map operation is trivial: the length of the mapped array is the degree of available parallelism.

Caches are large, but their size isn't the only reason for their complexity. The *cache coherency protocol* is one of the hardest parts of a modern CPU to make both fast and correct. Most of the complexity involved comes from supporting a language in which data is expected to be both shared and mutable as a matter of course. Consider in contrast an Erlang-style abstract machine, where every object is either thread-local or immutable (Erlang has a simplification of even this, where there is only one mutable object per thread). A cache coherency protocol for such a system would have two cases: mutable or shared. A software thread being migrated to a different processor would need its cache explicitly invalidated, but that is a relatively uncommon operation.

Immutable objects can simplify caches even more, as well as making several operations even cheaper. Sun Labs' Project Maxwell noted that the objects in the cache and the objects that would be allocated in a young generation are almost the same set. If objects are dead before they need to be evicted from the cache, then never writing them back to main memory can save a lot of power. Project Maxwell proposed a young-generation garbage collector (and allocator) that would run in the cache and allow memory to be recycled quickly. With immutable objects on the heap and a mutable stack, a garbage collector becomes a very simple state machine that is trivial to implement in hardware and allows for more efficient use of a relatively small cache.

A processor designed purely for speed, not for a compromise between speed and C support, would likely support large numbers of threads, have wide vector units, and have a much simpler memory model. Running C code on such a system would be problematic, so, given the large amount of legacy C code in the world, it would not likely be a commercial success.

There is a common myth in software

development that parallel programming is difficult. This would come as a surprise to Alan Kay, who was able to teach an actor-model language to young children, with which they wrote working programs with more than 200 threads. It comes as a surprise to Erlang programmers, who commonly write programs with thousands of parallel components. It is more accurate to say that parallel programming in a language with a C-like abstract machine is difficult, and given the prevalence of parallel hardware, from multicore CPUs to many-core GPUs, that is just another way of saying that C doesn't map to modern hardware very well. 

Related articles on queue.acm.org

The Challenge of Cross-Language Interoperability

David Chisnall

<https://queue.acm.org/detail.cfm?id=2543971>

Finding More than One Worm in the Apple

Mike Bland

<https://queue.acm.org/detail.cfm?id=2620662>

Coding for the Code

Friedrich Steimann and Thomas Kühne

<https://queue.acm.org/detail.cfm?id=1113336>

References

1. C Defect Report 260, 2004; http://www.open-std.org/jtc1/sc22/wg14/www/docs/dr_260.htm.
2. Chadwick, G.A. Communication centric, multi-core, fine-grained processor architecture. Technical Report 832. University of Cambridge, Computer Laboratory, 2013; <http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-832.pdf>.
3. Memarian, K., Matthiesen, J., Lingard, J., Nienhuis, K., Chisnall, D., Watson, R.N.M., and Sewell, P. Into the depths of C: Elaborating the de facto standards. In *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2016, 1–15; <http://dl.acm.org/authorize?N04455>.
4. Ou, A., Nguyen, Q., Lee, Y., and Asanović, K. A case for MVPs: Mixed-precision vector processors. In *Proceedings of the 2nd Intern'l Workshop on Parallelism in Mobile Platforms at the 41st Intern'l Symposium on Computer Architecture*. 2014.
5. Perlis, A. Epigrams on programming. *ACM SIGPLAN Notices* 17, 9 (1982).

David Chisnall is a researcher at the University of Cambridge, where he works on programming language design and implementation. He has authored books on Xen and the Objective-C and Go programming languages, and contributes to the LLVM, Clang, FreeBSD, GNUstep, and Étoilé open source projects.

Approved for public release; distribution is unlimited. Sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under contract FA8750-10-C-0237 ("CTSRD") as part of the DARPA CRASH research program. The views, opinions, and/or findings contained in this report are those of the authors and should not be interpreted as representing the official views or policies, either expressed or implied, of the Department of Defense or the U.S. government.

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



Think like an entrepreneur.

BY KATE MATSUDAIRA

How to Come Up with Great Ideas

“I would love to do a startup, but I don’t have any ideas.”

I started my career working in big companies but always dreamed of starting my own. I would read online forums and articles about successful entrepreneurs. I was enamored with the idea of doing a startup. The problem was I didn’t have any ideas.

Fast forward 10 years and I have so many ideas that choosing the right one is the challenge. I am constantly coming up with ideas and opportunities that could turn into a product, or a whole company. There is no shortage of things that I *could* do.

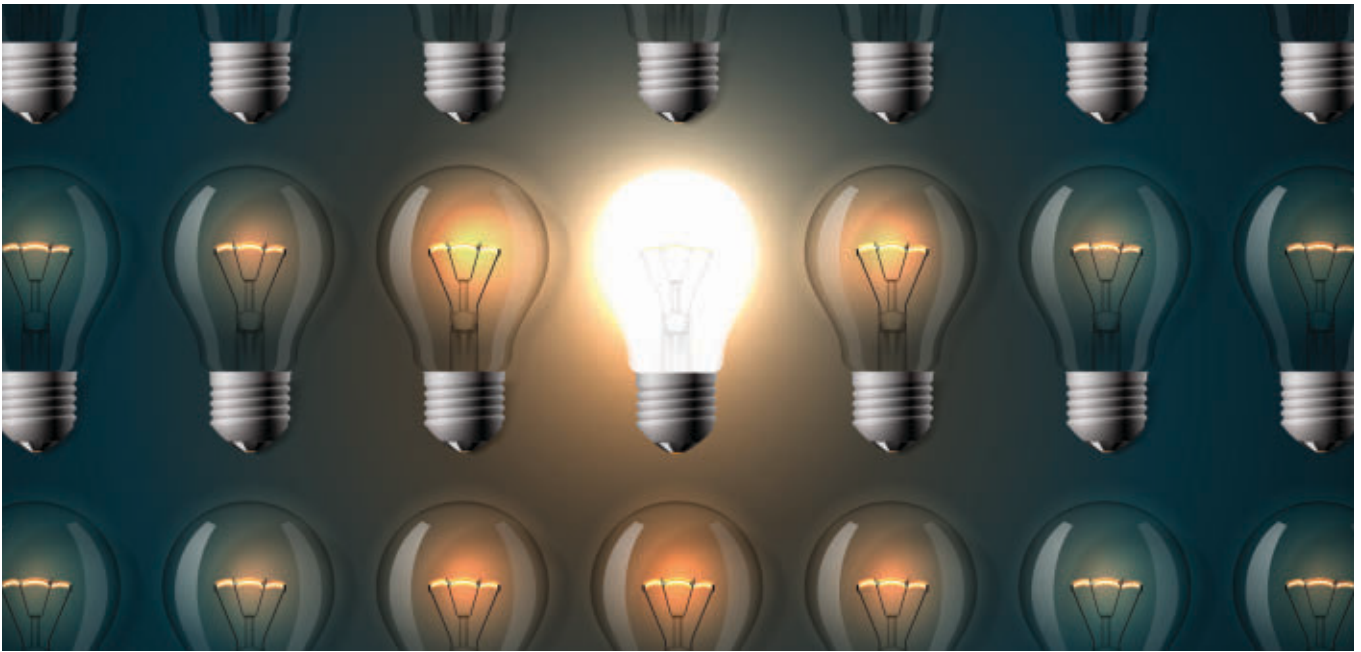
The key is you have to learn to think like an entrepreneur.

At this point, you might be thinking, “But I am a software engineer and happy in my role; why do I need to think like an entrepreneur?” Even if you don’t see yourself doing a startup with one idea, being able to come up with new ideas (especially good

ones!) will make you even better at your current job.

As a programmer, most of your time is probably spent understanding the technical nuances of what you are creating. As you grow in your career and experience, however, you are expected to contribute more than just code.

For example, if you work on a product, you may have the chance to collaborate with its designers to create a new customer experience. When there are bugs or issues, your time is spent diagnosing the symptoms to find root problems and exploring options that may serve as possible solutions. As you



build your expertise, you may even create your own modules, systems, or libraries to help solve problems.

There are many instances like these, where being creative and innovative and generating good ideas can help you contribute more to your company, profession, or community.

After working in a few startups, I have learned from people, books, and experience. Here are a few key pointers that may lead you to generate more great ideas.

Pay attention to friction. Whenever something doesn't work the way you think it should, pay attention and think about why. What could be changed that would make it better? For example, suppose there is a website you like to use, but when visiting the site, you always click on the wrong button. How would you change it? Would you relocate the button?

I think about this a lot when regarding my collection of travel mugs. Each one I buy has a serious flaw: the glass one chipped when I dropped it, the clear plastic one became stained after continued use, or the tops that are easy to open have a tendency to leak. By focusing on the points of friction, you can start thinking about how you would design one that addresses those problems.

Paying attention to how you compensate for missing functionality, or where you get frustrated using an item, will teach you to start focusing on problems that can be solved. As you

hone in on these problems over time, you will start to see opportunities that could be turned into new products or even companies.

What Makes Something Great?

When you use products you love, think about the features that delight you. By articulating what you like about a product, you can pick up patterns and gain a better understanding of motivation (which can then be applied to users of future products).

For example, I started using a workout app called Sweat, and I absolutely love it. Why?

- I like that it keeps me accountable tracking my progress.

Motivation: I am more likely to work out and stick with the program.

- I like that it shows a video of how to do each exercise.

Motivation: I know I am doing things correctly, so I won't get injured.

- I like that the workouts are around 30 minutes.

Motivation: I am busy, so the short duration allows me to fit exercise into my day.

When you understand what motivates users, you can help them reach their goals (easier, better, faster). It can also help you discover methods and patterns that can be applied to your next big idea and help you deliver more value to customers.

Brainstorm. If you want to have great ideas, the first step is to have lots

of ideas. In the *Ten Faces of Innovation*, Tom Kelley writes that when it comes to brainstorming, the focus should be on quantity of ideas rather than quality. This is because sometimes an idea that seems silly or obtuse can spark other ideas and paths of thinking.

One of the best ways to foster creative thinking and ideas is to brainstorm regularly. Choose a topic, and then either by yourself or with a group spend time coming up with ideas around that topic. When I get bored in meetings, sometimes I will spend a few minutes brainstorming or thinking about something that is on my mind.

Listen and observe. Recently I read Walter Isaacson's biography of *Leonardo da Vinci*, published in 2017. The book described the notebooks da Vinci kept; what struck me was how observant da Vinci was. He would study details and notice small things from the way hair curled (seen in his artwork), to the way water flowed (applied to his innovations on hydrodynamics), to the way people's faces change when they smile (seen in the *Mona Lisa*). By observing the world around him he was able to translate what he saw into new ideas and creations.

Acting as an anthropologist and observing people in their "natural habitat" can help you generate new ideas. Watch the way people do certain things and how they react in different situations. For example, whenever I am at the airport I watch people

wheeling around their suitcases—it is amazing how poorly designed some public spaces are, and I always think there are ample opportunities for innovation and improvement.

Be curious and ask lots of questions.

Many inventions come out of insights into the way other things work. For example, the Dyson vacuum that uses a cyclone to clean carpets was inspired by a visit to a sawmill that uses a cyclone to whisk away sawdust. When you expose yourself to a wide variety of things, what you learn can translate into other areas. Ideas are sparked from other ideas.

If you are looking for a place to start, think about everyday objects, for example, indoor plumbing, ballpoint pens, radio, Wi-Fi, and so on. Do you know how they really work?

When it comes to your work, don't just accept things at face value. Take the time to really understand what your compiler is doing, how the operating system works, and even how the network puts it all together. Spend time learning and going deep.

This is really about cultivating a sense of wonder and being curious about the world around you. When you meet people in different occupations, ask them lots of questions: "What are your biggest challenges?" or "What are the most interesting things you have learned in your work?" You can use every day and every conversation as a chance to expand your knowledge.

Embrace more of your ideas. I would be a terrible angel investor. I tend to have a conservative nature, and lots of the ideas I would consider dumb end up being successful (case in point: the Snuggie). As a result, I tend to dismiss many of my ideas. However, great ideas are often grown and inspired from other ideas.

If you have an idea, try discussing it with your peers (particularly the creative ones who get excited and can help you transform it, rather than the closed-minded friends who tend to see only the risks or problems). Brainstorm around it. How would you market the idea? What is its value? How could you transform it into a truly great idea? Even if you don't end up pursuing it, the exercise can help you think about things differently and build on your creative energy.

If you want to have great ideas, the first step is to have lots of ideas.

Putting It Into Practice

No matter what your profession, learning to think more innovatively and spark new ideas can help you. I have included some points and inspiration that have helped me, but the real key is changing your behavior and taking action. Here are some practical suggestions for fostering more great ideas.

► *Keep a list of your ideas.* I like to use Evernote because it syncs with my phone and computer, but any media will do. Just make sure you write things down so you don't forget.

► *Share your ideas with your friends.* This can be a great way to brainstorm, as well as to see different perspectives.

► *Get better at small talk by asking more questions.* Learn about other occupations and professions. What insights do they have to offer, and what challenges do they face?

► *Each week learn something new.* Carve out time each week to explore a topic you don't know much about. Investigate how things work and expand your knowledge of the world around you.

► *Read more.* It can be fiction or non-fiction, but just spend time learning.

► *Ask more questions.* When you don't understand something, ask. Or keep a list of the things you don't know, and when you have a spare 15 minutes spend time looking them up and investigating further.

Following this advice could inspire you to create more and expand your knowledge. Remember that all great ideas typically start from other ideas, so the first step you need to take is to start thinking.

I can't wait to see what you create. ■

Related articles on queue.acm.org

Fresh Starts

Kate Matsudaira

<https://queue.acm.org/detail.cfm?id=2996549>

Arrogance in Business Planning

Paul Vixie

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

Sink or Swim, Know When It's Time to Bail

Gordon Bell

<https://queue.acm.org/detail.cfm?id=966806>

Kate Matsudaira (katemats.com) is an experienced technology leader. She has worked at Microsoft and Amazon and successful startups before starting her own company, Popforms, which was acquired by Safari Books.

Copyright © 2018 held by owner/author.
Publication rights licensed to ACM. \$15.00

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

**Expert-curated guides to
the best of CS research.**

BY DEEPAK VASISHT

Research for Practice: Toward a Network of Connected Things

THIS INSTALLMENT OF Research for Practice features a curated selection from Deepak Vasisht, who takes us on a tour of systems and networking for the Internet of Things. Vasisht's selection spans energy harvesting to agriculture, providing a look into the future of IoT deployments and their usability.

—Peter Bailis

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



Over the past few years, we have started realizing the Internet of Things (IoT) dream. Amazon Echo, Dash buttons, Nest cameras, Google Home, and other devices have permeated our lives at home, and enterprises in various sectors such as retail, airlines, transportation, and logistics have started benefiting from industrial IoT solutions. Inspired by this impetus, General Electric recently estimated that investments in industrial IoT alone would top \$60 trillion over the next 15 years.

All this growth has been fueled by years of research tackling several challenges, ranging from low-power networking to new sensor designs to security and privacy. This installment of Research for Practice presents research papers that aim to make IoT deployments more pervasive, and to enable users to gain more utility from existing deployments.

Easing the Cost of Deployment

Zhang, P., Bharadia, D., Joshi, K., and Katti, S. HitchHike: Practical backscatter using commodity Wi-Fi. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, 2016, 259–271; <https://dl.acm.org/citation.cfm?id=2994565>.

One of the natural challenges of large-scale networked sensor deployments is the cost of powering them up. The high power cost of communication modules leads to frequent battery replacements, which, in turn, incur large labor costs. A recent sequence of backscatter solutions aims to change that by leveraging existing radio frequency transmissions to communicate. Specifically, backscatter communication systems allow devices to modulate and reflect existing Wi-Fi transmissions, thus enabling low-power communication modules that could be powered either by harvesting ambient power or by batteries that last several years.

HitchHike is unique for two reasons. First, not only can it reflect transmissions from commodity Wi-Fi devices, its reflections can also be received and

decoded by commodity Wi-Fi devices. This allows the widely prevalent Wi-Fi devices, such as your access point, to interact with the sensors at a very low cost for power. Second, HitchHike can achieve a data rate of 200Kbps at a distance of 54 meters. This data rate is high enough for most sensors and covers a very large area—larger than most homes and small enterprises. Going forward, HitchHike and others in this space promise to allow applications with severe power constraints, such as implantable sensors, wearables, sensors embedded in walls and bridges, among others.

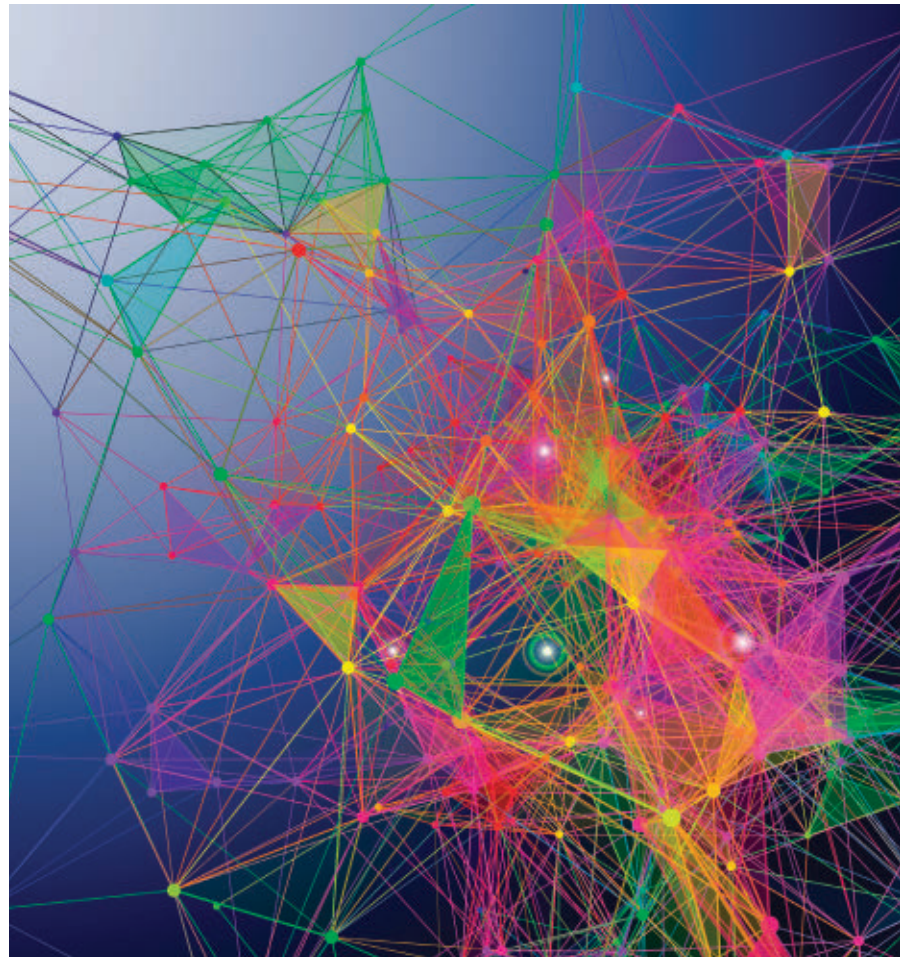
Designing Novel Sensing Mechanisms

Adib, F., Mao, H., Kabelac, Z., Katabi, D. and Miller, R.C.

Smart homes that monitor breathing and heart rate. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015, 837–846; <https://dl.acm.org/citation.cfm?id=2702200>.

A continuous thread of innovation in the IoT space has been the design of novel sensing mechanisms. A recent trend in this space has been monitoring health metrics such as breathing, heart rate, walking patterns, sleep stages, gait, and even emotional health in a completely contactless way. Amazon Echo or Google Home, for example, could be equipped with these capabilities, allowing users to know more about their physical and mental health.

This paper describes VitalRadio, a device that can monitor the breathing and heart rate of a user without any contact with the user at distances up to eight meters, even when the user is in a different room. VitalRadio presents the basic techniques that form the foundation of a lot of the later work on monitoring various other health metrics. On a high level, VitalRadio works by analyzing the reflections of radio signals from human bodies. As humans breathe (or their hearts beat), the reflections are affected by any minute change. VitalRadio extracts these small changes in the reflections to estimate the heart rate and breathing of indi-



viduals. While there are still some limitations on the operation of the system, like requirements for a one- to two-meter minimum separation between multiple users and quasi-static user behavior (watching TV, typing, and so on), none of these limitations is big enough to hinder mainstream health monitoring applications.

Exploiting Already-Deployed Sensors for New Services

Abari, O., Vasisht, D., Katabi, D. and Chandrakasan, A.

Caraoke: An e-toll transponder network for smart cities. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 297–310; <https://dl.acm.org/citation.cfm?id=2787504>.

The push for IoT over the past decade has ensured we have already deployed several billion sensors, such as toll tran-

sponders in cars, RFIDs in warehouses and restaurants, among others. These devices enable very specific functionality—automatic toll collection for cars or inventory tracking in warehouses. A key question, then, is what can be done to leverage these large deployments for more general-purpose applications?

Caraoke achieves this for e-toll transponders by using them to monitor traffic, locate and identify cars, detect speeding, and enable automated detection of empty parking spots—all without any changes to the e-toll transponders deployed on cars. Since such transponders are being used by 70%–89% of drivers in the U.S. (depending on the state) and are seeing increased adoption worldwide, Caraoke can play an important role in the push for smart cities where the traffic lights react to real-time traffic information, and driv-



The exciting research being done in IoT systems has put us ever closer not only to developing new services, but also to gathering new datasets for consumer and enterprise applications.



ers are automatically guided to empty parking spots.

The fundamental contribution of Caraoke is its ability to separate simultaneous transmissions from multiple transponders using novel signal processing techniques that exploit the frequency-domain structure of the signal. Caraoke incorporates these techniques into a new reader for transponders that can be deployed on streetlight poles and harvest solar energy for their operation. Going forward, such innovations in different domains can expand the utility of deployed IoT systems manifold.

Expanding IOT to “Untouched” Environments

Vasisht, D., Kapetanovic, Z., Won, J., Jin, X., Chandra, R., Kapoor, A., Sinha, S.N., Sudarshan, M. and Stratman, S.

FarmBeats: An IoT platform for data-driven agriculture. In *Proceedings of the 14th Usenix Symposium on Networked Systems Design and Implementation*, 2017; <https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/vasisht>.

While IoT has prospered in well-connected, well-powered environments such as urban homes and large enterprises, its adoption in harsher environments, without good sources of power and Internet, has been relatively low. Such environments include farming, construction, and mining—sectors that employ large sections of both the developing and the developed world. For example, even in the U.S., the process of data collection in farming remains primarily manual, which limits the adoption of advanced agricultural techniques to fewer than 20% of farmers.

FarmBeats attempts to tackle this challenge by focusing on the challenge of data-driven agriculture. Data-driven agricultural techniques such as precision irrigation can allow farmers to improve yields, reduce input cost, and enhance labor productivity. FarmBeats lets farmers employ these techniques by developing an end-to-end IoT platform for agriculture that enables seamless data collection from sensors, cameras, and drones.

FarmBeats uses three ideas to enable this platform. First, to enable connectivity on the farm, it uses a mix of TV white spaces (to allow long-range

connectivity over several miles) and Wi-Fi (to allow interfacing with commercial sensors). Second, to deal with weather-related outages and low bandwidths, it designs an IoT gateway that sits on the farm and provides services to the farmer while creating summaries for the cloud. Finally, it leverages machine-learning techniques to combine inputs from a drone and ground sensors to provide more accurate information and reduce the requirement for sensor deployments. The paper presents results from a multiseason deployment of FarmBeats on two different farms on two U.S. coasts.

Final Thoughts

The exciting research being done in IoT systems has put us ever closer not only to developing new services, but also to gathering new datasets for consumer and enterprise applications. Combined with recent significant advances in artificial intelligence and machine learning, these datasets can drive new applications. For example, VitalRadio has been extended to leverage novel deep-learning techniques to monitor sleep stages of a user completely passively. Researchers can also leverage new machine learning techniques as tools to design better systems. FarmBeats already shows how one can leverage AI to reduce the requirement for sensor placement and to guide the placement of sensors to maximize information.

While this scale of data presents new avenues for improvement, the key challenges for the everyday adoption of IoT systems revolve around managing this data. First, we need to consider where the data is being processed and stored—on the local edge computer or in the cloud—and what the privacy and systems implications of these policies are. Second, we need to develop systems that generate actionable insights from this diverse, difficult-to-interpret data for non-tech users. Solving these challenges will allow IoT systems to deliver maximum value to end users.

Deepak Vasisht is a Ph.D. candidate in electrical engineering and computer science at MIT. He has designed, built, and deployed systems that deliver ubiquitous sensing, accurate indoor positioning, enhanced communication capabilities, and new human computer interfaces.

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



@ACM_SUI



SpatialUserInteraction



<https://sui.acm.org>



SUI 2018

6th ACM Symposium on Spatial User Interaction

Berlin Oct 13-14

Park Inn, Alexanderpl. 7, 10178

WELCOME TO SUI 2018

SUI is an upcoming venue for leading-edge research
in the design of future 3D interactive interfaces

KEYNOTES



Alex Olwal Google

Gregory F. Welch University of Central Florida

IMPORTANT DATES

Full & short paper submissions **June 30 2018**

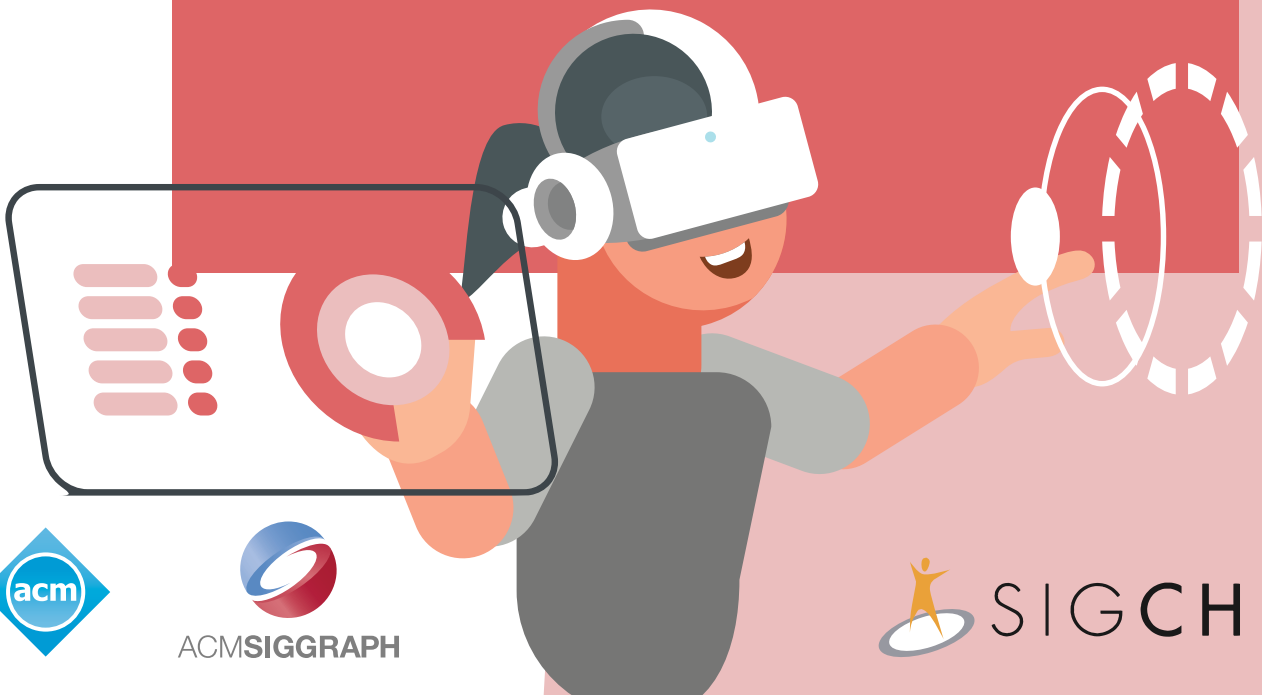
Poster / Demo submissions **August 11 2018**

*Selected contributions will be presented
in a special poster/demo track at UIST'18*

JOINT EVENT

UIST'18 and SUI'18 reception at **HCI Lab**

Hasso Plattner Institut <https://hpi.de/baudisch>



SIGCHI



DOI:10.1145/3134599

Such inputs distort how machine-learning-based systems are able to function in the world as it is.

**BY IAN GOODFELLOW, PATRICK MCDANIEL,
AND NICOLAS PAPERNOT**

Making Machine Learning Robust Against Adversarial Inputs

MACHINE LEARNING HAS advanced radically over the past 10 years, and machine learning algorithms now achieve human-level performance or better on a number of tasks, including face recognition,³¹ optical character recognition,⁸ object recognition,²⁹ and playing the game Go.²⁶ Yet machine learning algorithms that exceed human performance in naturally occurring scenarios are often seen as failing dramatically when an adversary is able to modify their input data even subtly. Machine learning is already used for many highly important applications and will be used in even more of even greater importance in the near future. Search algorithms, automated financial trading algorithms, data analytics, autonomous vehicles, and malware detection are all critically dependent on the underlying machine learning algorithms that interpret their respective domain inputs to provide intelligent outputs that facilitate the decision-making process of users or automated

systems. As machine learning is used in more contexts where malicious adversaries have an incentive to interfere with the operation of a given machine learning system, it is increasingly important to provide protections, or “robustness guarantees,” against adversarial manipulation.

The modern generation of machine learning services is a result of nearly 50 years of research and development in artificial intelligence—the study of computational algorithms and systems that reason about their environment to make predictions.²⁵ A subfield of artificial intelligence, most modern machine learning, as used in production, can essentially be understood as applied function approximation; when there is some mapping from an input x to an output y that is difficult for a programmer to describe through explicit code, a machine learning algorithm can learn an approximation of the mapping by analyzing a dataset containing several examples of inputs and their corresponding outputs. The learning proceeds by defining a “model,” a parametric function describing the mapping from inputs to outputs. Google’s image-classification system, Inception, has been trained with millions of labeled images.²⁸ It can classify images as cats, dogs, airplanes, boats, or more complex concepts on par or improving on human accuracy.

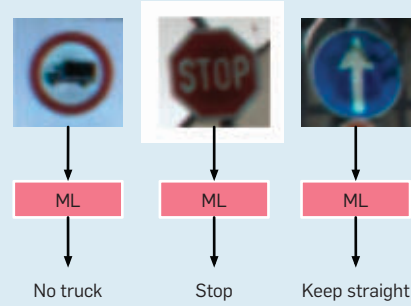
» key insights

- Machine learning has traditionally been developed following the assumption that the environment is benign during both training and evaluation of the model; while useful for designing effective algorithms, this implicitly rules out the possibility that an adversary could alter the distribution at either training time or test time.
- In the context of adversarial inputs at test time, few strong countermeasures exist for the many attacks that have been demonstrated.
- To end the arms race between attackers and defenders, we suggest building more tools for verifying machine learning models; unlike current testing practices, this could help defenders eventually gain a fundamental advantage.

SHEEP AND WOLF ILLUSTRATIONS BY ARNO



Figure 1. Example machine learning task: traffic sign classification.²⁷



Increases in the size of machine learning models and their accuracy is the result of recent advancements in machine learning algorithms,¹⁷ particularly to advance deep learning.⁷

One focus of the machine learning research community has been on developing models that make accurate predictions, as progress was in part measured by results on benchmark datasets. In this context, accuracy denotes the fraction of test inputs that a model processes correctly—the proportion of images that an object-recognition

algorithm recognizes as belonging to the correct class, and the proportion of executables that a malware detector correctly designates as benign or malicious. The estimate of a model's accuracy varies greatly with the choice of the dataset used to compute the estimate. The model's accuracy is generally evaluated on test inputs that were not used during the training process. The accuracy is usually higher if the test inputs resemble the training images more closely. For example, an object-recognition system trained on carefully curated photos may obtain high accuracy when tested on other carefully curated photos but low accuracy on photos captured more informally by mobile phone users.

Machine learning has traditionally been developed following the assumption that the environment is benign during both training and evaluation of the model. Specifically, the inputs x are usually assumed to all be drawn independently from the same probability distribution at both training and test time. This means that while test inputs x are new and previously unseen during the training process, they at least have the same statistical properties as the inputs used for training. Such assumptions have been useful for designing effective machine learning algorithms but implicitly rule out the possibility that an adversary could alter the distribution at either training time or test time. In this article, we focus on a scenario where an adversary chooses a distribution at test time that is designed to be exceptionally difficult for the model to process accurately. For example, an adversary might modify an image (slightly) to cause it to be recognized incorrectly or alter the code of an executable file to enable it to bypass a malware detector. Such inputs are called “adversarial examples”³⁰ because they are generated by an adversary.

The study of adversarial examples is in its infancy. Several algorithms have been developed for their generation and several countermeasures proposed. We follow with an overview of the foundations and advancements of this thriving research field and some predictions of where it might lead.

Machine Learning Systems in Adversarial Settings

To simplify our presentation in this article, we focus on machine learning

Figure 2. Example problem with two classes: the ideal decision boundary between the two models and the approximate boundary learned by the model.

Note that in higher dimensions, all examples are “close” to decision boundaries, as illustrated in this low-dimensional problem by the “pocket” of red class points included in the blue class.

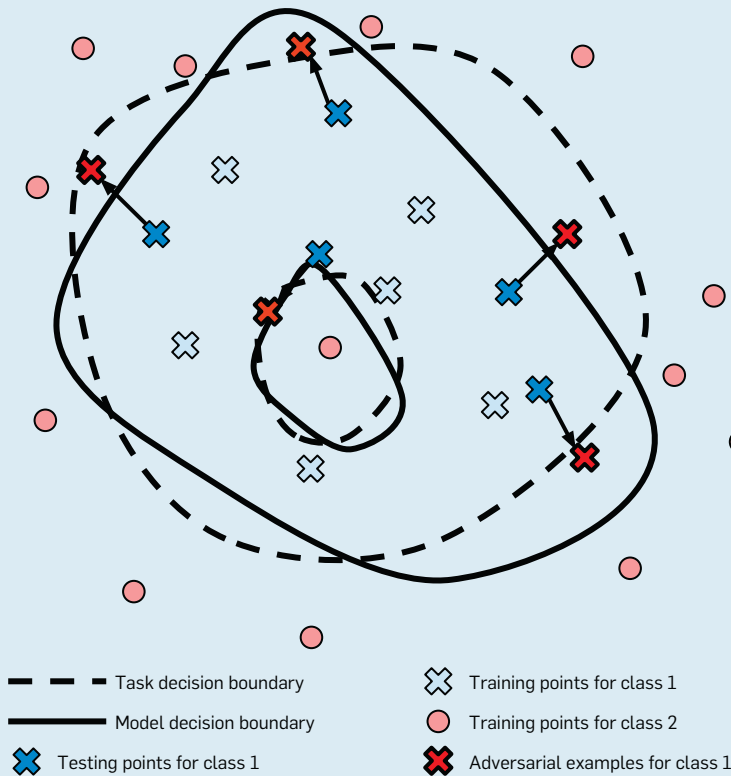
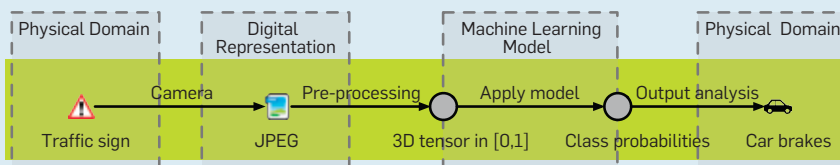


Figure 3. Example machine learning pipeline in the context of autonomous driving.²³



algorithms that perform “classification,” learning a mapping from an input x to a discrete variable y where y represents the identity of a class. As a unifying example, we discuss road-sign image recognition; the different values of y correspond to different types of road signs (such as stop signs, yield signs, and speed limit signs). Examples of input images and expected outputs are shown in Figure 1. Though we focus on image classification, the principles of adversarial machine learning apply to much more general artificial intelligence paradigms (such as reinforcement learning).¹²

Anatomy of a machine learning task. A machine learning algorithm is expected to produce a model capable of predicting the correct class of a given input. For instance, when presented with an image of a STOP sign, the model should output the class designating “STOP.” The generic strategy adopted to produce such a model is twofold: a family of parameterized representations, the model’s architecture, is selected, and the parameter values are fixed.

The architecture is typically chosen from among well-studied candidates (such as support vector machines and neural networks). The choice is made through either an exhaustive search or expert knowledge of the input domain. The chosen architecture’s parameter values are fixed so as to minimize the model’s prediction error over large collections of example pairs of inputs and expected outputs, the classes.

When this training phase is complete, the model can be used to predict the class of test inputs unseen during training. We can think of the classifier as defining a map of the input space, indicating the most likely class within each input region. The classifier will generally learn only an approximation of the true boundaries between regions for a variety of reasons (such as learning from few samples, using a limited parametric model family, and imperfect optimization of the model parameters), as shown schematically in Figure 2. The model error between the approximate and expected decision boundaries is exploited by adversaries, as explained in the following paragraphs.

The machine learning pipeline. Machine learning models are frequently deployed as part of a data pipeline; in-

puts to the model are derived from a set of preprocessing stages, and outputs of the model are used to determine the next states of the overall system.²³ For example, our running example of a traffic-sign classifier could be deployed in an autonomous vehicle, as illustrated in Figure 3. The model would be given as inputs images captured by a camera monitoring the side of the road and coupled with a detection mechanism for traffic signs. The class predicted by the machine learning model could then be used to decide what action should be taken by the vehicle (such as come to a stop if the traffic sign is classified as a “STOP” sign).

Attacking the system. As outlined earlier in this article, most machine learning models are designed, at least partially, based on the assumption that the data at test time is drawn from the same distribution as the training data, an assumption that is often violated. It is common for the accuracy of the model to degrade due to some relatively benign change in the distribution of test data. For example, if a different camera is used at test time from the one used to collect training images, the trained model might not work well on the test images. More important, this phenomenon can be exploited by adversaries capable of manipulating inputs before they are presented to the machine learning model. The adversary’s motivation for “controlling” the

model’s behavior this way stems from the implications of machine learning predictions on consequent steps of the data pipeline.²³ In our running example of an autonomous vehicle, an adversary capable of crafting STOP signs classified as “yield” signs may cause the autonomous vehicle to disobey traffic laws and potentially cause an accident. Machine learning is also applied to other sensitive domains (such as financial fraud³ and malware detection¹) where adversarial incentives to have the model mis-predict are evident.

As is common in modeling the security of any domain, the domain of adversaries against machine learning systems can be structured around a taxonomy of capabilities and goals.^{2,22,23} As reflected in Figure 4, the adversary’s strength is characterized by its ability to access the model’s architecture, parameter values, and training data. Indeed, an adversary with access to the model architecture and parameter values is capable of reproducing the targeted system on its own machine and thus operates in a “white-box” scenario. In addition to different degrees of knowledge about the model internals, adversaries may also be distinguished by their ability to submit inputs directly to the machine learning model or only indirectly through the data pipeline in Figure 3. For instance, Kurakin et al.¹⁶ demonstrated that adversaries can manipulate a machine learning

Figure 4. A taxonomy of adversaries against machine learning models at test time.²²

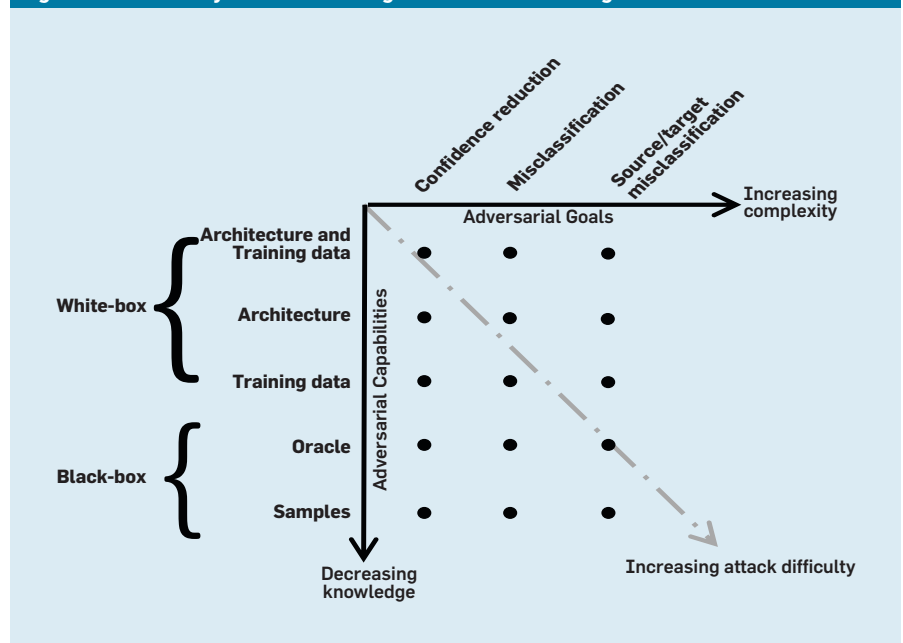


Figure 5. Original image (left) and adversarial image (right).

image classification model's predictions through physical perturbations of images before they are recorded by a camera and pre-processed. It should also be noted that this characterization of an adversary's strength significantly differs from bounds on available computational resources as is commonly employed in various areas of security and cryptography. As the field of adversarial machine learning matures, we expect the research community to reflect and expand on this preliminary measure of strength limited to the adversary's knowledge of a system's potential weakness(es).

Moreover, adversarial incentives may define a range of attack goals. When the adversary is interested in having a given input classified in any class that is different from its correct class, we refer to this as an "untargeted misclassification attack." Instead, when the adversary is interested in defining the target class in which it intends to have the model misclassify an input from any (correct) source class, this is a "source-target misclassification attack" (also called a "targeted attack" in the literature).

Poisoning the model. Note that an adversary may also attack the training process itself. If the adversary can insert its own samples or otherwise corrupt the data used for training, the adversary can thus influence the model to induce incorrect connections between input features and classes (called "false learning") or reduce confidence in the labeling, decreasing model accuracy. In either case, the corruption of the training process compromises the integrity of the model and decreases its

robustness against adversarial inputs. Although such attacks are being studied and countermeasures considered by the research community, they are beyond the scope of this article.

Adversarial Example Generation

The challenge for the adversary is figuring out how to generate an input with the desired output, as in the source-target-misclassification attack. In such an attack, the adversary starts with a sample that is legitimate (such as a STOP sign) and modifies it through a perturbation process to attempt to cause the model to classify it in a chosen target class (such as the one corresponding to a YIELD sign).

For an attack to be worth studying from a machine learning point of view, it is necessary to impose constraints that ensure the adversary is not able to truly change the class of the input. For example, if the adversary could physically replace a stop sign with a yield sign or physically paint a yield symbol onto a stop sign, a machine learning algorithm must be able to still recognize it as a yield sign. In the context of computer vision, we generally consider only modifications of an object's appearance that do not interfere with a human observer's ability to recognize the object. The search for misclassified inputs is thus done with the constraint that these inputs should be visually very similar to a legitimate input. Consider the two images in Figure 5, potentially consumed by an autonomous vehicle. To the human eye, they appear to be the same, and our biological classifiers (vision) identify each one as a stop sign. The image on

the left is indeed an ordinary image of a stop sign. We produced the image on the right by adding a small, precise perturbation that forces a particular image classification deep neural network to classify it as a yield sign. Here, the adversary could potentially use the altered image to cause the car to behave dangerously, if the car lacks fail-safes (such as maps of known stop-sign locations). In other application domains, the constraint differs. When targeting machine learning models used for malware detection, the constraint becomes that the input—or malware software—misclassified by the model must still be in a legitimate executable format and execute its malicious logic when executed.¹⁰

White-box vs. black-box. One way to characterize an adversary's strength is the amount of access the adversary has to the model. In a white-box scenario, the adversary has full access to the model whereby the adversary knows what machine learning algorithm is being used and the values of the model's parameters. In this case, we show in the following paragraphs that constructing an adversarial example can be formulated as a straightforward optimization problem. In a black-box scenario, the attacker must rely on guesswork, because the machine learning algorithm used by the defender and the parameters of the defender's model are not known. Even in this scenario, where the attacker's strength is limited by incomplete knowledge, the attacker might still succeed. We now describe the white-box techniques first because they form the basis for the more difficult black-box attacks.

White-box attacks. An adversarial example x^* is found by perturbing an originally correctly classified input x . To find x^* , one solves a constrained optimization problem. One very generic approach, applicable to essentially all machine learning paradigms, is to solve for the x^* that causes the most expected loss (a metric reflecting the model's error), subject to a constraint on the maximum allowable deviation from the original input x ; in the case of machine learning models solving classification tasks, the loss of a model can be understood as its prediction error. Another approach, specialized to classifiers, is to impose a constraint that

the perturbation must cause a misclassification and solve for the smallest possible perturbation

$$x^* = x + \arg \min \{ \|z\| : f(x+z) = t \} \quad (1)$$

where x is an input originally correctly classified, $\|\cdot\|$ a norm that appropriately quantifies the similarity constraints discussed earlier, and t is the target class chosen by the adversary. In the case of “untargeted attacks,” t can be any class different from the correct class $f(x)$. For example, for an image the adversary might use the ℓ_0 “norm” to force the attack to modify very few pixels, or the ℓ_∞ norm to force the attack to make only very small changes to each pixel. All of these different ways of formulating the optimization problem search for an x^* that should be classified the same as x (because it is very similar to x) yet is classified differently by the model. These optimization problems are typically intractable, so most adversarial example-generation algorithms are based on tractable approximations.

Gradient-based search algorithms. The optimization algorithms described earlier are, in principle, intractable for most interesting models, because most interesting models use nonlinear, non-convex functions. In practice, gradient-based optimization algorithms reliably find solutions that cause misclassification, presumably because a point x^* can cause misclassification without being an optimal solution to Equation (1).

Several approaches using gradient-based optimization have been introduced to date. We present here three canonical examples of gradient-based attacks: the L-BFGS approach, the Fast Gradient Sign Method (FGSM), and the Jacobian Saliency Map Approach (JSMA).

Szegedy et al.³⁰ adapted the L-BFGS method to the constrained optimization problem outlined earlier. They were the first researchers to demonstrate perturbations indistinguishable from human observers that were sufficient to force a computer-vision model to misclassify an image encoded by x^* . Recently, the L-BFGS method was revisited by Carlini et al.⁴ who found that using the Adam optimizer along with customized objectives reduces the size of the perturbation required to ensure



The adversary's strength is characterized by its ability to access the model's architecture, parameter values, and training data.



an input will be misclassified. However, both approaches come at a high computational cost. It is possible to reduce that cost, usually at the price of also reducing the effectiveness of the attack. In general, attacks exist along a continuum. For example, it is possible to simply run the L-BFGS algorithm for fewer iterations to obtain a less-expensive attack with a lower success rate.

One attack with especially low computational cost is the FGSM,⁹ an approach that maximizes the model's prediction error while keeping the ℓ_0 norm of the perturbation added to the input constant. This attack is based on the observation that many machine learning models, even neural networks, are very linear as a function of the input x . One way to formulate an adversarial attack is

$$x^* = \max_x J_f(x) \text{ subject to } \|x^* - x\|_\infty \leq \epsilon, \quad (2)$$

where J_f is the expected loss incurred by the machine learning model, and is a way to measure the model's prediction error. This optimization algorithm is typically intractable, but if the true J_f is replaced with a first-order Taylor series approximation of J_f formed by taking the gradient at x , the optimization problem can be solved in closed form

$$x^* = x + \epsilon \cdot \text{sign}(\nabla_x J_f(x, y)) \quad (3)$$

Because the linear approximation used by the Taylor series expansion approximately holds, it often finds adversarial examples despite its low runtime requirements.

In the JSMA, the adversary chooses a target class in which the sample should be misclassified by the model.²² Given model f , the adversary crafts an adversarial sample $x^* = x + \delta_x$ by adding a perturbation δ_x to a subset of the input components x_i . To choose the perturbation, the adversary would sort the components by decreasing adversarial saliency value. Intuitively, a saliency value is a measure of how important a particular feature is to determining a given output class (such as the importance of a particular pixel or group of pixels in determining what kind of sign is in an image). The adversarial saliency value $S(f, x, t)[i]$ of component i for an adversarial target class t is de-

defined as

$$S(f, x, t)[i] = \begin{cases} 0 & \text{if } \frac{\partial f_t}{\partial x_i}(x) < 0 \text{ or } \sum_{j \neq t} \frac{\partial f_j}{\partial x_i}(x) > 0 \\ \frac{\partial f_t}{\partial x_i}(x) \left(\sum_{j \neq t} \frac{\partial f_j}{\partial x_i}(x) \right) & \text{otherwise} \end{cases} \quad (4)$$

where matrix

$$J_f = \left[\frac{\partial f_j}{\partial x_i} \right]_{ij}$$

is the model's Jacobian matrix. Input components i are added to perturbation δ_x in order of decreasing adversarial saliency value $S(x, t)[i]$ until the resulting adversarial sample $x^* = x + \delta_x$ is misclassified by f . The perturbation introduced for each selected input component can vary, and larger perturbations usually mean that fewer components need to be perturbed.

Each algorithm has its own benefits and drawbacks. The FGSM is well suited for fast crafting of many adversarial

examples with relatively large perturbations and is potentially easier to detect. JSMA and L-BFGS (or other iterative optimization algorithms) both produce stealthier perturbations at greater computational cost. FGSM sometimes works better than L-BFGS if the gradient is very small, because the sign operation removes the dependence on the gradient magnitude. All of these algorithms can fail to fool the classifier. On typical machine learning benchmark problems, all three algorithms have a near-100% success rate against normal machine learning algorithms. Defense techniques can thwart a high percentage of FGSM and JSMA attacks, but L-BFGS is essentially a brute-force white-box approach that almost always succeeds regardless of defense techniques, given enough runtime.

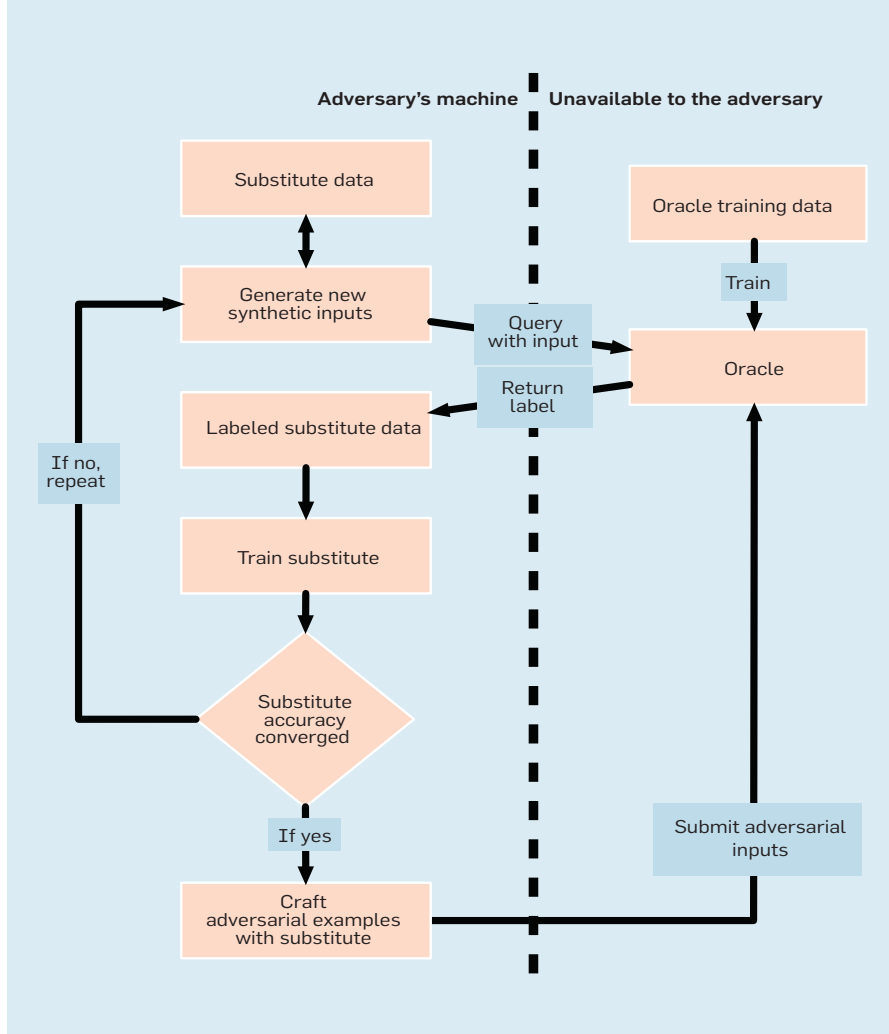
Black-box attacks. Although in some cases it is possible for the adversary to have access to a model's parameters,

in most realistic threat models, the adversary has access to the model only through a limited interface. Additional strategies are thus required to conduct attacks without access to the model's gradients, which are unavailable in a limited-interface black-box scenario.

In such threat models, a possible strategy for the adversary is to train another model, different from the one being targeted. This model, called the "substitute," is used by an adversary to compute the gradients required for the attack. Assuming that both the substitute and the targeted models operate in similar ways (such as they have a similar decision boundary), an adversarial example computed using the substitute model is highly likely to be misclassified by the targeted one. This transfer of adversarial examples from a substitute model to a target model was first demonstrated by Szegedy et al.³⁰

There are two ways to pursue such a strategy. One is for the attacker to collect and label the attacker's own training set.³⁰ This approach works with absolutely no access to the model but can be expensive because it might require gathering a large number of real input examples and human effort to label each example. When the attacker is able to query the target model by sending it inputs and observing the returned outputs, a much less-expensive approach is to strategically send algorithmically generated inputs in order to reverse engineer the target model, without any (or very little) training data. Such a strategy was introduced by Papernot et al.²¹ and is illustrated in Figure 6. In this work, the adversary has access only to the model through an API that returns the class predicted by the model for any input chosen by the adversary. That is, the API acts as an "oracle" for the model. In addition to the model architecture and training data being unavailable to the adversary, the adversary does not even need to know what type of architecture is used to create the machine learning model, as if the model is, say, a support vector machine, a neural network, or something else. Through interactions with the API, the adversary is going to create a training set for its substitute model, in a way that ensures the substitute makes predictions similar to the one

Figure 6. Black-box attack strategy introduced by Papernot et al.²¹



made by the targeted model and is, in turn, a good indicator of where the targeted model will make mistakes. The idea is that the adversary uses the oracle to explore and reconstruct the decision boundary of the victim model. The key is being intelligent about that exploration.

The main challenge resides in selecting the inputs the adversary sends to the API to reduce the total number of queries made—to reduce the detectability of the attack. The approach taken by Papernot et al.²¹ is to generate synthetic inputs using a few real inputs collected by the adversary. For instance, when targeting a traffic-sign classifier, the adversary would collect a small set of images of each traffic sign and then run the following augmentation technique for each of these images to find new inputs that should be labeled with the API. The augmentation technique takes a set of training inputs S_p , the labels they were given $\forall x \in S_p, f(x)$ by the targeted model f , and the current substitute model g

$$S_{p+1} \leftarrow \{x + \lambda \cdot \text{sign}(J_g[f(\bar{x})]) : x \in S_p\} \cup S_p \quad (5)$$

where J_g denotes here the Jacobian matrix of g . The substitute is trained by successively labeling and augmenting S_p .

When the substitute is sufficiently accurate, the adversary can use it to run one of the attacks described earlier, as in, for instance, the FGSM or the JSMA. The adversarial examples produced by these algorithms on the substitute g are likely to transfer to the targeted model f . This “transferability” property, first observed among deep neural networks and linear models by Szegedy et al.,³⁰ is known to hold across many types of machine learning models.²⁰ Indeed, Figure 7 reports transferability rates, the number of adversarial examples misclassified by a model B, despite being crafted with a different model A, for several pairs of machine learning models trained on a standard image-recognition benchmark as in the MNIST dataset of handwritten digits. For many such pairs, transferability is generally consistent between the two models. Adversarial examples even transfer to an ensemble of models that make predictions based on a majority vote among the class predicted by a collection of independent models.

Black-box attacks using these strategies have been demonstrated against proprietary models and accessible through a “machine learning as a service” query APIs (such as those from Google and Amazon^{21,33}).

Yet other strategies exist for black-box attacks, with more identified every day. Notably, Xu et al.³⁵ found that genetic algorithms produce adversarial examples. However, this requires that the adversary is able to access the output confidence values (such as output probabilities) returned by the model or at least to approximate these confidence values.⁵ These probabilities are used to compute a fitness function for genetic variants of the original input, and, in turn, retain the variants that are most “fit” to achieve the adversarial goal of misclassification. A drawback of this strategy is that, compared to other black-box strategies,^{21,30} the algorithm must be able to make a large number of model-prediction queries before it finds evasive variants and is thus more likely to be detected at runtime.

Defenses and Their Limitations

Given the existence of adversarial samples, an important question is: What can be done to defend models against them? The defensive property of a machine learning system in this context is called “robustness against adversarial samples.” We now explore several known defenses categorized into three classes: model training, input validation, and architectural changes.

Model training. Adversarial training and defensive distillation are two techniques proposed to train models that explicitly attempt to be robust to adversarial examples. Adversarial training seeks to improve the generalization of a model when presented with adversarial examples at test time by proactively generating adversarial examples as part of the training procedure. This idea was first introduced by Szegedy et al.³⁰ but not yet practical due to the high computational cost of generating adversarial examples. Goodfellow et al.⁹ then showed how to generate adversarial examples inex-

Figure 7. Transferability matrix. The source model is used to craft adversarial examples and the target model to predict their class.

The transferability reported is the average rate of mistakes made by the target model on adversarial examples crafted using the source model. The models considered are a deep neural network, logistic regression, support vector machine, decision tree, and k -nearest neighbor. The ensemble target model refers to an ensemble making predictions based on a majority vote among the class predicted by each of the five previous models. All adversarial examples were produced with similar perturbation norms.²⁰

Source Machine Learning Technique	Target Machine Learning Technique						
	DNN	LR	SVM	DT	kNN	Ens.	
	DNN	38.27	23.02	64.32	79.31	8.36	20.72
	LR	6.31	91.64	91.43	87.42	11.29	44.14
	SVM	2.51	36.56	100.0	80.03	5.19	15.67
	DT	0.82	12.22	8.85	89.29	3.31	5.11
	kNN	11.75	42.89	82.16	82.95	41.65	31.92

pensively with the fast-gradient-sign method and made it computationally efficient to continuously generate new adversarial examples every time the model parameters change during the training process. The model is then trained to assign the same label to the adversarial example as to the original example.

Defensive distillation smooths the model's decision surface in adversarial directions that could be exploited by the adversary.²⁴ Distillation is a training procedure whereby one model is trained to predict the probabilities output by another model that was trained earlier. Distillation was introduced by Hinton et al.¹¹ aiming for a small model to mimic a large, computationally expensive model. Defensive distillation has a different goal—make the final model's responses more smooth—so it works even if both models are the same size. It may seem counterintuitive to train one model to predict the output of another model with the same architecture. The reason it works is that the first model is trained with “hard” labels (100% probability an image is a STOP sign rather than a YIELD sign) and then provides “soft” labels (95% probability an image is a STOP sign rather than a YIELD sign) used to train the second model. The second distilled model is more robust to attacks (such as the fast gradient sign method and the Jacobian-based saliency map approach).

Both adversarial training and defensive distillation suffer from limitations, however. They are generally effective against inexpensive white-box attacks but can be broken using black-box attacks²¹ or computationally expensive attacks based on iterative optimization.⁴ These two strategies are examples of defenses that perform gradient masking,^{21,23} removing the gradient of the model used by the adversary to find good-candidate directions to construct adversarial examples. However, the adversary can still evade the model practically if it can find other ways to identify candidate adversarial directions (such as through a black-box attack) or start the gradient-based search outside the input region impacted by the defense (such as by first taking a step in a random direction³²).

Input validation and preprocessing.



Future work may reveal architectures that resist adversarial examples yet are amenable to effective training.



Perhaps the most obvious defense is to validate the input before it is given to the model and possibly preprocess it to remove potentially adversarial perturbations. In many application domains there are verifiable properties of inputs that should never be violated in practice. For example, an input image from a camera sensor can be checked for realism; for example, certain properties of cameras and light ensure certain pixel neighborhoods (such as neighbor pixels with exceptionally high contrast) never occur. Such defenses are limited in that they are highly domain dependent and subject to environmental factors. Moreover, it is not clear that the constraints placed on the domain just increase the difficulty of adversarial sample generation or provide broad protections. It is highly likely that the effectiveness of input constraints as a countermeasure is also domain specific.

Architecture modifications. It is also possible to defend against adversarial samples by altering the structure of the machine learning system.

Highly nonlinear machine learning models are more robust to adversarial examples but also more difficult to train and generally do not perform well in a baseline non-adversarial setting.⁹

Future work may reveal architectures that resist adversarial examples yet are amenable to effective training.

From Testing to Verification

The limitations of existing defenses point to the lack of theory and practice of verification and testing of machine learning models. To design reliable systems, engineers engage in both testing and verification. By testing, we mean evaluating the system under various conditions and observing its behavior, watching for defects. By “verification,” we mean producing a compelling argument that the system will not misbehave under a broad range of circumstances.

Machine learning practitioners have traditionally relied primarily on testing. A classifier is usually evaluated by applying the classifier to several examples drawn from a test set and measuring its accuracy on these examples.

To provide security guarantees, it is necessary to ensure properties of the model besides its accuracy on naturally occurring test-set examples. One well-studied property is robustness to

adversarial examples. The natural way to test robustness to adversarial examples is simply to evaluate the accuracy of the model on a test set that has been adversarially perturbed to create adversarial examples.³⁰

Unfortunately, testing is insufficient for providing security guarantees, as an attacker can send inputs that differ from the inputs used for the testing process. For example, a model that is tested and found to be robust against the fast gradient sign method of adversarial example generation⁹ may be vulnerable to computationally expensive methods like attacks based on L-BFGS.³⁰

In general, testing is insufficient because it provides a “lower bound” on the failure rate of the system when an “upper bound” is necessary for providing security guarantees. Testing identifies n inputs that cause failure, so the engineer can conclude that at least n inputs cause failure; the engineer would prefer to have a means of becoming reasonably confident that at most n inputs cause failure.

Putting this in terms of security, a defense should provide a measurable guarantee that characterizes the space of inputs that cause failures. Conversely, the common practice of testing can only provide instances that cause error and is thus of limited value in understanding the robustness of a machine learning system. Development of an input-characterizing guarantee is central to the future of machine learning in adversarial settings and will almost certainly be grounded in formal verification.

Theoretical verification of machine learning. Verification of machine learning model robustness to adversarial examples is in its infancy. Current approaches verify that a classifier assigns the same class to all points within a specified neighborhood of a point x . Huang et al.¹³ developed the first verification system for demonstrating that the output class is constant across a desired neighborhood. This first system uses an SMT solver. Its scalability is limited, and scaling to large models requires making strong assumptions (such as that only a subset of the units of the network is relevant to the classification of a particular input point). Such assumptions mean the system can no longer provide an absolute guarantee of the absence of adversarial examples, as an adversarial

example that violates the assumptions could evade detection. Reluplex¹⁵ is another verification system that uses linear programming solvers to scale to much larger networks. Reluplex is able to become much more efficient by specializing on rectified linear networks^{6,14,18} and their piecewise linear structure.

These current verification systems are limited in scope because they verify only that the output class remains constant in some specified neighborhood of some specific point x . It is infeasible for a defender to fully anticipate all future attacks when specifying the neighborhood surrounding x to verify. For instance, a defender may use a verification system to prove there are no adversarial examples within a max-norm ball of radius ϵ , but then an attacker may devise a new way of modifying x that should leave the class unchanged yet has a high max-norm. An even greater challenge is verifying the behavior of the system near new test points x' .

In a traditional machine learning setting there are clear theoretical limits as to how well a machine learning system can be expected to perform on new test points. For example, the “no free lunch theorem”³⁴ states that all supervised classification algorithms have the same accuracy on new test points when averaged over all possible datasets.

One important open theoretical question is: Can the no-free-lunch theorem be extended to the adversarial setting? If we assume attackers operate by making small perturbations to the test set, then the premise of the no-free-lunch theorem, where the average is taken over all possible datasets, including those with small perturbations, should not be ignored by the classifier, no longer applies. Depending on the resolution of this question, the arms race between attackers and defenders could have two different outcomes. The attacker might fundamentally have the advantage due to inherent statistical difficulties associated with predicting the correct value for new test points. If we are fortunate, the defender might have a fundamental advantage for a broad set of problem classes, paving the way for the design and verification of algorithms with robustness guarantees.

Reproducible testing with CleverHans. While verification is a challenge

even from a theoretical point of view, straightforward testing can likewise be a challenge from a practical point of view. Suppose a researcher proposes a new defense procedure and evaluates that defense against a particular adversarial example attack procedure. If the resulting model obtains high accuracy, does it mean the defense was effective? Possibly, but it could also mean the researcher’s implementation of the attack was weak. A similar problem occurs when researchers test a proposed attack technique against their own implementation of a common defense procedure.

To resolve these difficulties, we created the CleverHans¹⁹ library with reference implementations of several attack and defense procedures. Researchers and product developers can use CleverHans to test their models against standardized, state-of-the-art attacks and defenses. This way, if a defense obtains high accuracy against a CleverHans attack, the test would show that the defense overcomes this standard implementation of the attack, and if an attack obtains a high failure rate against a CleverHans defense, the test would show that the attack is able to defeat a rigorous implementation of the defense. While such standardized testing of attacks and defenses does not substitute in any way to rigorous verification, it does provide a common benchmark. Moreover, results in published research are comparable to one another, so long as they are produced with the same version of CleverHans in similar computing environments.

Future of Adversarial Machine Learning

Adversarial machine learning is at a turning point. In the context of adversarial inputs at test time, we have several effective attack algorithms but few strong countermeasures. Can we expect this situation to continue indefinitely? Can we expect an arms race with attackers and defenders repeatedly seizing the upper hand in turn? Or can we expect the defender to eventually gain a fundamental advantage?

We can explain adversarial examples in current machine learning models as the result of unreasonably linear extrapolation⁹ but do not know what will happen when we fix this

particular problem; it may simply be replaced by another equally vexing category of vulnerabilities. The vastness of the set of all possible inputs to a machine learning model seems to be cause for pessimism. Even for a relatively small binary vector, there are far more possible input vectors than there are atoms in the universe, and it seems highly improbable that a machine learning algorithm would be able to process all of them acceptably. On the other hand, one may hope that as classifiers become more robust, it could become impractical for an attacker to find input points that are reliably misclassified by the target model, particularly in the black-box setting.

These questions may be addressed empirically, by actually playing out the arms race as new attacks and new countermeasures are developed. We may also be able to address these questions theoretically, by proving the arms race must converge to some asymptote. All these endeavors are difficult, and we hope many will be inspired to join the effort.

Acknowledgments

Author Nicolas Papernot is supported by a Google Ph.D. Fellowship in Security. Research was supported in part by the Army Research Laboratory under Cooperative Agreement Number W911NF-13-2-0045 (ARL Cyber Security CRA) and the Army Research Office under grant W911NF-13-1-0421. The views and conclusions contained in this article are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. government. The U.S. government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright notation hereon. 

References

1. Arp, D., Spreitzenbarth, M., Hubner, M., Gascon, H., Rieck, K., and Siemens, C.E.R.T. Drebin: Effective and explainable detection of Android malware in your pocket. In *Proceedings of the NDSS Symposium* (San Diego, CA, Feb.). Internet Society, Reston, VA, 2014, 23–26.
2. Barrero, M., Nelson, B., Sears, R., Joseph, A.D., and Tygar, J.D. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security* (Taipei, Taiwan, Mar. 21–24). ACM Press, New York, 2006, 16–25.

3. Bolton, R.J. and Hand, D.J. Statistical fraud detection: A review. *Statistical Science* 17, 3 (2002), 235–249.
4. Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. *arXiv preprint*, 2016; <https://arxiv.org/pdf/1608.04644.pdf>
5. Dang, H., Yue, H., and Chang, E.C. Evading classifier in the dark: Guiding unpredictable morphing using binary-output blackboxes. *arXiv preprint*, 2017; <https://arxiv.org/pdf/1705.07535.pdf>
6. Glorot, X., Bordes, A., and Bengio, Y. Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics* (Ft. Lauderdale, FL, Apr. 11–13, 2011), 315–323.
7. Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, Cambridge, MA, 2016; <http://www.deeplearningbook.org/>
8. Goodfellow, I.J., Bulatov, Y., Ibarz, J., Arnaud, S., and Shet, V. Multi-digit number recognition from Street View imagery using deep convolutional neural networks. In *Proceedings of the International Conference on Learning Representations* (Banff, Canada, Apr. 14–16, 2014).
9. Goodfellow, I.J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint*, 2014; <https://arxiv.org/pdf/1412.6572.pdf>
10. Grosse, K., Papernot, N., Manoharan, P., Backes, M., and McDaniel, P. Adversarial perturbations against deep neural networks for malware classification. In *Proceedings of the European Symposium on Research in Computer Security* (Oslo, Norway, 2017).
11. Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint*, 2015; <https://arxiv.org/abs/1503.02531>
12. Huang, S., Papernot, N., Goodfellow, I., Duan, Y., and Abbeel, P. Adversarial attacks on neural network policies. *arXiv preprint*, 2017; <https://arxiv.org/abs/1702.02284>
13. Huang, A., Kwiatkowska, M., Wang, S., and Wu, M. Safety verification of deep neural networks. In *Proceedings of the International Conference on Computer-Aided Verification* (2016); https://link.springer.com/chapter/10.1007/978-3-319-63387-9_1
14. Jarrett, K., Kavukcuoglu, K., Ranzato, M.A., and LeCun, Y. What is the best multi-stage architecture for object recognition? In *Proceedings of the 12th IEEE International Conference on Computer Vision* (Kyoto, Japan, Sept. 27–Oct. 4). IEEE Press, 2009.
15. Katz, G., Barrett, C., Dill, D., Julian, K., and Kochenderfer, M. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Proceedings of the International Conference on Computer-Aided Verification*. Springer, Cham, 2017, 97–117.
16. Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. In *Proceedings of the International Conference on Learning Representations* (2017); <https://arxiv.org/abs/1607.02533>
17. Murphy, K.P. *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, MA, 2012.
18. Nair, V. and Hinton, G.E. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the International Conference on Machine Learning* (Haifa, Israel, June 21–24, 2010).
19. Papernot, N., Goodfellow, I., Sheatsley, R., Feinman, R., and McDaniel, P. CleverHans v2.1.0: An adversarial machine learning library; <https://github.com/tensorflow/cleverhans>
20. Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: From phenomena to black-box attacks using adversarial samples. *arXiv preprint*, 2016; <https://arxiv.org/abs/1605.07277>
21. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., and Swami, A. Practical black-box attacks against deep learning systems using adversarial examples. In *Proceedings of the ACM Asia Conference on Computer and Communications Security* (Abu Dhabi, UAE). ACM Press, New York, 2017.
22. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., and Swami, A. The limitations of deep learning in adversarial settings. In *Proceedings of the 2016 IEEE European Symposium on Security and Privacy* (Saarbrücken, Germany, Mar. 21–24). IEEE Press, 2016, 372–387.
23. Papernot, N., McDaniel, P., Sinha, A., and Wellman, M. Towards the science of security and privacy in machine learning. In *Proceedings of the Third IEEE European Symposium on Security and Privacy* (London, U.K.); <https://arxiv.org/abs/1611.03814>
24. Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. In

- Proceedings of the 37th IEEE Symposium on Security and Privacy* (San Jose, CA, May 23–25). IEEE Press, 2016, 582–597.
25. Russell, S. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 1995, 25–27.
26. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. et al. Mastering the game of Go with deep neural networks and tree search. *Nature* 529, 7587 (2016), 484–489.
27. Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks* (2012); <https://doi.org/10.1016/j.neunet.2012.02.016>
28. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, C., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Press, 2015, 1–9.
29. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the Inception architecture for computer vision. *ArXiv e-prints*, Dec. 2015; <https://arxiv.org/abs/1512.00567>
30. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *Proceedings of the International Conference on Learning Representations*, 2014.
31. Taigman, Y., Yang, M., Ranzato, M.A., and Wolf, L. DeepFace: Closing the gap to human-level performance in face verification. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. IEEE Press, 2014.
32. Tramèr, F., Kurakin, A., Papernot, N., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv preprint*, 2017; <https://arxiv.org/abs/1705.07204>
33. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., and Ristenpart, T. Stealing machine learning models via prediction APIs. In *Proceedings of the USENIX Security Conference* (San Francisco, CA, Jan. 25–27). USENIX Association, Berkeley, CA, 2016.
34. Wolpert, D.H. The lack of a priori distinctions between learning algorithms. *Neural Computation* 8, 7 (1996), 1341–1390.
35. Xu, W., Qi, Y., and Evans, D. Automatically evading classifiers. In *Proceedings of the 2016 Network and Distributed Systems Symposium* (San Diego, CA, Feb. 21–24). Internet Society, Reston, VA, 2016.

Ian Goodfellow (goodfellow@google.com) is a staff research scientist at Google Brain, Mountain View, CA, USA, and inventor of Generative Adversarial Networks.

Patrick McDaniel (mcdaniel@cse.psu.edu) is the William L. Weiss Professor of Information and Communications Technology in the School of Electrical Engineering and Computer Science at Pennsylvania State University, University Park, PA, USA, and a fellow of both IEEE and ACM.

Nicolas Papernot (ngp5056@cse.psu.edu) is a Google Ph.D. Fellow in Security in the Department of Computer Science and Engineering at Penn State University, University Park, PA, USA.

Copyright held by the authors.



Watch the authors discuss their work in this exclusive *Communications* video. <https://cacm.acm.org/videos/making-machine-learning-robust-against-adversarial-inputs>

Designers can create designs that nudge users toward the most desirable option.

**BY CHRISTOPH SCHNEIDER, MARKUS WEINMANN,
AND JAN VOM BROCKE**

Digital Nudging: Guiding Online User Choices through Interface Design

LIFE IS FULL of choices, often in digital environments. People interact with e-government applications; trade financial products online; buy products in Web shops; book hotel rooms on mobile booking apps; and make decisions based on content presented in organizational information systems. All such choices are influenced by

the choice environment, as reflected in this comment: “What is chosen often depends upon how the choice is presented.”¹⁶ Why? People have cognitive limitations, so their rationality is bounded,²⁷ and heuristics and biases drive their decision making.³⁴ Designers of choice environments, or “choice architects,”³² can thus use these heuristics and biases to manipulate the choice environment to subtly guide users’ behavior by gently “nudging” them toward certain choices.

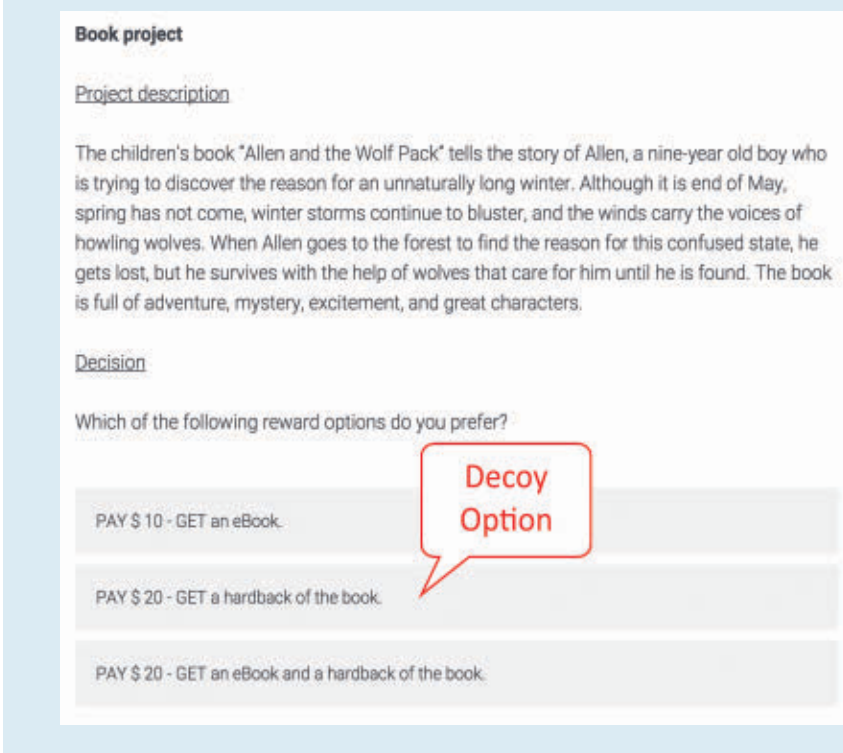
These observations are more than theory. We are being nudged every

day of our lives. Supermarkets position items with the highest markups at eye level to nudge customers into making unplanned purchases. Likewise, supermarkets limit the number of units customers are allowed to buy,

» key insights

- Heuristics and biases influence offline and online behavior.
- User-interface design influences choices, even unintentionally.
- Thorough design and testing can help achieve a designer’s intended behavioral effects.

Figure 1. The decoy effect in reward-based crowdfunding; screenshot shows the decoy condition.



thereby influencing their buying decisions; customers subconsciously anchor their decisions on the maximum number and adjust downward from there, resulting in purchases of greater quantities.³⁶ This effect has been demonstrated in the context of everyday items; for example, introducing a quantity limit of 12 cans of soup helped double the average quantity purchased from 3.3 to seven cans.³⁶ Nudges are not, however, used only by marketers trying to sell more products or services; for example, when asking people to consent to being an organ donor, simply changing defaults can influence people's choices. Setting the default to "dissent," whereby donors have to opt out, rather than "consent" whereby donors have to opt in, can nearly double the percentage of organ donors.¹⁵ These examples show that largely imperceptible nudges are effective in a variety of offline contexts.

As in offline environments, online environments offer no neutral way to present choices. Any user interface, from organizational website to mobile app, can thus be viewed as a digital choice environment.³⁷ Digital choice environments nudge people by deliberately presenting choices or organizing workflows, making digital nudging—"the use of user-interface design elements to guide people's behavior in digital choice environments"³⁷—a powerful tool in any choice architect's toolbox. Choosing the most effective nudge involves trade-offs, however, because predicting the consequences of implementing certain nudges is not always possible.

Existing guidelines for implementing nudges have been developed primarily for offline environments, and digital nudging has only recently begun to attract programmer interest; see, for example, Gregor and Lee-Archer¹⁰ and Weinmann et al.³⁷ In addition, guidelines that are effective offline may not always be directly transferred to a digital context; for example, online users are more willing to disclose information but are also more cautious about accepting default options.² To this end, this article shows how designers can consider the effects of nudges when designing digital choice environments.

Figure 2. The decoy effect in reward-based crowdfunding; adding a decoy option can make another option more attractive.

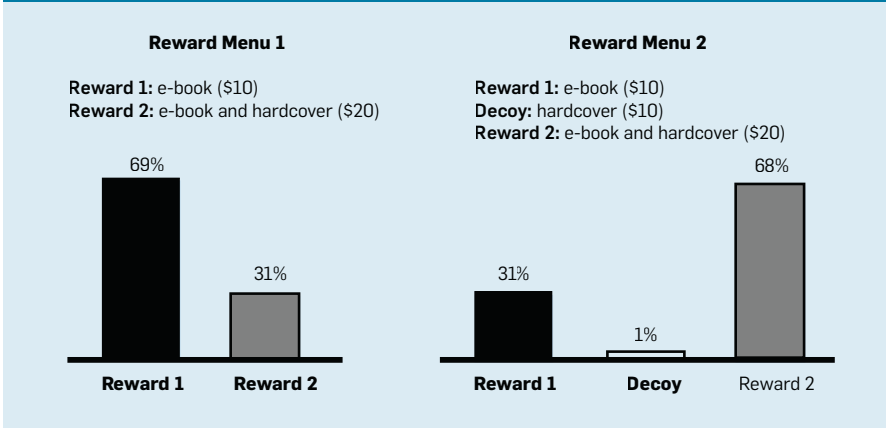


Figure 3. The scarcity effect in reward-based crowdfunding; limiting either reward changes pledging behavior of potential backers.

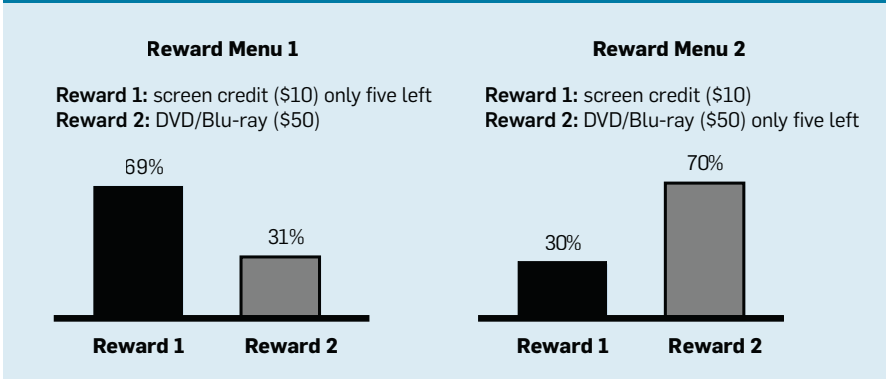
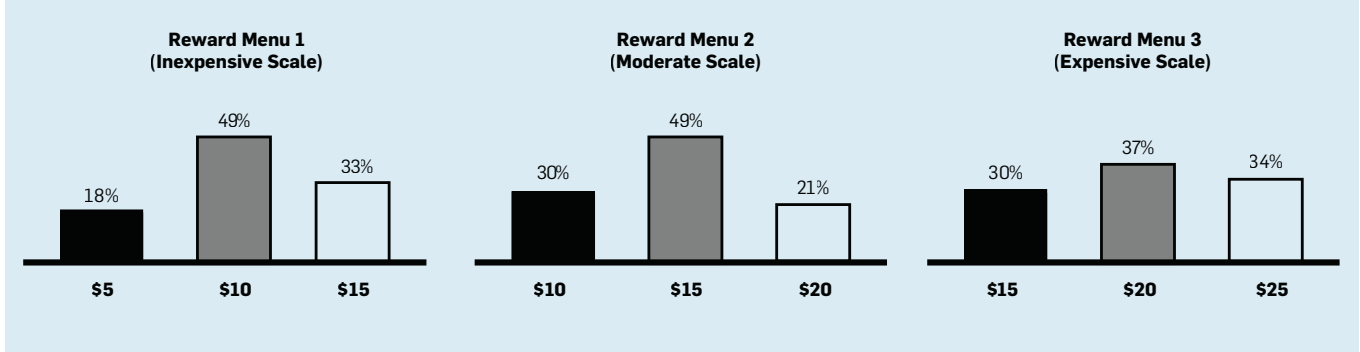


Figure 4. The middle-option bias in reward-based crowdfunding; even when the investment scale is increased, backers tended to select the middle option.

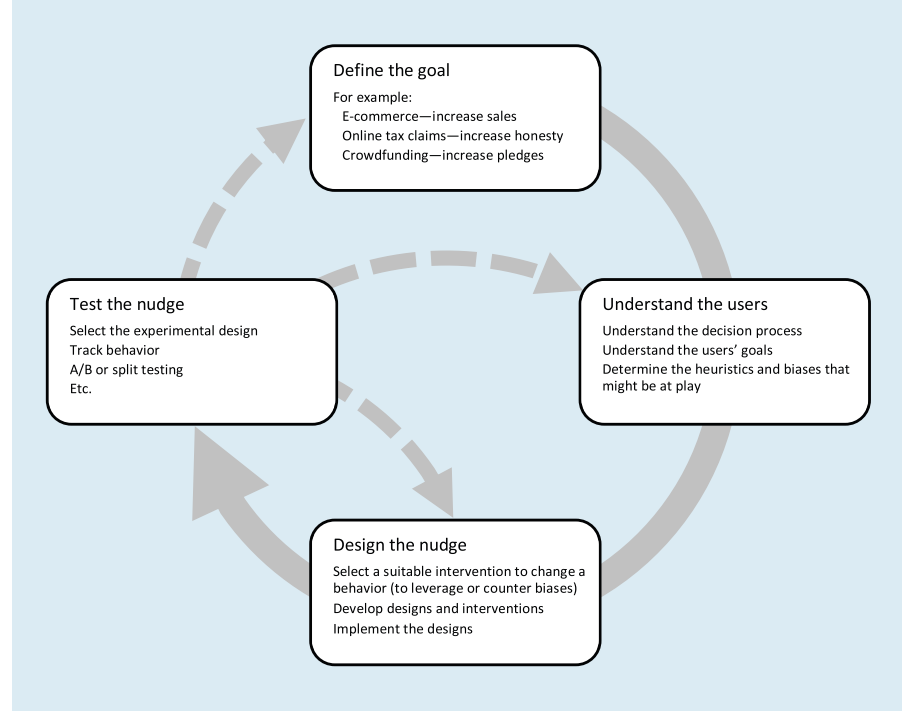


Guiding Choices

As in offline contexts, online decision making is almost always influenced by heuristics and biases; consequently, the concept of digital nudging applies not only to online consumers' decision making but also to various other contexts, from e-health systems to social media apps to organizational information systems. Whereas such factors as presenting reviews or highlighting markdowns are well known for having a strong effect on user behavior in general, digital nudges influence decisions at the point and moment of decision making.^{a,22} In particular, digital nudging works by either modifying what is presented—the content of a choice^{6,35}—or how it is presented—the visualization of a choice—as in, say, changing the design of the user interface.¹⁶ For example, the mobile payment app Square presents a “tipping” option by default, so customers must select “no tipping” if they prefer not to give a tip; this modification is likely an attempt to nudge people into giving tips, motivating them to tip even where tipping is uncommon.³

To illustrate the effects of digital nudges, we briefly explore the results of a series of experiments in the context of reward-based crowdfunding.^{28,33,38} In reward-based crowdfunding, project creators collect small amounts of money from a large number of people, or “backers.” Backers pledge money for projects and receive non-financial

Figure 5. Designing digital nudges follows a cycle; based on Datta and Mullainathan⁵ and Ly et al.¹⁹



rewards in return (such as an e-book).¹ To test how digital nudges influence backers' pledges, researchers at the University of Liechtenstein modified the content and/or visualization of a choice environment to nudge backers toward a particular option through three particular heuristics and biases, known as the “decoy effect,”³³ “scarcity effect,”³⁸ and “middle-option bias.”²⁸

Decoy effect. The decoy effect increases an option's attractiveness by presenting the option alongside an unattractive option no one would reasonably choose—the decoy.¹³ In a study conducted in the context of crowdfunding (N = 96), the researchers showed how decoys can nudge users to select certain rewards;³³ when backers were presented with a choice of receiving

an e-book in return for a \$10 pledge or both an e-book and a hardcover book for a \$20 pledge, most backers chose to pledge \$10. However, when a third option—a decoy nudge—was included that offered only the hardcover book in return for a \$20 pledge (see Figure 1), most backers chose to pledge \$20 to receive both the e-book and the hardcover book. Including the decoy option thus led many backers to move from the \$10 pledge to the \$20 pledge (see Figure 2).

Scarcity effect. People tend to perceive scarce items as more attractive or desirable.⁹ In the context of crowdfunding (N = 166), the researchers showed that limiting the availability of rewards—a “scarcity nudge”—can lead them to choose a particular re-

a Digital nudging, with its focus on the design of digital choice environments, can be viewed as a subset of persuasive computing/technology, which is generally defined as technology designed to change attitudes or behaviors and includes aspects of human-computer interaction beyond interface design.^{8,26}

ward.³⁸ For a fictitious movie project, backers were offered a choice between two rewards: pledge \$10 to be listed in the screen credits or pledge \$50 to receive the movie on a DVD/Blu-ray disc (see Figure 3). When the availability of the low-price reward was limited, 69% of the backers chose that reward, as in Figure 3, left side, whereas when the availability of the high-price reward was limited, 70% chose that reward, as in Figure 3, right side. Merely presenting information about the limited availability of either reward, even the higher-price one, thus caused more backers to choose that reward.

Middle-option bias. People presented with three or more options (ordered sequentially, as by price) tend to select the middle option.⁴ Testing the effect of the middle-option bias in the context of crowdfunding (N = 282), the researchers showed that backers can be nudged into choosing the reward presented in the middle.²⁸ They tested it by varying the pledges of the offered rewards by, in particular, shifting the scales such that Condition 1: \$5, \$10, \$15; Condition 2: \$10, \$15, \$20, and Condition 3:

\$15, \$20, \$25. The researchers told the participants that their pledge would be doubled as a reward if the project would be successful. However, irrespective of the scale, most backers tended to choose the middle option, and by shifting the scales, the researchers could nudge the participants toward selecting rewards associated with higher pledge amounts (see Figure 4).

These examples show that designers can create digital nudges on the basis of psychological principles of human decision making to influence people's online behavior. Unintended effects may arise, however, if designers of digital choice environments are unaware of the principles. For example, in the context of crowdfunding, presenting decoys or limiting the availability of rewards without considering their effect can unintentionally lead backers to select lower-price rewards; that is, as virtually all user-interface design decisions influence user behavior,^{20,30} designers must understand the effects of their designs so they can choose whether to nudge users or reduce the effects of nudges.

Designing a Digital Nudge

While a number of researchers have suggested guidelines for selecting and implementing nudges in offline contexts,^{5,6,16,19,21,31} information systems present unique opportunities for harnessing the power of nudging. For example, Web technologies allow real-time tracking and analysis of user behavior, as well as personalization of the user interface, and both can help test and optimize the effectiveness of digital nudges; moreover, mobile apps can provide a wealth of information about the context (such as location and movement) in which a choice is made. Given these advantages, information systems allow rapid content modification and visualization to achieve the desired nudging effect.

Drawing on guidelines for implementing nudges in offline contexts, we now highlight how designers can create digital nudges by exploiting the inherent advantages of information systems. Just as developing an information system follows a cycle, as in, say, the systems development life cycle—planning, analysis, design, and implementation—so does designing choices to nudge users (see Figure 5)—define the goal, understand the users, design the nudge, and test the nudge. We discuss each step in turn, focusing on the decisions designers must make.

Step 1: Define the goal. Designers must first understand an organization's overall goals and keep them in mind when designing particular choice situations. For instance, the goal of an e-commerce platform is to increase sales, the goal of a governmental taxing authority's platform is to make filing taxes easier and encourage citizens to be honest, and the goal of project creators on crowdfunding platforms is to increase pledges and overall donation amounts. These goals determine how choices are to be designed, particularly the type of choice to be made. For example, subscribing to a newsletter is a binary choice—yes/no, agree/disagree—selecting between items is a discrete choice, and donating monetary amounts is a continuous choice, though it could also be presented as a discrete choice. The type of choice determines the nudge to be used (see the table here). The choice architect, however, must consider not only the goals but also the ethical im-

Applying the digital nudging design cycle (selected examples).

Step 1 Type of choice to be influenced	Step 2 Heuristic/Bias	Step 3 Example design elements and user-interface patterns and possible nudges and mechanisms
Binary (yes/no)	Status quo bias (defaults)	Radio buttons (with default choice)
Discrete choice (such as two products)	Status quo bias (defaults)	Use of defaults in Radio buttons Check boxes Dropdown menus
	Decoy effect	Presentation of decoy option(s) in Radio buttons Check boxes Dropdown menus
	Primacy and recency effect	Positioning of presentation of desired option(s) Earlier (primacy) Later (recency)
	Middle-option bias	Addition of higher- and lower-price alternatives around preferred option Ordering of alternatives Modification of the option scale
Continuous	Anchoring and adjustment	Variation of slider endpoints Use of default slider position Predefined values in text boxes for quantities
	Status quo bias (defaults)	Use of default slider position
Any type of choice	Norms	Display of popularity (social norms) Display of honesty codes (moral norms)
	Scarcity effect (loss aversion)	Use of default slider position

plications of deliberately nudging people into making particular choices, as nudging people toward decisions that are detrimental to them or their wellbeing is unethical and might thus backfire, leading to long-term negative effects for the organization providing the choice.³⁰ In short, overall organizational goals and ethical considerations drive the design of choice situations, a high-level step that influences all subsequent design decisions.

Step 2: Understand the users. People's decision making is susceptible to heuristics and biases. Heuristics, commonly defined as "rules of thumb,"¹⁴ can facilitate human decision making by reducing the amount of information to be processed when addressing simple, recurrent problems. Conversely, heuristics can influence decisions negatively by introducing cognitive biases—systematic errors—when one faces complex judgments or decisions that should require more extensive deliberation.⁷ Researchers have studied a wide range of psychological effects that subconsciously influence people's behavior and decision making.^b In addition to the middle-option bias, decoy effect, and scarcity effect described earlier, common heuristics like the "anchoring-and-adjustment" heuristic, or people being influenced by an externally provided value, even if unrelated; the "availability" heuristic, or people being influenced by the vividness of events that are more easily remembered; and the "representativeness" heuristic, or people relying on stereotypes when encountering and assessing novel situations,³⁴ influence how alternatives are evaluated and what options are ultimately selected. Other heuristics and biases that can have a strong effect on choices include the "status quo bias," or people tending to favor the status quo so they are less inclined to change default options;¹⁸ the "primacy and recency effect," or people recalling options presented first or last more vividly, so those options have a stronger influence on choice;²⁴ and "appeals to

Questions Designers Need to Address

Define goals:

- ▶ What is the use scenario?
- ▶ What are the overall organizational goals?
- ▶ What specific goals are to be achieved in this situation?
- ▶ What are the ethical implications of nudging people into making a certain decision?

Understand the decision process:

- ▶ What are the users' goals?
- ▶ What are the users' decision-making processes?
- ▶ What heuristics might influence users' choices?

Design the nudge:

- ▶ What types of nudges could counter the influence of biases?
- ▶ What types of nudges could increase the influence of biases?
- ▶ What nudges could inadvertently influence users' choices?
- ▶ How can the design of the user interface be modified to include the preferred nudges?
- ▶ How can we analyze users' behavior to adapt the choice environment dynamically?

Test the nudge:

- ▶ How effective are the various nudges?
- ▶ Does the effectiveness differ across users?
- ▶ Do the nudges fit the context and the goals?
- ▶ Do we have a thorough understanding of the users' decision-making process?

norms," or people tending to be influenced by the behavior of others.²³ Understanding these heuristics and biases and the potential effects of digital nudges can thus help designers guide people's online choices and avoid the trap of inadvertently nudging them into decisions that might not align with the organization's overall goals.

Step 3: Design the nudge. Once the goals are defined (see Step 1: Define the goal) and the heuristics and biases are understood (see Step 2: Understand the users), the designer can select the appropriate nudging mechanism(s) to guide users' decisions in the designer's intended direction. Common nudging frameworks a designer could use to select appropriate nudges include the Behavior Change Technique Taxonomy,²¹ NUDGE,³¹ MINDSPACE,⁶ and Tools of a Choice Architecture.¹⁶ Selecting an appropriate nudge and how to implement it through available design elements, or user-interface patterns, is determined by both the type of choice to be made—binary, discrete, or continuous^c—and

the heuristics and biases at play; see the table for examples. For example, a commonly used nudge in binary choices is to preselect the desired option to exploit the status quo bias. When attempting to nudge people in discrete choices, choice architects can choose from a variety of nudges to nudge people toward a desired option. For example, in the context of crowdfunding, with the goal of increasing pledge amounts, choice architects could present the desired reward option as the default option; add (unattractive) choices as decoys; present the desired option first or last to leverage primacy and recency effects; or arrange the options so as to present the preferred reward as the middle option. When attempting to nudge people in continuous choices (such as when soliciting monetary donations), choice architects could pre-populate input fields (text boxes) with a particular value so as to exploit the "anchoring and adjustment" effect. Likewise, when using a slider to elicit numerical responses, the position of the slider and the slider endpoints serve as implicit anchors. Presenting others' choices next to rewards to leverage people's tendency to conform to norms or presenting limited availability of rewards to exploit the

b See Stanovich²⁰ for a taxonomy of rational thinking errors and biases; see also Wikipedia for an extensive list of cognitive biases that influence people's online and offline behavior (https://en.wikipedia.org/wiki/List_of_cognitive_biases).

c In most cases, the type of decision is an externality, and many decisions allow for only one type; for example, consenting to something (whether organ donation or signing up for a newsletter) would normally always be a binary choice—yes/no.

scarcity effect can be used to nudge people in binary, discrete, or continuous choices.

As the same heuristic can be addressed through multiple nudges, in most situations, designers have a variety of “nudge implementations” at their disposal. Unlike in offline environments, implementing nudges in digital environments can be done at relatively low cost, as system designers can easily modify a system’s user interface (such as by setting defaults, displaying/hiding design elements, or providing information on others’ pledges). Likewise, digital environments enable dynamic adjustment of the options presented on the basis of certain attributes or characteristics of the individual user (such as when a crowdfunding platform presents particular rewards depending on the backers’ income, gender, or age). Notwithstanding the choice of nudges, designers should follow commonly accepted design guidelines for the respective platforms (such as Apple’s Human Interface Guidelines and Microsoft’s Universal Windows Platform design guidelines) to ensure consistency and usability.

Step 4: Test the nudge. Digital environments allow alternative designs to be generated easily, so their effects can be tested quickly, especially when designing websites. The effectiveness of digital nudges can be tested through online experiments (such as A/B testing and split testing). Testing is particularly important, as the effectiveness of a nudge is likely to depend on both the context and goal of the choice environment and the target audiences. For example, a digital nudge that works well in one context (such as a hotel-booking site like <https://www.booking.com>) may not work as well in a different context (such as a car-hailing service like <https://www.uber.com>); such differences may be due to different target users, the unique nature of the decision processes, or even different layouts or color schemes on the webpages; a hotel may use colors and shapes that evoke calmness and cleanliness, whereas a car-hailing service may use colors and shapes that evoke speed and efficiency. As choice



Big-data analytics can be used to analyze behavioral patterns observed in real time to infer users’ personalities, cognitive styles, or even emotional states.



architects have various nudge implementations at their disposal, thorough testing is thus imperative for finding the nudge that works best for a given context and users.

Especially in light of the increasing focus on integrating user-interface design and agile methodologies, using discount usability techniques (such as heuristic evaluation, as introduced by Nielsen²⁵) is often recommended to support rapid development cycles (see, for example, Jurca et al.¹⁷). Likewise, agile methodologies include the quick collection of feedback from real users. However, such feedback from conscious evaluations should be integrated with caution because the effects of nudges are based on subconscious influences on behavior, and experimental evaluations can provide more reliable results. If a particular nudge does not produce the desired effect, a first step for system designers is to evaluate the nudge implementation to determine whether the nudge is, say, too obvious or not obvious enough (see Step 3: Design the nudge). In some instances, though, reexamining the heuristics or biases that influence the decision-making process (see Step 2: Understand the users) or even returning to Step 1: Define the goal and redefining the goals may be necessary (see the sidebar, “Questions Designers Need to Address”).

Conclusion

Understanding digital nudges is important for the overall field of computing because user-interface designers create most of today’s choice environments. With increasing numbers of people making choices through digital devices, user-interface designers become choice architects who knowingly or unknowingly influence people’s decisions. However, user-interface design often focuses primarily on usability and aesthetics, neglecting the potential behavioral effects of alternative designs. Extending the body of knowledge of the computing profession through insights into digital nudging will help choice architects leverage the effects of digital nudges to support organizational goals. Choice architects can use the digital nudging

design cycle we have described here to deliberately develop such choice environments.

One final note of caution is that the design of nudges should not follow a “one-size-fits-all” approach, as their effectiveness often depends on a decision maker’s personal characteristics.¹⁶ In digital environments, characteristics of users and their environment can be inferred from a large amount of data, allowing nudges to be tailored. System designers might design the choice environment to be adaptive on the basis of, say, users’ past decisions or demographic characteristics. Likewise, big-data analytics can be used to analyze behavioral patterns observed in real time to infer users’ personalities, cognitive styles, or even emotional states.¹² For example, Bayesian updating can be used to infer cognitive styles from readily available clickstream data and automatically match customers’ cognitive styles to the characteristics of the website (such as through “morphing”¹¹). Designers of digital choice environments can attempt to “morph” digital nudges on the basis of not only the organizational goals but also users’ personal characteristics.

Any designer of a digital choice environment must be aware of its effects on users’ choices. In particular, when developing a choice environment, designers should carefully define the goals, understand the users, design the nudges, and test those nudges. Following the digital-nudging design cycle we have laid out here can help choice architects achieve their organizational goals by understanding both the users and the potential nudging effects so intended effects can be maximized and/or unintended effects minimized.

Acknowledgments

This work was partially supported by research grants from the University of Liechtenstein (Project No. wi-2-14), City University of Hong Kong (Project No. 7004563), and City University of Hong Kong’s Digital Innovation Laboratory in the Department of Information Systems. We wish to thank Joseph S. Valacich for valuable comments on earlier versions, as well as the anonymous reviewers for their insightful comments. 

References

1. Belleflamme, P., Lambert, T., and Schwenbacher, A. Individual crowdfunding practices. *Venture Capital* 15, 4 (May 2013), 313–333.
2. Benartzi, S. and Lehrer, J. *The Smarter Screen: Surprising Ways to Influence and Improve Online Behavior*. Penguin Books, New York, 2015.
3. Carr, A. *How Square Register’s UI Guilt’s You Into Leaving Tips*. Fast Company, New York, 2013; <http://www.fastcodesign.com/3022182/innovation-by-design/how-square-registers-ui-guilt-s-you-into-leaving-tips>
4. Christenfeld, N. Choices from identical options. *Psychological Science* 6, 1 (Jan. 1995), 50–55.
5. Datta, S. and Mullainathan, S. Behavioral design: A new approach to development policy. *Review of Income and Wealth* 60, 1 (Feb. 2014), 7–35.
6. Dolan, P., Hallsworth, M., Halpern, D., King, D., Metcalfe, R., and Vlaev, I. Influencing behaviour: The Mindspace way. *Journal of Economic Psychology* 33, 1 (Feb. 2012), 264–277.
7. Evans, J.S.B.T. Dual-processing accounts of reasoning, judgment, and social cognition. *Annual Review of Psychology* 59, 1 (Jan. 2008), 255–278.
8. Fogg, B.J. *Persuasive Technology: Using Computers to Change What We Think and Do*. Elsevier, Oxford, U.K., 2003.
9. Fromkin, H.L. and Snyder, C.R. The search for uniqueness and valuation of scarcity. Chapter 3 in *Social Exchange*, K.J. Gergen, M.S. Greenberg, and R.H. Willis, Eds. Springer U.S., Boston, MA, 1980, 57–75.
10. Gregor, S. and Lee-Archer, B. The digital nudge in the Social Security Administration. *International Social Security Review* 69, 3–4 (July–Dec. 2016), 63–83.
11. Hauser, J.R., Urban, G.L., Liberali, G., and Braun, M. Website morphing. *Marketing Science* 28, 2 (Mar. 2009), 202–223.
12. Hibbeln, M., Jenkins, J.L., Schneider, C., Valacich, J.S., and Weinmann, M. How is your user feeling? Inferring emotion through human-computer interaction devices. *MTS Quarterly* 41, 1 (Mar. 2017), 1–21.
13. Huber, J., Payne, J.W., and Puto, C. Adding asymmetrically dominated alternatives: Violations of regularity and the similarity hypothesis. *Journal of Consumer Research* 9, 1 (June 1982), 90.
14. Hutchinson, J.M.C. and Gigerenzer, G. Simple heuristics and rules of thumb: Where psychologists and behavioural biologists might meet. *Behavioural Processes* 69, 2 (May 2005), 97–124.
15. Johnson, E.J. and Goldstein, D.G. Do defaults save lives? *Science* 302, 5649 (Nov. 2003), 1338–1339.
16. Johnson, E.J., Shu, S.B., Dellaert, B.G.C. et al. Beyond nudges: Tools of a choice architecture. *Marketing Letters* 23, 2 (June 2012), 487–504.
17. Jurca, G., Hellmann, T.D., and Maurer, F. Integrating agile and user-centered design: A systematic mapping and review of evaluation and validation studies of agile-UX. In *Proceedings of the 2014 Agile Conference* (Kissimmee, FL, July 28–Aug. 1). IEEE Computer Society, Piscataway, NJ, 2014.
18. Kahneman, D., Knetsch, J.L., and Thaler, R.H. Anomalies: The endowment effect, loss aversion, and status quo bias. *The Journal of Economic Perspectives* 5, 1 (Winter 1991), 193–206.
19. Ly, K., Mazar, N., Zhao, M., and Soman, D. *A Practitioner’s Guide to Nudging*. Rotman School of Management Working Paper No. 2609347, May 2013, 1–28; <https://ssrn.com/abstract=2609347>
20. Mandel, N. and Johnson, E.J. When webpages influence choice: Effects of visual primes on experts and novices. *Journal of Consumer Research* 29, 2 (Sept. 2002), 235–245.
21. Michie, S., Richardson, M., Johnston, M. et al. The behavior change technique taxonomy (v1) of 93 hierarchically clustered techniques: Building an international consensus for the reporting of behavior change interventions. *Annals of Behavioral Medicine* 46, 1 (Aug. 2013), 81–95.
22. Miesler, L., Scherrer, C., Seiler, R., and Bearth, A. Informational nudges as an effective approach in raising awareness among young adults about the risk of future disability. *Journal of Consumer Behaviour* 16, 1 (Jan./Feb. 2017), 15–22.
23. Muchnik, L., Aral, S., and Taylor, S.J. Social influence bias: A randomized experiment. *Science* 341, 6146 (Aug. 2013), 647–651.
24. Murdock, B.B. The serial position effect of free recall. *Journal of Experimental Psychology* 64, 2 (Nov. 1962), 482–488.
25. Nielsen, J. Usability engineering at a discount. In *Proceedings of the Third International Conference on Human-Computer Interaction on Designing and Using Human-Computer Interfaces and Knowledge-Based Systems* (Boston, MA). Elsevier, New York, 1989, 394–401.
26. Oinas-Kukkonen, H. and Harjumaa, M. Persuasive systems design: Key issues, process model, and system features. *Communications of the Association for Information Systems* 24, Article 28 (Mar. 2009), 485–500.
27. Simon, H.A. A behavioral model of rational choice. *Quarterly Journal of Experimental Psychology* 69, 1 (Feb. 1955), 99–118.
28. Simons, A., Weinmann, M., Tietz, M., and Brocke, vom, J. Which reward should I choose? Preliminary evidence for the middle-option bias in reward-based crowdfunding. In *Proceedings of the Hawaii International Conference on System Sciences* (Hilton Waikoloa Village, HI, Jan. 4–7). University of Hawaii, Manoa, HI, 2017, 4344–4353.
29. Stanovich, K.E. *Rationality and the Reflective Mind*. Oxford University Press, New York, 2011.
30. Sunstein, C.R. Nudging and choice architecture: Ethical considerations. *Yale Journal on Regulation* 32, 2 (2015), 413–450.
31. Thaler, R.H. and Sunstein, C.R. *Nudge: Improving Decisions About Health, Wealth, and Happiness*. Yale University Press, New Haven, CT, and London, U.K., 2008.
32. Thaler, R.H., Sunstein, C.R., and Balz, J.P. Choice Architecture. *SSRN Electronic Journal* (2010), 1–18; <https://ssrn.com/abstract=1583509>
33. Tietz, M., Simons, A., Weinmann, M., and Brocke, vom, J. The decoy effect in reward-based crowdfunding: Preliminary results from an online experiment. In *Proceedings of the International Conference on Information Systems* (Dublin, Ireland, Dec. 11–14). Association for Information Systems, Atlanta, GA, 2016, 1–11.
34. Tversky, A. and Kahneman, D. Judgment under uncertainty: Heuristics and biases. *Science* 185, 4157 (Sept. 1974), 1124–1131.
35. Tversky, A. and Kahneman, D. The framing of decisions and the psychology of choice. *Science* 211, 4481 (Jan. 1981), 453–458.
36. Wansink, B., Kent, R.J., and Hoch, S. An anchoring and adjustment model of purchase quantity decisions. *Journal of Marketing Research* 35, 1 (Feb. 1998), 71–81.
37. Weinmann, M., Schneider, C., and Brocke, vom, J. Digital nudging. *Business & Information Systems Engineering* 58, 6 (Dec. 2016), 433–436.
38. Weinmann, M., Simons, A., Tietz, M., and Brocke, vom, J. Get it before it’s gone? How limited rewards influence backers’ choices in reward-based crowdfunding. In *Proceedings of the International Conference on Information Systems* (Seoul, South Korea, Dec. 10–13). Association for Information Systems, Atlanta, GA, 2017, 1–10.

Christoph Schneider (christoph.schneider@cityu.edu.hk) is an assistant professor in the Department of Information Systems at City University of Hong Kong, Kowloon, Hong Kong SAR.

Markus Weinmann (markus.weinmann@uni.li) is an assistant professor in the Department of Information Systems at the University of Liechtenstein, Vaduz, Liechtenstein.

Jan vom Brocke (jan.vom.brocke@uni.li) is a professor of information systems, the Hilti Chair of Business Process Management, Director of the Institute of Information Systems, and Vice President for Research and Innovation at the University of Liechtenstein, Vaduz, Liechtenstein.

Copyright held by the authors.



Watch the authors discuss their work in this exclusive *Communications* video. <https://cacm.acm.org/videos/digital-nudging>

DOI:10.1145/3213770

Performance measurements often go wrong, reporting surface-level results that are more marketing than science.

BY JOHN OUSTERHOUT

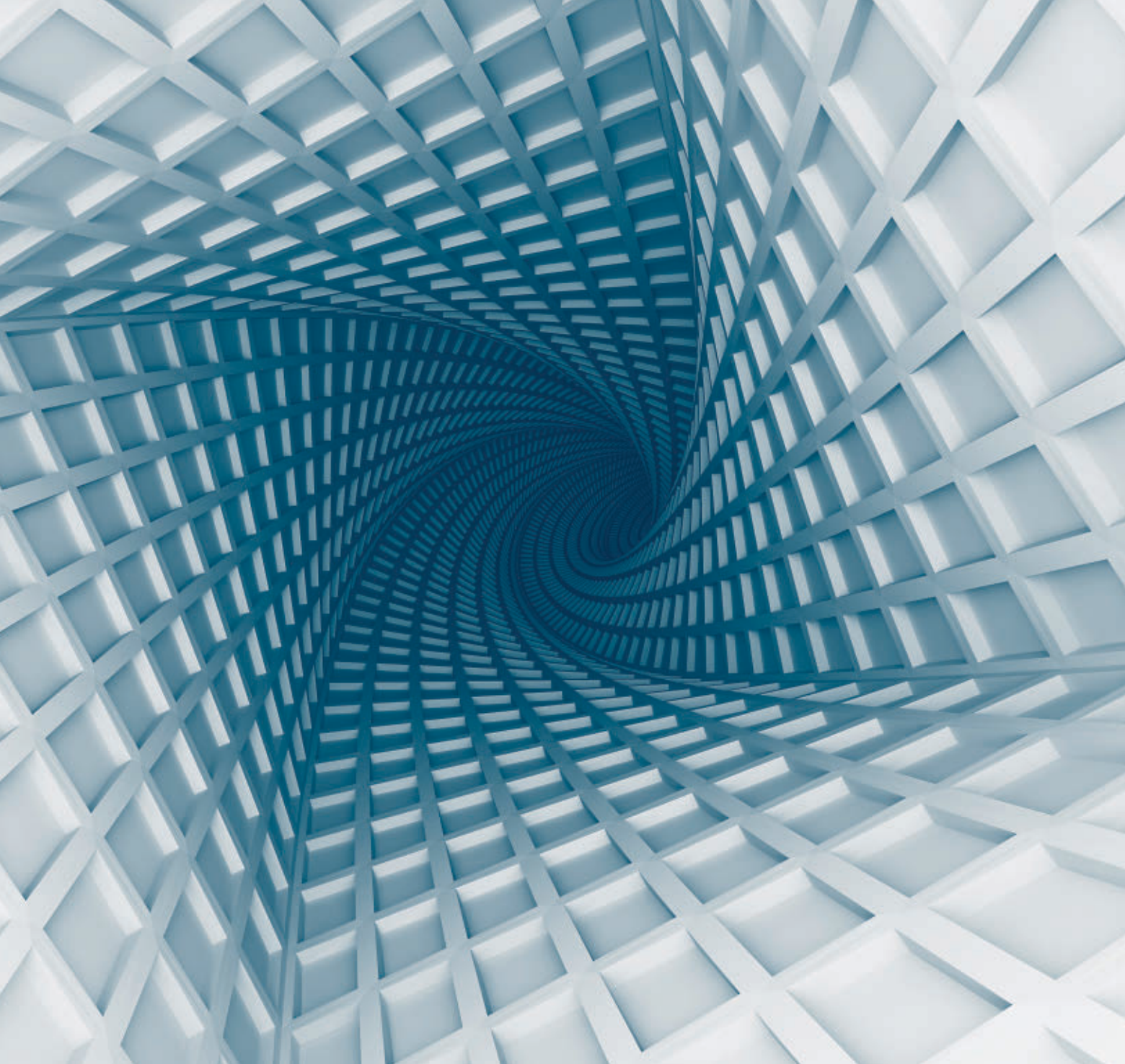
Always Measure One Level Deeper

PERFORMANCE MEASUREMENT IS one of the most important parts of software development. In academic research a thorough performance evaluation is considered essential for many publications to prove the value of a new idea. In industry, performance evaluation is necessary to maintain a high level of performance across the lifetime of a product. For example, cloud services promise to maintain particular performance levels; service providers must thus be able to detect when performance drops below acceptable levels and quickly identify and fix the problem.

A good performance evaluation provides a deep understanding of a system's behavior, quantifying not only the overall behavior but also its internal mechanisms and policies. It explains why a system behaves the way it does, what limits that behavior, and what problems must be addressed in order to

» key insights

- Performance measurement is less straightforward than it might seem; it is easy to believe results that are incorrect or misleading and overlook important system behaviors.
- The key to good performance measurement is to make many more measurements besides the ones you think will be important; it is crucial to understand not just the system's performance but also why it performs that way.
- Performance measurement done well results in new discoveries about the system being measured and new intuition about system behavior for the person doing the measuring.



improve the system. Done well, performance evaluation exposes interesting system properties that were not obvious previously. It not only improves the quality of the system being measured but the developer's intuition, resulting in better systems in the future.

Unfortunately, there is no widespread understanding or agreement as to how to measure performance. Performance evaluation is rarely taught in computer science classes. And new faculty lack well-developed performance-measurement skills, making it difficult for them to train their students. The

only way to become expert is through trial and error.

As a result, performance measurement is often done poorly, even by experienced developers. For example, if you have written a conference paper on a software system, it probably unfolded like this: The system implementation took longer than expected, so performance evaluation could not begin until a week or two before the paper submission deadline. The first attempts to run benchmarks resulted in system crashes, so you spent the next week fixing bugs. At this point the benchmarks

ran, but the system's performance was not much better than the comparison systems. You tried different experiments, hoping to find one where the system looked good; this exposed yet more bugs that had to be fixed. Time was running out, so you stopped measuring as soon as you found an experiment that produced positive results. The paper focused on this experiment, omitting the results that were less favorable. There were a few things about these results that did not make complete sense, but you did your best to come up with plausible explanations

for them. There was not enough time to validate or double-check the numbers, and you could only hope there were not too many errors.

Measurements gathered this way are likely incomplete, misleading, or even erroneous. This article describes how to conduct performance measurement well. I first discuss five mistakes that account for most of the problems with performance measurements, all of which occurred in the scenario I just outlined. I then spell out four rules to follow when evaluating performance. These rules will help you avoid the mistakes and produce high-quality performance evaluations. Finally, I offer four suggestions about infrastructure to assist in performance evaluation.

The most important idea overall, as reflected in this article's headline, is to dig beneath the surface, measuring the system in depth and detail from multiple angles to create a complete and accurate understanding of performance.

Most Common Mistakes

When performance measurements go wrong, it is usually due to five common mistakes:

Mistake 1: Trusting the numbers.

Engineers are easily fooled during performance measurements because measurement bugs are not obvious. Engineers are used to dealing with functional bugs, which tend to be noticeable because they cause the system to crash or misbehave. If the system produces the desired behavior, it is probably working. Engineers tend to apply the same philosophy to performance measurements; if performance numbers are being generated and the system is not crashing, they assume the numbers are correct.

Performance-measurement code is just as likely to have bugs as any other code, but the bugs are less obvious. Most bugs in performance-measurement code do not cause crashes or prevent numbers from appearing; they simply produce incorrect numbers. There is no easy way to tell from a number whether it is right or wrong, so engineers tend to assume the numbers are indeed correct. This is a mistake. There are many ways for errors to creep into performance measurements. There may be bugs in the benchmarks or test applications, so the measured behav-

Performance measurements should be considered guilty until proven innocent.

ior is not the desired behavior. There may be bugs in the code that gathers metrics and processes them, as when, say, a clock is read at the wrong time or the 99th percentile is miscomputed. The system being measured may have functional bugs. And, finally, the system may have performance bugs, so the measurements do not reflect the system's true potential.

I have been involved in dozens of performance-measurement projects and cannot recall a single one in which the first results were correct. In each case there were multiple problems from the list just outlined. Only after working through them all did my colleagues and I obtain measurements that were meaningful.

Mistake 2: Guessing instead of measuring. The second common mistake is to draw conclusions about a system's performance based on educated guesses, without measurements to back them up. For example, I found the following explanation in a paper I reviewed recently: "... throughput does not increase with the number of threads ... This is because the time taken to traverse the relatively long linked list bounds server performance." There was no indication that the authors had measured the actual length of the list or the time taken to traverse it, yet they stated their conclusion as fact. I frequently encounter unsubstantiated conclusions in papers; there were at least five other occurrences in the paper with the quote.

Educated guesses are often correct and play an important role in guiding performance measurement; see Rule 3 (Use your intuition to ask questions, not answer them). However, engineers' intuition about performance is not reliable. When my students and I designed our first log-structured file system,⁴ we were fairly certain that reference patterns exhibiting locality would result in better performance than those without locality. Fortunately, we decided to measure, to be sure. To our surprise, the workloads with locality behaved worse than those without. It took considerable analysis to understand this behavior. The reasons were subtle, but they exposed important properties of the system and led us to a new policy for garbage collection that improved the system's performance significantly. If we

had trusted our initial guess, we would have missed an important opportunity for performance improvement.

It is unsafe to base conclusions on intuition alone, yet engineers do it all the time. A common mistake is for an engineer to hypothesize that a particular data structure is too slow and then replace it with a new data structure the engineer believes will be faster. If the problem is not verified by measuring performance, there is a good chance the optimization will not improve performance. The code change will simply waste a lot of time and probably introduce unnecessary complexity.

When I find a guess presented as fact and ask for justification, I sometimes get this response: “What else could it possibly be?” But this is a cop-out, suggesting it is up to others to prove the theory wrong and OK to make unsubstantiated claims until someone else proves them false. In some cases the person making the comment feels a process of elimination had been used, ruling out all possible alternatives. Unfortunately, a process of elimination is not reliable in performance evaluation, because it is not possible to know with certainty that every possible cause has been considered. Many factors can influence performance, and the ultimate cause of behavior is often something non-obvious, meaning a process of elimination will not consider it. It is unsafe to present an explanation as fact unless measurements confirm the specific behavior(s).

Mistake 3: Superficial measurements. Most performance measurements I see are superficial, measuring only the outermost visible behavior of a system (such as the overall running time of an application or the average latency of requests made to a server). These measurements are essential, as they represent the bottom line by which a system is likely to be judged, but they are not sufficient. They leave many questions unanswered (such as “What are the limits that keep the system from performing better?” and “Which of the improvements had the greatest impact on performance?”). In order to get a deep understanding of system performance, the internal behavior of a system must be measured, in addition to its top-level performance.

Superficial measurements are often

combined with Mistake 1 (Trusting the numbers) and Mistake 2 (Guessing instead of measuring); the engineers measure only top-level performance, assume the numbers are correct, and then invent underlying behaviors to explain the numbers. For example, I found the following claim in a paper I reviewed recently (system names obscured to preserve author anonymity): “Unlike YYY, XXX observes close-to-linear-throughput scaling with more publishers due to its lock-free resolution of write-write contentions.” The paper measured scaling, but there were no measurements of write-write contention, and systems XXX and YYY differed in many ways, so other explanations were possible for the performance difference.

Mistake 4: Confirmation bias. Performance measurement is rarely indifferent; when you measure performance, you are probably hoping for a particular outcome. If you have just built a new system or improved an existing one, you probably hope the performance measurements will show your ideas were good ones. If the measurements turn out well, it increases the likelihood your paper will be accepted or your boss will be impressed.

Unfortunately, such hope results in a phenomenon called “confirmation bias.”¹ Confirmation bias causes people to select and interpret data in a way that supports their hypotheses. For example, confirmation bias affects your level of trust. When you see a result that supports your hypothesis, you are more likely to accept the result without question. In contrast, if a measurement suggests your new approach is not performing well, you are more likely to dig deeper to understand exactly what is happening and perhaps find a way to fix the problem. This means that an error in a positive result is less likely to be detected than is an error in a negative result.

When choosing benchmarks, you are more likely to choose ones that produce the desired results and less likely to choose ones that show the weaknesses of your approach. For example, a recent paper described a new network protocol and compared it to previous proposals. The previous proposals had all measured latency using the 99th-percentile worst case, but this par-

ticular paper measured at the median. The results appeared favorable for the new proposal. My students reran the measurements for the new protocol and discovered its 99th-percentile latency was significantly worse than the comparison protocols. We wondered if the paper’s authors had intentionally switched metrics to exaggerate the performance of their protocol.

Confirmation bias also affects how you present information. You are more likely to include results that support your hypothesis and downplay or omit results that are negative. For example, I frequently see claims in papers of the form: “XXX is up to 3.5x faster than YYY.” Such claims cherry-pick the best result to report and are misleading because they do not indicate what performance can be expected in the common case. Statements like this belong in late-night TV commercials, not scientific papers.

If applied consciously, bias is intellectually dishonest. Even if not applied consciously, it can cause results to be reported in a way that is more marketing than science; it sounds like you are trying to sell a product rather than uncover the truth about a system’s behavior. Confirmation bias makes it more likely that results will be inaccurate (because you did not find bugs) or misleading (because you did not present all relevant data).

Mistake 5: Haste. The last mistake in performance evaluation is not allowing enough time. Engineers usually underestimate how long it takes to measure performance accurately, so they often carry out evaluations in a rush. When this happens, they will make mistakes and take shortcuts, leading to all the other mistakes.

The time issue is particularly problematic when working under a deadline (such as for a conference publication). There is almost always a rush to meet the submission deadline. The system implementation always takes longer than expected, delaying the start of performance measurement; there is often only a week or two for evaluation before the submission deadline, resulting in a sloppy evaluation. In principle, authors could keep working on the measurements while waiting for the paper to be reviewed, but in practice this rarely happens. In-

stead, they tell themselves: “Let’s not spend more time on the paper until we see whether it is accepted.” Once the paper is accepted, there are only a few weeks before the deadline for final papers, so there is yet another rush.

Keys to High-Quality Performance Analysis

Consider four rules that are likely to prevent the mistakes from the preceding section and lead to trustworthy and informative evaluations:

Rule 1: Allow lots of time. The first step toward high-quality performance measurements is to allow enough time. If you are measuring a non-trivial system, you should plan on at least two to three months. I tell my graduate students to aim for a complete set of preliminary measurements at least one month before the submission deadline; even this is barely enough time to find and fix problems with both the measurements and the system.

Performance analysis is not an instantaneous process like taking a picture of a finished artwork. It is a long and drawn-out process of confusion, discovery, and improvement. Performance analysis goes through several phases, each of which can take anywhere from a few days to a few weeks. First, you must add instrumentation code to the system to record the desired metrics. You must then get benchmark applications running, either by writing them or by downloading and installing existing programs. Running benchmarks will probably stress the system enough to expose bugs, and you will need to then track down and fix them. Eventually, the system will run well enough to start producing performance numbers. However, these numbers will almost certainly be wrong. The next step is to find and fix bugs in the measurements. Once you have verified the accuracy of the measurements, you will start to uncover problems with the system itself. As you look over the performance measurements, you will probably uncover additional functional bugs. Once they have been fixed, you can start analyzing the performance in depth. You will almost certainly discover opportunities to improve performance, and it is important to have enough time to make these im-

provements. You will encounter many things that do not make sense; in order to resolve them, you will need to add new metrics and validate them. To get the best results, you must iterate several times improving the metrics, measuring performance, and improving the system.

Rule 2: Never trust a number generated by a computer. Under Mistake 2 (Guessing instead of measuring), I discussed how it is tempting to believe performance numbers, even though they are often wrong. The only way to eliminate this mistake is to distrust every measurement until it has been carefully validated. Performance measurements should be considered guilty until proven innocent. When students come to me with measurements, I often challenge them by asking: “Suppose I said I don’t believe these measurements. What can you say to convince me that they are correct?”

The way to validate a measurement is to find different ways to measure the same thing:

Take different measurements at the same level. For example, if you are measuring file-system throughput, do not measure just the throughput seen by a user application; also measure the throughput observed inside the operating system (such as at the file block cache). These measurements should match;

Measure the system’s behavior at a lower level to break down the factors that determine performance, as I discuss later under Rule 4 (Always measure one level deeper);

Make back-of-the-envelope calculations to see if the measurements are in the ballpark expected; and

Run simulations and compare their results to measurements of the real implementation.

As you begin collecting measurements, compare them and be alert for inconsistencies. There will almost always be things that do not make sense. When something does not make complete sense, stop and gather more data. For example, in a recent measurement of a new network transport protocol, a benchmark indicated that a server could handle no more than 600,000 packets per second. However, my colleagues and I had seen servers process more than 900,000 packets per second

with other protocols and believed the new protocol was at least as efficient as the old ones. We decided to gather additional data. As a result, we discovered a bug in the flow-control mechanism on the client side: clients were not transmitting data fast enough to keep the server fully loaded. Fixing the bug improved performance to the level we expected.

Further analysis will sometimes show the unexpected behavior is correct, as in the log-structured file system example discussed under Mistake 2 (Guessing instead of measuring); such situations are usually interesting, and you will learn something important as you resolve the contradiction. In my experience, initial performance measurements are always riddled with contradictions and things we don’t understand, and resolving them is always useful; either we fix a problem or we deepen our understanding of the system.

Above all, do not tolerate anything you do not understand. Assume there are bugs and problems with every measurement, and your job is to find and fix them. If you do not find problems, you should feel uneasy, because there are probably bugs you missed. Curmudgeons make good performance evaluators because they trust nothing and enjoy finding problems.

Rule 3: Use your intuition to ask questions, not to answer them. Intuition is a wonderful thing. As you accumulate knowledge and experience in an area, you will start having gut-level feelings about a system’s behavior and how to handle certain problems. If used properly, such intuition can save significant time and effort. However, it is easy to become overconfident and assume your intuition is infallible. This leads to Mistake 2 (Guessing instead of measuring).

The best way to use intuition is to identify promising areas for further exploration. For example, when looking over performance measurements, ask yourself if they make sense. How does the performance compare to what you expected? Does it seem too good to be true? Does the system scale more poorly than you had hoped? Does a curve jump unexpectedly when you expected it to be smooth? Do some benchmarks exhibit behavior that is dramatically

different from others? Consider anything that does not match your intuition a red flag and investigate it, as described in Rule 2 (Never trust a number generated by a computer). Intuition can be very helpful in identifying problems.

Intuition is great for directing attention but not reliable enough to make decisions on it alone. Intuition should always be validated with data before making decisions or claims. If your intuition suggests why a particular result is occurring, follow it up with measurements that prove or disprove the intuition. Draw conclusions based on the measurements, not the intuition, and include some of the measured data in the conclusion, so others know it is not just a guess.

If you continually form intuitions and then test them you will gain knowledge that helps you form better intuition in the future. Every false intuition means there was something you did not fully understand; in the process of testing it and discovering why it is false, you will learn something useful.

Rule 4: Always measure one level deeper. If you want to understand the performance of a system at a particular level, you must measure not just that level but also the next level deeper. That is, measure the underlying factors that contribute to the performance at the higher level. If you are measuring overall latency for remote procedure calls, you could measure deeper by breaking down that latency, determining how much time is spent in the client machine, how much time is spent in the network, and how much time is spent on the server. You could also measure where time is spent on the client and server. If you are measuring the overall throughput of a system, the system probably consists of a pipeline containing several components. Measure the utilization of each component (the fraction of time that component is busy). At least one component should be 100% utilized; if not, it should be possible to achieve a higher throughput.

Measuring deeper is the best way to validate top-level measurements and uncover bugs. Once you have collected some deeper measurements, ask yourself whether they seem consistent with the top-level measurements and with each other. You will almost cer-



Curmudgeons make good performance evaluators because they trust nothing and enjoy finding problems.



tainly discover things that do not make sense; make additional measurements to resolve the contradictions. For example, in a recent analysis of a distributed transaction processing system, deeper measurements by my students included network throughput and disk throughput. We were surprised to see that the network throughput was greater than the disk throughput; this did not make sense, since every byte had to pass through both the network and the disk. It turned out that the disk subsystem had been configured with no limit on queue length; the disk was not keeping up, and its output queue was growing without bound. Once the students set a limit on queue length, the rest of the system throttled itself to match the disk throughput. Unfortunately, this meant our initial measurements of overall throughput were too optimistic.

Measuring deeper will also indicate whether you are getting the best possible performance and, if not, how to improve it. Use deeper measurements to find out what is limiting performance. Try to identify the smallest elements that have the greatest impact on overall performance. For example, if the overall metric is latency, measure the individual latencies of components along the critical path; typically, there will be a few components that account for most of the overall latency. You can then focus on optimizing those components.

In recent measurements of a new network transport, one of my students found that round-trip tail latency was higher than our simulations had predicted. The student measured software latency in detail on both the sending and the receiving machines but found nothing that could account for the high tail latency. At this point we were about to conclude that the delays must be caused by the network switch. What else could it be? This would have been Mistake 2 (Guessing instead of measuring). Before giving up, we decided to dig deeper and measure precise timings for each individual packet. The measurements surprised us, showing that outlier delays were not isolated events. Delay tended to build up over a series of packets, affecting all of the packets from a single sender over a relatively long time interval, including packets for different destinations. This was a crucial clue. After several additional measure-

ments, the student discovered that long queues were building up in the sender's network interface due to a software bug. The transport included code to estimate the queue length and prevent queue buildup, but there was a bug in the estimator caused by underflow of an unsigned integer. The underflow was easy to fix, at which point tail latency dropped dramatically. Not only did this process improve the system's performance, it taught us an important lesson about the risks of unsigned integers.

Another way to measure deeper is to consider more detail. Instead of just looking at average values, graph the entire distribution and noodle over the shape to see if it provides useful information. Then look at some of the raw data samples to see if there are patterns. In one measurement of RPC latency, a student found that the average latency was higher than we expected. The latency was not intolerably high, and it would have been easy to simply accept this level of performance. Fortunately, the student decided to graph the times for individual RPCs. It turned out the data was bimodal, whereby every other RPC completed quickly, but the intervening ones were all significantly slower. With this information, the student tracked down and fixed a configuration error that eliminated all of the slow times. In this case, the average value was not a good indicator of system behavior.

The examples in this article may seem so esoteric that they must be outliers, but they are not. Every performance measurement project I have seen has had multiple such examples, which are extremely subtle, difficult to track down, and defy all intuition, until suddenly a simple explanation appears (such as unsigned integer underflow). Deeper measurements almost always produce substantial performance improvement, important discoveries about system behavior, or both.

Do not spend a lot of time agonizing over which deeper measurements to make. If the top-level measurements contain contradictions or things that are surprising, start with measurements that could help resolve them. Or pick measurements that will identify performance bottlenecks. If nothing else, choose a few metrics that are most obvious and easiest to collect, even if you are not sure they will be particularly illu-



It can be as fancy as an interactive webpage or as simple as a text file, but a dashboard is essential for any nontrivial measurement effort.



minating. Once you look at the results, you will almost certainly find things that do not make sense; from this point on, track down and resolve everything that does not make perfect sense. Along the way you will discover other surprises; track them down as well. Over time, you will develop intuition about what kinds of deeper measurements are most likely to be fruitful.

Measuring deeper is the single most important ingredient for high-quality performance measurement. Focusing on this one rule will prevent most of the mistakes anyone could potentially make. For example, in order to make deeper measurements you will have to allocate extra time. Measuring deeper will expose bugs and inconsistencies, so you will not accidentally trust bogus data. Most of the suggestions under Rule 2 (Never trust a number generated by a computer) are actually examples of measuring deeper. You will never need to guess the reasons for performance, since you will have actual data. Your measurements will not be superficial. Finally, you are less likely to be derailed by subconscious bias, since the deeper measurements will expose weaknesses, as well as strengths.

Measurement Infrastructure

Making good performance measurements takes time, so it is worth creating infrastructure to help you work more efficiently. The infrastructure will easily pay for itself by the time the measurement project is finished. Furthermore, performance measurements tend to be run repeatedly, making infrastructure even more valuable. In a cloud service provider, for example, measurements must be made continuously in order to maintain contractual service levels. In a research project, the full suite of performance measurements will be run several times (such as before submission, after the paper is accepted, and again during the writing of a Ph.D. dissertation). It is important to have infrastructure that makes it easy to rerun tests.

Automate measurements. It should be possible to type a single command line that invokes the full suite of measurements, including not just top-level measurements but also the deeper measurements. Each run should produce a large amount of performance data in an easy-to-read

form. It should also be easy to invoke a single benchmark by itself or vary the parameters for a benchmark. Also useful is a tool that can compare two sets of output to identify nontrivial changes in performance.

Create a dashboard. A dashboard is a display that shows all performance measurements from a particular run of a particular benchmark or from a deployed system. If you have been measuring deeply, the dashboard can easily show hundreds of measurements. A good dashboard brings together a lot of data in one place and makes it easy to examine performance from many angles. It can be as fancy as an interactive webpage or as simple as a text file, but a dashboard is essential for any nontrivial measurement effort.

Figure 1 shows approximately one-third of a dashboard my students created to analyze the performance of crash recovery in a distributed storage system.³ In this benchmark, one of the system's storage servers has crashed, and several other servers ("recovery masters") reconstruct the lost data by reading copies stored on a collection of backup servers. This sample illustrates several important features of dashboards. Any dashboard should start with a summary section, giving the most important metric(s)—total recovery time in this case—and the parameters that controlled the benchmark. Each of the remaining sections digs deeper into one specific aspect of the performance. For example, "Recovery Master Time" analyzes where the recovery masters spent their time during recovery, showing CPU time for each component as both an absolute time and a percentage of total recovery time; the percentages help identify bottlenecks. It was important for the storage system being analyzed to make efficient use of the network during recovery, so we added a separate section to analyze network throughput for each of the servers, as well as for the cluster as a whole. Most measurements in the dashboard show averages across a group of servers, but in several cases the worst-case server is also shown. The dashboard also has a special section showing the worst-case performance in several categories, making it possible to see whether outliers are affecting overall performance.

You should create a simple dashboard as soon as you start making measurements; initially, it will include just the benchmark parameters and a few overall metrics. Every time you think of a new question to answer or a deeper measurement to take, add more data to the dashboard. Never remove metrics from the dashboard, even if you think you will never need them again. You probably will.

If you make a change and performance suddenly degrades, you can scan the dashboard for metrics that have changed significantly. The dashboard might indicate that, for example, the network is now overloaded or the fraction of time waiting for incoming segments suddenly increased. You can maintain a "good" dashboard for comparing with later dashboards and record dashboards at regular time intervals to track performance over long periods of time. A dashboard serves a purpose for performance measurement similar to that of unit tests for functional testing—providing a detailed analysis and making it easy to detect regressions.

Do not remove the instrumentation.

Leave as much instrumentation as possible in the system at all times, so

performance information is constantly available. For online services that run continuously, the dashboard should take the form of a webpage that can be displayed at any time. For applications that are run manually, it is convenient to have a command-line switch that will cause performance metrics to be recorded during execution and dumped when the application finishes.

One simple-yet-effective technique is to define a collection of counters that accumulate statistics (such as number of invocations of each externally visible request type and number of network bytes transmitted and received). Incrementing a counter is computationally inexpensive enough that a system can include a large number of them without hurting its performance. Make it easy to define new counters and read out all existing counters. For long-running services, it should be possible to sample the counters at regular intervals, and the dashboard should display historical trends for the counters.

Presentation matters. If you want to analyze performance in depth, measurements must be displayed in a way that exposes a lot of detail and allows it to be understood easily. In addition, the presentation must clarify the things

Figure 1. Excerpt from the dashboard used to evaluate crash recovery in a large-scale main memory storage system.³

```

=== Summary ===
Recovery time:                2.58 s
Failure detection time:       0.32 s
Recovery + detection time:    2.90 s
Masters:                      73
Backups:                     146
Total nodes:                  73
Replicas:                     3
Objects per master:           592950
Object size:                   1055.81 bytes
Total recovery segment entries: 43685317
Total live object space:       43584 MB
Total recovery segment space w/ overhead: 43713 MB

=== Recovery Master Time ===
Total (90.5% of recovery time): 2333.64 ms avg / 2533.71 ms max / 100.00% avg
Waiting for incoming segments: 766.78 ms avg / 924.04 ms max / 32.86% avg
Inside recoverSegment:         1283.48 ms avg / 1657.36 ms max / 55.00% avg
Final log sync time:           21.20 ms avg / 50.85 ms max / 0.91% avg
Removing tombstones:           0.00 ms avg / 0.00 ms max / 0.00% avg
Other:                         262.18 ms avg / 673.43 ms max / 10.17% avg

=== Network Utilization ===
Aggregate:                    994.43 Gb/s avg / 54.49%
Master in:                    4.57 Gb/s avg / 333.82 Gb/s total
Master out:                    6.35 Gb/s avg / 463.59 Gb/s total
Master out during replication: 7.61 Gb/s avg / 555.87 Gb/s total
Master out during log sync:   12.06 Gb/s avg / 880.42 Gb/s total
Backup in:                     3.63 Gb/s avg / 530.01 Gb/s total
Backup out:                    3.63 Gb/s avg / 529.98 Gb/s total

=== Slowest Servers ===
Backup opens, writes:          rc26 / 729.2 ms
Stalled reading segs from backups: rc21 / 924.0 ms
Reading from disk:             rc29 / 170.3 MB/s
Writing to disk:               rc23 / 71.3 MB/s

```

that are most important. Think of this as feeding your intuition. The way to discover interesting things is to absorb a lot of information and let your intuition go to work, identifying patterns, contradictions, and things that seem like they might be significant. You can then explore them in more detail.

When students bring their first measurements to me, the measurements are often in a barely comprehensible form (such as unaligned comma-separated values), telling me they did not want to spend time on a nice graph until they knew what data is important. However, the early phase of analysis, where you are trying to figure out what is happening and why, is when it is most important for information to be presented clearly. It is worth getting in the habit of always present-

ing data clearly from the start (such as with graphs rather than tables). Do not waste time with displays that are difficult to understand. Making graphs takes little time once you have learned how to use the tools, and you can reuse old scripts for new graphs. Consider clarity even when printing raw data, because you will occasionally want to look at it. Arrange the data in neat columns with labels, and use appropriate units (such as microseconds), rather than, say, “1.04e-07.”

Figure 2 and Figure 3 show how the organization of a graph can have a big effect on how easy (or difficult) it is to visualize performance data. In Figure 2 my students and I aimed to understand tail latency (99.9th or 99.99th percentile worst-case performance) for write requests in the RAMCloud stor-

age system. A traditional cumulative distribution function (CDF) like the one in Figure 2a emphasizes the mean value but makes it difficult to see tail latency. When we switched to a reverse cumulative distribution function with log-scale axes (see Figure 2b) the complete tail became visible, all the way out to the slowest of 100 million total samples. Figure 2b made it easy to see features worthy of additional study (such as the “shoulders” at approximately 70 μ s and 1 ms); additional measurements showed the shoulder at 1 ms was caused by interference from concurrent garbage collection. If we had only considered a few discreet measurements of tail latency we might not have noticed these features.

Figure 3 arose during development of a new network transport protocol. My students and I wanted to understand the effect of a particular parameter setting on the delivery time for messages of different size in a given workload. The first question we had to address in graphing the data was what metric to display. Displaying the absolute delivery times for messages would not be very useful, since it would not be obvious whether a particular time is good. Furthermore, comparisons between messages of different lengths would not be meaningful, as longer messages inherently take more time to deliver. Instead, we displayed slowdown, the actual delivery time for a message divided by the best possible time for messages of that size. This choice made it easy to see whether a particular time is indeed good; 1.0 is perfect, 2.0 means the message took twice as long as necessary, and so on. Slowdown also made it possible to compare measurements for messages of different length, since slowdown takes into account the inherent cost for each length.

The second question was the choice of the x-axis. An obvious choice would have been a linear x-axis, as in Figure 3a. However, the vast majority of messages is very small, so almost all the messages are bunched together at the left edge of that graph. A log-scale x-axis (see Figure 3b) makes it easier to see the small messages but still does not indicate how many messages were affected by each value of the parameter. To address this problem, we rescaled the x-axis to match the distribution of

Figure 2. Two different ways to display tail latency: (a) a traditional CDF with linear axes; and (b) a complementary CDF (each y-value is the fraction of samples greater than the corresponding x-value) with log-scale axes.

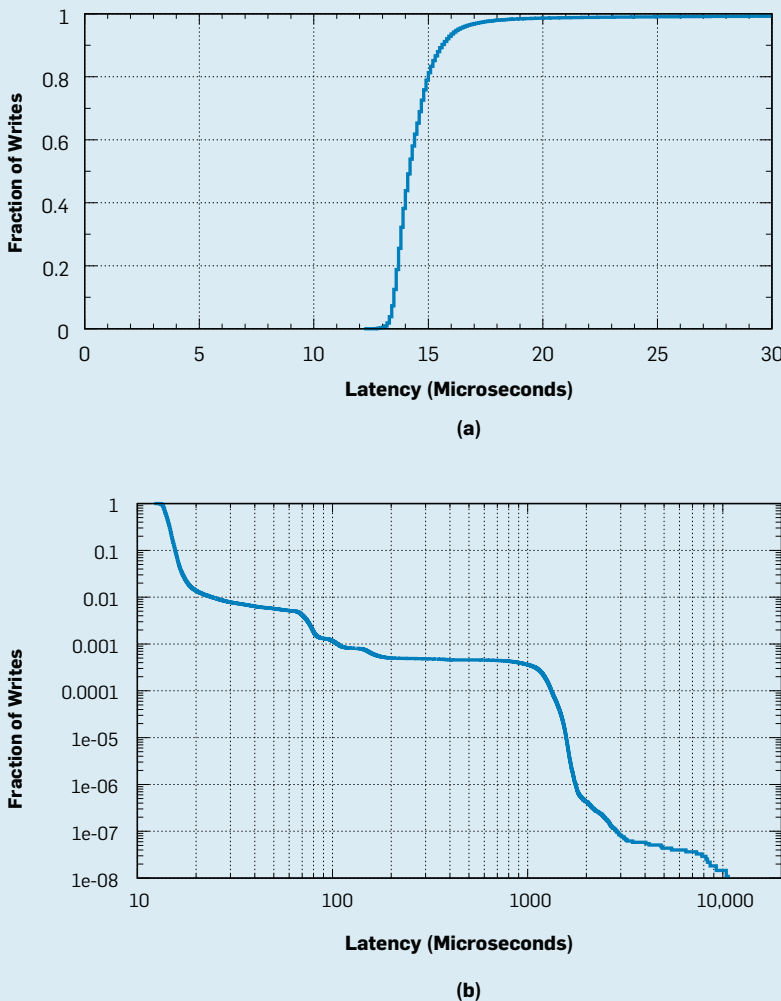
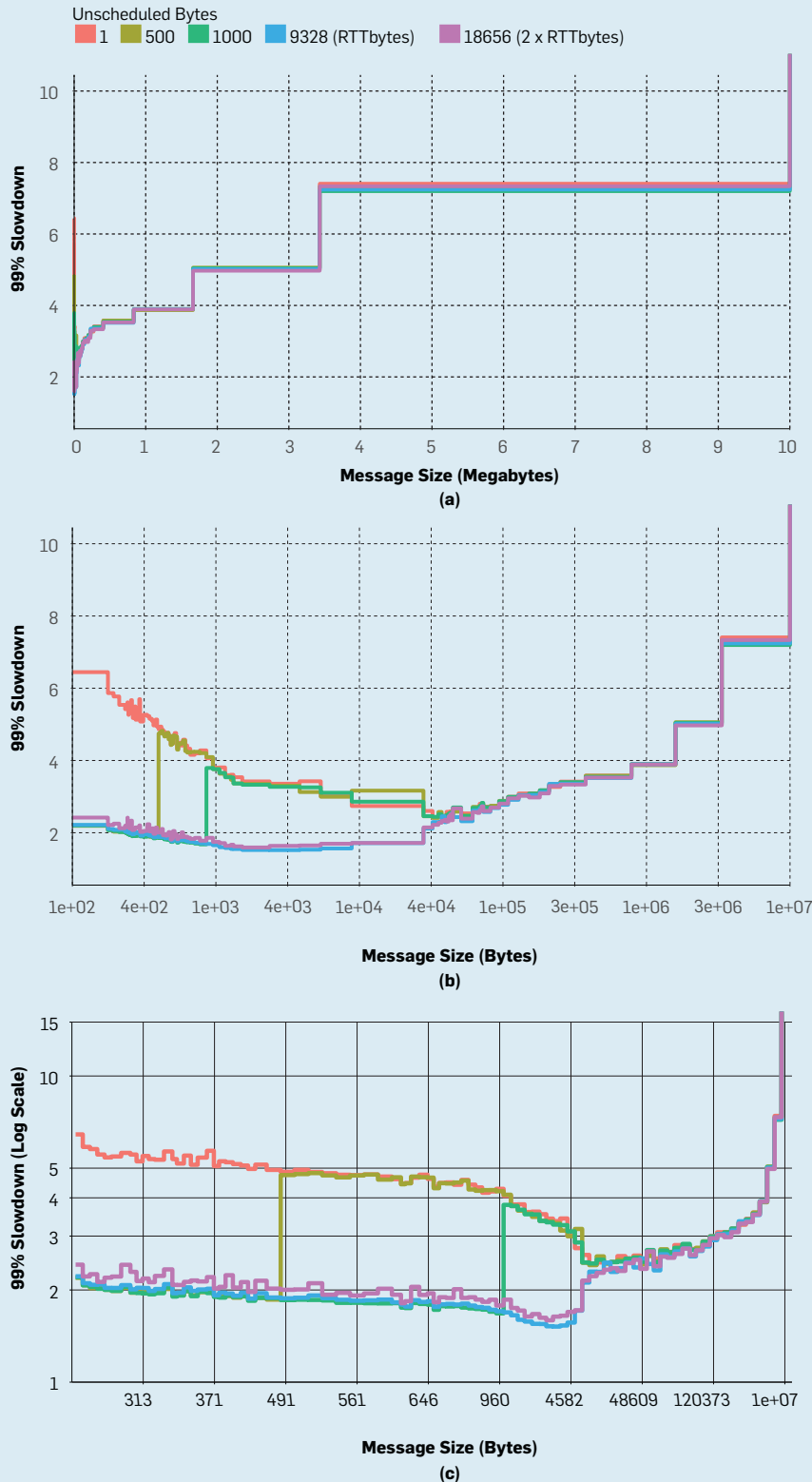


Figure 3. Each figure displays 99th-percentile slowdown (delivery time for messages of a given size, divided by the best possible time for messages of that size) as a function of message size in a given workload: (a) x-axis is linear; (b) x-axis is logarithmic; and (c) x-axis is scaled to match the CDF of message lengths. Different curves correspond to different settings of the “unscheduled bytes” parameter.



message lengths (see Figure 3c); the x-axis is labeled with message size but is linear in number of messages, with each of the 10 tickmarks corresponding to 10% of all messages. With this view of the data it became easy to see that the parameter matters, as it affected approximately 70% of all messages in the experiment (those smaller than approximately 5 Kbytes).

Figure 3c includes more information than the other graphs; in addition to displaying slowdown, it also displays the CDF of message sizes via the x-axis labels. As a result it is easy to see that messages in this workload are mostly short; 60% of all messages require no more than 960 bytes. Figure 3c makes it clear that Figure 3a and Figure 3b are misleading.

Conclusion

The keys to good performance evaluation are a keen eye for things that do not make sense and a willingness to measure from many different angles. This takes more time than the quick and shallow measurements that are common today but provides a deeper and more accurate understanding of the system being measured. In addition, if you apply the scientific method, making and testing hypotheses, you will improve your intuition about systems. This will result in both better designs and better performance measurements in the future.

Acknowledgments

This article benefited from comments and suggestions from Jonathan Ellithorpe, Collin Lee, Yilong Li, Behnam Montazeri, Seojin Park, Henry Qin, Stephen Yang, and the anonymous *Communications* reviewers. 

References

1. Nickerson, R.S. Confirmation bias: A ubiquitous phenomenon in many guises. *Review of General Psychology* 2, 2 (June 1998), 175–220.
2. Ousterhout, J. *A Philosophy of Software Design*. Yaknyam Press, Palo Alto, CA, 2018.
3. Ousterhout, J., Gopalan, A., Gupta, A., Kejriwal, A., Lee, C., Montazeri, B., Ongaro, D., Park, S.J., Qin, H., Rosenblum, M. et al. The RAMCloud storage system. *ACM Transactions on Computer Systems* 33, 3 (Aug. 2015), 7.
4. Rosenblum, M. and Ousterhout, J.K. The design and implementation of a log-structured file system. *ACM Transactions on Computer Systems* 10, 1 (Feb. 1992), 26–52.

John Ousterhout (ouster@cs.stanford.edu) is the Bosack Lerner Professor of Computer Science at Stanford University, Stanford, CA, USA.

Digital effectiveness is not the same as mastering the ICTs, rather it is the art of using them in a purposeful, healthy way.

BY CARLO GABRIEL PORTO BELLINI

The ABCs of Effectiveness in the Digital Society

THE USE OF information and communication technologies (ICTs) by individuals is a long-time concern for researchers and practitioners. ICT use starts with the inclusion of people in the digital society and progresses toward the equalization of their capabilities and opportunities in technology-mediated information and communication processes. Approaches to inclusion and equality have become increasingly sophisticated through developments in human-centered computing and human-computer interaction that replace the old focus on people's mere access to the ICTs. At the same time, a third, more empowering moment of ICT use is attracting scholars, professionals and, hopefully, public agents—the effectiveness with

which people use the technology. In this article, I discuss inclusion, equality and effectiveness under the concept of one's digital effectiveness.

Digital effectiveness manifests in three dimensions—access, cognition, and behavior. It refers to one's use of ICTs for private or professional purposes according to an arbitrarily defined effectiveness criterion. The focal point of digital effectiveness is the individual ICT user and the levels of purposeful ICT use he or she achieves. That is, the focus is on the basic building block of the digital society—the embryo of a society's digital culture and digital health. Digital effectiveness describes an individual's use of ICTs in desirable ways, regardless of whether the individual masters the ICTs or not.

This approach is an extension of the two-order digital divide perspective^{4,5,6} coupled with developments in use effectiveness.^{3,7,8} The rationale is as follows: (1) an individual should have proper access to the ICTs, (2) possess the cognitive potential to use them, and (3) activate the needed behaviors to operate the ICTs in practice (4) for a specific purpose (5) in reference to an effectiveness criterion (6) arbitrarily defined and measured against a stakeholder's utility function (7) that takes as input the individual's digital capabilities and limitations.

» key insights

- The use of ICTs can be described according to three moments of maturation—the inclusion of individuals in the digital society, the equalization of their digital capabilities and opportunities, and their effectiveness in using the ICTs according to personally meaningful purposes.
- ICT-related access, cognition, and behavior are the three critical dimensions of one's capabilities and limitations in using the ICTs in individually and systematically desirable ways.
- Digital effectiveness is the resulting measure of one's digital limitations and capabilities in terms of ICT-related access, cognition, and behavior. It is a relativistic concept, dependent on a stakeholder's ICT use purpose and on the systemic impacts of that purpose.

A < == Access >
B < == Behavior >
C < == Cognition >
D < == Digital >
E < == Effectiveness >

This approach therefore deals with the enablers of ICT access, the cognitive enablers of potential ICT use, and the behavioral enablers to leverage the actual benefits from ICT use. Furthermore, the three critical dimensions of access, cognition and behavior are here conceived in broad terms. Access to the ICTs is not oversimplified as the individual's socio-material setting or mere contact with technology at home, work, school, cybercafés or someone else's venue; rather, access also includes the contextual conditions of ICT use, that is, environmental ergonomics. ICT-related cognition, in turn, includes all technology-mediated information and communication mental activity reflecting formal and informal education, personal experiences, emotions, and the chain of attitudinal factors that precede actual behavior. Finally, ICT-related behavior includes the actual use of technology as a result of one's latent potential, personal deliberation, and real possibilities. In all three dimensions, the diagnostic of effectiveness depends on, first, defining who is the interested party (the main stakeholder of ICT use), and, second, the effectiveness criterion (the ICT use purpose) that serves as reference for action. Thus, it is meaningless to address digital effectiveness in theory or in practice before those two definitions are available.

As noted earlier, an individual's three dimensions of digital effectiveness often manifest in natural sequence—access first, then cognition, then behavior. But they may be also causally linked in

certain circumstances, when a particular digital limitation gives rise to another limitation in the same or in a different dimension. For instance, an individual's poor ICT skills (a cognitive limitation) may lead to technophobia (a cognitive or behavioral limitation, depending on how it is defined and measured), and vice-versa. The possible causal link between limitations is one of the reasons why we measure limitations instead of capabilities to estimate one's digital effectiveness. After the measurement of limitations in a given dimension, the level of capabilities in that same dimension is the difference between the upper limit of the chosen scale (say, 100, as in a percent scale) and the computed level of limitations. Finally, the digital effectiveness value is the difference between the aggregated, normalized digital capabilities and limitations in all three dimensions of access, cognition, and behavior. This is discussed in more detail later.

The digital effectiveness approach contributes in many ways to theory and practice, such as in:

- Broadening, organizing, and unifying the complex discussion on digital divide and inequality, along with introducing technology use effectiveness as a related concern;
- Differentiating between technology mastering and use effectiveness;
- Stimulating public and social agents, organizational managers, community leaders, families, and individuals to better understand the complexity of technology use as critically dependent on proper access to technology,

the development of cognitive potential to deal with technology-mediated information and communication processes, and the development of healthy behaviors toward technology;

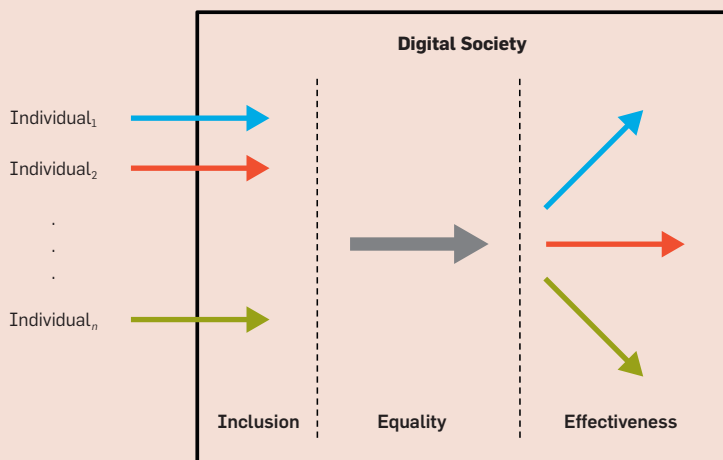
- Identifying innate personal traits and attitudinal mechanisms that impact one's technology use effectiveness;
- Stimulating technology-related personal awareness (by doing self-evaluations about the use of ICTs), family counseling (by mentoring family members about the ICTs), organizational improvement (by rationalizing the use of ICTs at work), community empowerment (by promoting digital literacy and citizenship), efficiency and transparency in public administration (by streamlining electronic government processes), and social change (by means of user-generated content and regulation in virtual social networks).

Effectiveness in the Digital Age

The development of digital capabilities and the mitigation of digital limitations is a concern for individuals and groups. The digital society demands personal awareness, family counseling, community leadership, organizational vision and public policies that promote the proper access to, and the purposeful use of, the ICTs. Digital effectiveness is how I broadly refer to this desirable state of positive outcomes to be realized in the interaction of humans, information, and technology.

Bellini et al.¹ outlined a three-dimension digital limitations model expected to inform the digital divide/inequality literature in the fields of communications, information science, sociology, and public policy making. I now extend their propositions and introduce ICT use effectiveness as a related concern. Together, the digital limitations model and the concept of ICT use effectiveness form the digital effectiveness approach to the dimensions of ICT-related access, cognition, and behavior. In particular, I posit that each digital effectiveness dimension can be measured within a *diglim* range of values for a given ICT user in reference to an ideal parameter—an arbitrarily defined, context-specific ICT use expectation, which is set by the stakeholder who is the focal beneficiary of ICT use. As a result, anyone who expects particular outcomes from the use of ICTs (for example, an employee who

Figure 1. From inclusion to effectiveness in the digital society.



performs an ICT-mediated task, or an employer who provides a new ICT infrastructure to an organizational unit) may design evaluation instruments and collect empirical data to address ICT use outcomes from a *diglim*-level digitally limited individual or group of individuals. This relativistic rather than optimizing view of technology use builds on developments on the interplay between cognition, affect and intentions toward the Internet;⁶ Internet access types known as motivational, material, skills, and usage access;¹³ Internet access, skills, and use;⁹ the conceptualization of use effectiveness;^{3,7,8} the causally linked digital divides known as access, capability, and outcome divides;¹⁴ and the idea that ICT diffusion with social embeddedness follows the stages of access to technology, use of technology, and the impact of technology use.¹⁰

Limitations, Capabilities, and Effectiveness

Figure 1 shows three distinctive moments of the relationship between humans and ICTs. The moments nearly equate to how the literature evolved toward its focal interests in the last two decades. The human-ICT relationship starts with the inclusion of individuals in the digital society, progresses through equalizing their digital capabilities and opportunities, and eventually the individuals will develop their own relativistic perception of effectiveness and preferred route in the digital society. The relativistic perception is due to individuals having different ICT use purposes and possessing different personal traits and attitudinal mechanisms. As such, ICT-capable individuals reach at different, personally meaningful states of ICT use.

Figure 2 illustrates the three dimensions of one's digital effectiveness. The three dimensions are plotted in a strictly positive Cartesian space that represents an individual's digital limitations. The strictly positive assumption means that magnitudes are measured on a ratio scale (so, negative values are meaningless) and that it is axiomatically impossible to eliminate all residues of digital limitations (so, a zero is also meaningless).

Digital effectiveness is then defined as the difference between digital capabilities and limitations. The proposi-

Digital effectiveness manifests in three dimensions—access, cognition, and behavior.

tion of measuring limitations instead of capabilities in order to draw the coordinate (a,b,c) in Figure 2 and infer about effectiveness is based on the idea that it is more efficient to identify what does *not* match a pattern than the opposite. This is true in a vast number of situations. For instance, in the philosophy of science, all statements are assumed to be falsifiable,¹¹ that is, there is an exception to every rule—so, we should look for the exceptions. Another example is that the process of building and using computer information systems is “never ending and error prone,”¹² thus an important principle in software testing is to search for failure. Accordingly, if we focus on identifying one's digital limitations instead of capabilities, we simply need to look for any deviance from the pattern (the expected digital capability), in a management-by-exception fashion. Also, a digital limitation may be the direct cause of another limitation, so priority should be given to identifying limitations instead of capabilities.

The first dimension where we measure the occurrence of digital limitations refers to one's social, material and contextual barriers to properly access and use the ICTs in information and communication processes. It reflects *access limitations* (A_{lim}). Access limitations manifest through the levels of social exclusion, the lack of Internet access and desirable bandwidth, obsolete hardware and software, poorly designed human-computer interfaces and office furniture, rooms that are not noise- or smoke-free, rooms that are not clean or controlled for temperature and privacy, insufficient time to perform the tasks in the computer, and other factors.

The second dimension refers to barriers in one's neurological structure, educational background, information-processing capabilities, and hands-on experience that undermine the potential use of ICTs. It reflects *cognitive limitations* (C_{lim}). Cognitive limitations manifest through how one tries to search, select, process and apply ICT-mediated information, with origins in neuropsychological traits and mental disorders, incomplete formal education, lack of digital literacy and computer experience, poor general experience (that causes low functional variety), lack of interest in information processing and problem solving, unrealistic

beliefs, wishful thinking, anxiety, pessimism, sadness, low self-confidence, overconfidence, and other factors.

The third dimension refers to barriers in one's complex intertwining of beliefs, attitudes and intentions that eventually result in negative behaviors toward the ICTs. It reflects *behavioral limitations* (B_{lim}). Behavioral limitations manifest through at least three archetypical behaviors: psychological barriers toward the ICTs that imply a pathological type of technology non-use (technophobia); the unnecessary, excessive use of technology (technoadiction); and the use of technology in undesirable ways for the self or others (predatory technophilia). An individual's behavior is shaped by his or her neurological constitution, personal discretion in decision-making, traumas and addictions developed spontaneously or by influence of external sources. An individual can effect positive behaviors toward the ICTs (such as using them productively in electronic commerce, electronic government, online banking, distance learning, virtual social networks, and so on) or negative behaviors (such as using the ICTs for leisure during work or for work during leisure, propagating false information and computer viruses, promoting unethical behavior, using any source of information in excess so that it becomes a personal bias, giving less attention than needed to available information relative to a particular issue, and so on). Behavioral limitations reveal that ICT use effectiveness is not only dependent on technology access and cognitive potential, but also on the actual interaction of humans and computers in order to meet voluntarily espoused or externally defined utilitarian purposes.^a

^a I am interested exclusively in modeling utilitarian ICT use. Any other use purpose would be meaningless to develop the 3D digital space, that is, it is not possible to plot a point in that space if there is no common way to measure ICT use effectiveness based on one's digital limitations—what is clearly dependent on the definition of use purposes. However, if, say, hedonic or emotion-based use is defined according to a utility function, it may be measured by this approach as well. On the other hand, the definition of the utility of ICT use (that is, the intended purpose of use) may be influenced by mental dysfunctions. This poses enormous complexity to any model. So, I do not question whether a stakeholder's purpose is rational or not.

However, a given behavior is technically considered a personal limitation only when that behavior does not promote the arbitrarily defined purpose of ICT use. Therefore, behaviors that could be seen as limiting one's digital effectiveness, such as non-work-related computing (also known as cyberslacking or cyberloafing) and routine-preserving behavior (also known as resistance to change), will be considered actual behavioral limitations and a threat to digital effectiveness only if they prevent one to meet the criterion of ICT use effectiveness that is defined by those who have an interest in, or who are affected by, the particular behaviors.

Table 1 synthesizes the important concepts of digital effectiveness and provides a few examples of phenomena that may be addressed with the digital effectiveness approach.

It is important to note that cognitive, affective and hedonic events are here conceived as a single dimension. In fact, it is very difficult—if not impossible—to separate rational thinking from pure sentiments. Cognition and the other two dimensions are modeled here to be critical and self-explanatory: the access dimension addresses phenomena that are mostly external to the individual; the cognitive dimension addresses phenomena that occur exclusively in the individual's mind (cognition, affection, hedonism, and the stock of data, information, knowledge, and wisdom¹²); and the behavioral dimension addresses phenomena that the individual instantiates through real actions. If the digital effectiveness approach distinguished between types of mental processes, there would be overlapping areas of mental dimensions and much confusion on how to plot the Cartesian coordinate for a given mental phenomenon. The cognitive dimension thus includes all ICT-related personal evaluations (apart from the effective actions they precede) that are rational or irrational, deliberate or instinctive, conscious or unconscious, planned or emergent, evidence-based or emotional.

The three dimensions can be thus plotted like in Figure 2 to illustrate one's digital limitations after the components of each dimension are identified, measured, and normalized. Normalization is needed because the components will

likely refer to very different phenomena. Also, as the effectiveness criterion is arbitrarily defined by a stakeholder in a specific situation of ICT use, the instruments to measure the components need to be customized.

The notation $A_{lim} + B_{lim} + C_{lim} = D_{lim}$ synthesizes the assumption that one's Digital limitations are the result of computing his or her Access limitations, Behavioral limitations, and Cognitive limitations in order to also estimate the complementary symptom—one's digital capabilities (Figure 3). A digital limitation is the degree to which an individual is limited in his or her capabilities to use the ICTs in reference to an arbitrarily defined, context-specific ICT use objective. And digital effectiveness is the difference between capabilities and limitations. The resulting value represents an excess of capabilities (a positive value), an excess of limitations (a negative value), the equivalence of capabilities and limitations (a zero value), or the individual's digital evolution (a variation across measurements).

Hypothetically, if we adopt a percent scale to measure each digital limitation dimension, and if $diglim_{\delta}$, $\delta \in \{A, B, C\}$, is the degree of one's particular digital limitation, then his or her digital effectiveness is $((100 - diglim_A) - diglim_A + (100 - diglim_B) - diglim_B + (100 - diglim_C) - diglim_C) / 3$, or $(\sum_{\delta \in \{A, B, C\}} 100 - 2 * diglim_{\delta}) / 3$. As an illustration, if someone has an access limitation of 30% (access capability of 70%), a cognitive limitation of 50% (cognitive capability of 50%) and a behavioral limitation of 10% (behavioral capability of 90%), his or her digital effectiveness is 40%, that is, $((70 - 30) + (50 - 50) + (90 - 10)) / 3$. This value will be useful if three conditions hold:

- The three dimensions are critical, that is, no dimension should be weighted for being more important than another;
- The components of all three dimensions are normalized; and
- The components selected to describe each dimension follow the 80–20 rule (Pareto principle), that is, most of the total variance of the phenomenon results from vital few components.

Measuring digital effectiveness as the difference between capabilities and limitations gives rise to interpretations. If the computed average of limitations in the three dimensions is 30% and the computed average of ca-

pabilities is 70%, then the individual's digital effectiveness is 40%—but what does this mean in practice?

So far, the best interpretation possible is given by theory-based versus heuristics-based knowledge. In most practical decisions of everyday life, we decide in favor of a given alternative on the basis of estimating its benefits and costs, irrespectively of whether we know of any supporting theory. If one says “weighing up the pros and cons, I prefer to stay in my current job,” he or she is selecting an alternative solution based on an estimated difference between costs and benefits, and that difference is deemed positive and significant by the incumbent of the decision. Those two computations (the difference and the significance) are mostly based on personal experience and purpose, not on theory—that is, the decision maker arbitrarily assigns an expected utility one is able to compute based on the evidence he or she is able to collect and organize when thinking about the space of alternative solutions.

Personal determination, family and community counseling, organizational mentoring, and public policies should seek to minimize an individual's or a group's $diglim_\delta$ in order to maximize digital capabilities and, ultimately, digital effectiveness. High levels of digital effectiveness lead society to a state of digital culture, digital literacy, and, hopefully, digital health—that is, to the positive outcomes that might stem from the proper use of ICTs. Therefore, digital effectiveness is a state of desirable digital access, behavior, and cognition, or simply $A + B + C = D$. In other words, digital effectiveness is not a matter of mastering the ICTs, but of using them effectively in regard to use objectives that promote desired outcomes for individuals, families, communities, organizations, and the larger environment. This is critical to understand that digital effectiveness promotes positive social development. And the opposite is also true. For instance, if someone masters the technology (cognitive limitations can be assumed to be low) but uses it to harm people or other systems (behavioral limitations can be assumed to be high), we (as stakeholders) may conclude that his or her digital effec-

Table 1. Key concepts and examples of limitations.

Dimension	Examples of cases that are typically considered limitations*
Access The mediating conditions an individual has to deal with in order to have contact with the ICTs and the information they convey.	Social exclusion, lack of Internet access and desirable bandwidth, obsolete hardware and software, poorly designed human-computer interfaces and office furniture,** rooms that are not noise- or smoke-free, rooms that are not clean or controlled for temperature and privacy, insufficient time to perform tasks in the computer, and unstable supply of technology by local retailers.
Cognition Information- and technology-oriented psychological processes of an individual, including both rational and affect-based processes.	Neuropsychological traits and mental disorders, incomplete formal education, lack of digital literacy and hands-on computer experience,** poor general experience, lack of interest in information processing and problem-solving, unrealistic beliefs, wishful thinking, computer anxiety,** low computer self-efficacy, and overconfidence.**
Behavior The ways an individual uses the ICTs, including deliberate nonuse.	Technophobia, technoadiction, propagating false information and viruses, promoting unethical behavior, excessive use of limited sources of information, impression management,** giving less attention than needed to available information, non-work-related computing,** ICT use for work during leisure, and routine-preserving behavior.**

Use expectation, use preference, subjective utility

The criterion that guides the identification of digital limitations. It is based on a stakeholder's judgment about how the ICTs should be used in a specific situation. The stakeholder may be the very incumbent of ICT use, his or her superior at work, family members, state officials, etc. When there are multiple stakeholders involved in a use situation, they will have different priorities and hidden agendas, and as such, power, negotiation and the criticality of ICT use will be the driving forces for defining the group's digital effectiveness criterion.

* Being considered a limitation or not is dependent on the stakeholder's effectiveness criterion.

** Cases that my team and I already studied empirically, such as how the levels of computer self-efficacy and anxiety reflect capabilities and limitations of digital natives in unexpected ways.²

Figure 2. An individual's 3D digital limitations that compromise digital effectiveness.

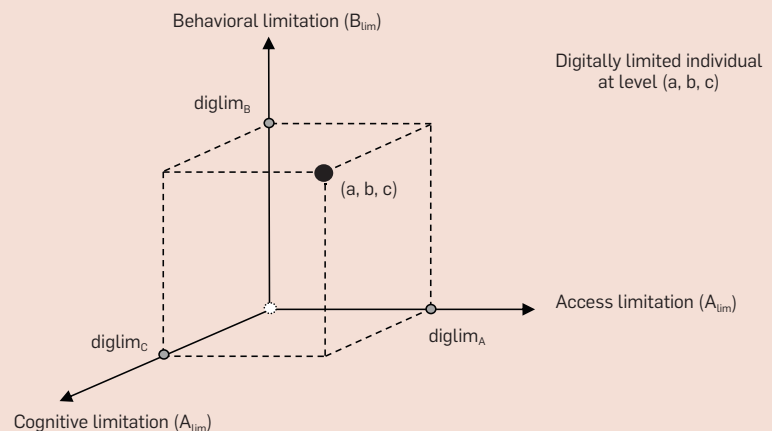
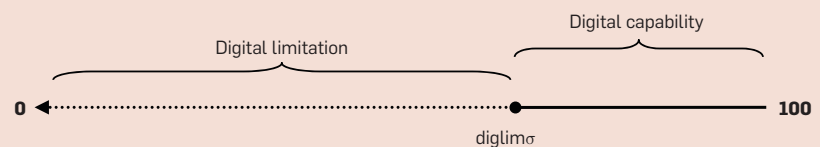


Figure 3. An illustrative dimension of digital effectiveness.



Note: The arrow to the left indicates that the minimization of limitations is the goal.

tiveness is low.^b On the other hand, a very limited person in regard to ICT-related cognition who is able to deploy technology as demanded in professional routines may deliver high levels of digital effectiveness.

A Case

I will exemplify the main aspects of the approach with a case on electronically filing tax returns, and doing it while at

b It is also possible that, if ICT use objectives are defined so as to harness people or other systems, and if one's actions are effective that end, we could argue that the ICT user's digital effectiveness criterion was met. Although the present approach may include such cases, I take for granted that we need a responsible digital society, so that ICT use objectives and outcomes should not be detrimental to people and the environment.

work. Procrastinating tax submissions is a common phenomenon. In 2017, 50% of U.S. taxpayers did not file their tax returns until the last two weeks before Tax Day, while 40% did not until the last week; and, comparing the submissions in equivalent weeks of 2017 and 2016, the numbers were worse in 2017 in all comparisons before and after the due date.^c Contrasting the U.S. numbers with those in a country with a different economic and social reality but comparable size and population such as Brazil, the situation is similar.^d Among the U.S. taxpayers who filed their forms

c <https://www.irs.gov/newsroom/2017-and-prior-year-filing-season-statistics>
https://goo.gl/bRcKm
d <https://goo.gl/ZwvPtB>

until one week before Tax Day, 92% did it electronically,^e while in Brazil all submissions are in electronic form since 2011. Therefore, filing tax returns is an ICT-dependent process marked by significant procrastination in practice. Although electronic tax submissions have several benefits over other forms of submission, users see the process as complex, tedious, and time consuming. However, procrastinating tax submissions or doing it incorrectly may impose important penalties on individuals, while also impeding governments to see the big picture of taxes earlier.

As many people spend hours at work each day, it is reasonable to expect non-work-related computing (NWRC) in companies when the deadline for tax submission is approaching. However, many companies have policies restricting the use of organizational resources, including the ICTs, for personal matters, whereas people use different ethical standards to guide personal behavior, sometimes indulging themselves in doing NWRC. A digital effectiveness situation thus arises, which involves at least three stakeholders—the employee (taxpayer), the employer, and the government. Table 2 synthesizes the analytical aspects for the specific case in which the employee is the focal stakeholder, that is, the one whose ICT use purpose should be met.

The approach promotes reflection about the ICT use situation, as there is always a need to accommodate the personal interests (the ICT use purpose) and the opportunities for action (the personally and environmentally defined limitations and capabilities). As in Table 2, enforcing or wisely relaxing the norms about ICT use in organizations is a matter of managing priorities and reducing possibly unnecessary tension, that is, recognizing the presence of different rationalities, personal needs and paths to group effectiveness. In the particular case of NWRC, it should not be always seen as detrimental to work, either because people will inevitably search for the satisfaction of pressing needs, or because judicious NWRC can be compared to a coffee break—a needed pause at work.

e <http://fortune.com/2017/04/14/tax-day-2017-april-18-not-april-15/>

Table 2. A case about preparing and submitting personal tax forms using the organizational ICTs at work.

Stakeholder	Digital effectiveness criterion
Employee	The employee wants to file his or her tax return using the available ICTs at work.
Employer	The employee should use the company's ICTs exclusively for work due to productivity and security issues.
Government	The taxpayer (employee) should file his or her tax return electronically by the due date regardless of other issues.
Context The three stakeholders differ in their digital effectiveness criteria about the employee's due actions. Here I will focus on the employee's perspective. Let me call him John. John is a new 21-year-old customer-service attendant in a department store. He works at the store during eight hours per weekday. John depends on the store's ICTs to prepare and submit the tax forms, as his smartphone has an obsolete operating system that does not run the needed apps or websites, whereas, at home, John devotes attention to family. He also does incidental gardening services when time permits.	
Access limitations John planned to use the store's computers and Internet access to prepare and submit the tax forms during break times and fractions of working time. There was no private room to do it, so John was afraid of doing NWRC and people seeing his tax numbers. Moreover, the shared computer rooms were noisy, the air conditioning system was being repaired, and people frequently interrupted John to chat. He also needed permission from the technical support to have access to external websites and install temporary tax apps. Finally, the tax tools required registration and authentication by the user, thus imposing more delays on John's access to the actually needed ICT functionality.	
Cognitive limitations It was the first time John used computer tools for tax submission. Although he pertains to the digital natives generation, he had a learning curve to overcome. Also, John had significant changes in his tax profile in the preceding year, as the family increased and he moved to this new job and company. To make things more problematic, John did not know much about tax preparation, and, in his country, alternative tools were available for electronic tax filing, with each tool having a different interface and options for free (basic) and paid (full) services.	
Behavioral limitations John delayed until the very last week to prepare the tax forms. He also procrastinated to organize the documents reporting his expenditures and earnings in the preceding year. And when John had the chance to use the store's computers to prepare the tax forms, he often got distracted by checking the virtual social networks and reading online reviews about his consumption passion—musical instruments.	
Digital effectiveness John had significant digital limitations in all three dimensions, but eventually he was able to file the forms by the due date. Particularly helpful were his capabilities to adapt technology and the work environment to the situational needs, learn fast due to information-processing capabilities grounded in formal education and innate analytical talents, and find support from others who alleviated the policies on NWRC and temporarily exchanged task responsibilities. However, the process was stressful and the quality of assigned tasks at work was compromised. From a purely utilitarian perspective, though, John achieved digital effectiveness to some degree.	

An Agenda for Research and Action

I described the key aspects of an approach to integrate digital inclusion, digital equality, and ICT use effectiveness. Someone is digitally included, equal, and effective if he or she has the desired/needed^f access to the ICTs, the desired/needed cognitive potential to use the ICTs, and the desired/needed behavior to leverage the benefits that may stem from ICT use.

Academic studies, individual and group practice, and public policy making benefit from this approach in organizing the key factors in research models, personal actions, organizational planning and social programs that target inclusion, equality, and effectiveness in the digital society. However, the digital effectiveness approach is not an operating model to be put into action directly, as it is highly abstract. Rather, it is a guide for research, practice, and policy making in positioning, measuring, and interpreting phenomena in one, two, or three interrelated dimensions and according to several assumptions. The approach is expected to have the qualities of being systemic and systematic, but it must be parameterized in each application. In particular, the components of each digital dimension and the stakeholder's effectiveness criterion should be modeled beforehand, for the measure of effectiveness to be as accurate and useful as possible. Here, I present ideas for studies and applications:

► Current studies on particular digital effectiveness topics can be positioned in the most appropriate dimension of this approach. Also, it is interesting to answer what is the main focus of already published studies in light of the approach—are current studies more interested in the components of digital limitations in specific situations, the independent variables that act on them, the intervening variables that moderate the effects, or the outcome variables of digital effectiveness? New studies might also start with the digital effectiveness approach to organize the analytical frame of reference before going to the empirical field.

► In the absence of more specific scales to measure a given limitation,

each of its components might be measured according to a normalized five-point scale ranging from “fully absent” to “fully present” or according to a percent scale. Contrarily, if case-specific scales are available, Likert and percent scales might be also useful to normalize the measurements for 3D plotting.

► I proposed an interpretation for the digital effectiveness value—the arithmetic difference between capabilities and limitations. However, is it possible to define an ideal proportion or difference between limitations and capabilities in each effectiveness dimension, so that we could look directly at the proportion or the difference and straightforwardly decide if an individual is digitally effective or not? That is, is there a reference value based on statistical records of decisions made in practice and their reported outcomes on a given digital effectiveness situation?

► Another question is if the effectiveness criteria defined by different stakeholders significantly differ in most situations. For instance, what is the typical difference of ICT-related purposes between employers and employees, teachers and students, family members, or the users of a shared tool for online communications?

► Also, what is the typical positioning of group members and their clustering in the 3D digital effectiveness space, such as a work unit, a family, a class of students, or an electoral district? That is, is there a typical distance between group members in certain situations?

► Another effort would be to identify the typical individual that is most limited in each digital effectiveness dimension, given that this may reflect demographic phenomena that we are not aware of.

► The digital effectiveness approach originally describes the capabilities and limitations of a single individual. Nevertheless, it can be extended to describe how the effectiveness level of one individual impacts the effectiveness level of another individual. For instance, an individual's behavioral limitation regarding the misuse of netiquette in a virtual social network, such as when generating irrelevant information—noise—in excess, may limit the access of others to relevant information. Therefore, research on digital effectiveness might also address the interplay between the

level of effectiveness of multiple individuals in a group.

The goal of the digital effectiveness approach is to promote awareness about the requisites of a purposeful, healthy digital society, starting at the individual level. Such a society has specific demands for human-centered design of technology and for how ICT innovation, diffusion, and use are conceived. In particular, I advocate that organizations, scholars and public agents should devise actions to include and equalize people in the digital society, while also assuring the positive conditions for the individuals, by themselves, to define and pursue effectiveness in the use of ICTs according to their interests and the interests of the larger environment. C

References

1. Bellini, C.G.P., Giebelen, E. and Casali, R.R.B. Limitações digitais. *Informação & Sociedade* 20, 2 (2010), 25–35.
2. Bellini, C.G.P., Isoni Filho, M.M., De Moura, Jr., P.J. and Pereira, R.C.F. Self-efficacy and anxiety of digital natives in face of compulsory computer-mediated tasks: A study about digital capabilities and limitations. *Computers in Human Behavior* 59, 1 (2016), 49–57.
3. Burton-Jones, A. and Grange, C. From use to effective use: A representation theory perspective. *Information Systems Research* 24, 3 (2013), 632–658.
4. Dewan, S. and Riggins, F.J. The digital divide: Current and future research directions. *J. AIS* 6, 12 (2005), 298–337.
5. DiMaggio, P. and Hargittai, E. *From the 'digital divide' to 'digital inequality': Studying Internet use as penetration increases*. (2001), Working paper #15, Princeton, NJ, Center for Arts and Cultural Policy Studies, Princeton University. <http://www.princeton.edu/~artspol/workpap15.html>. Retrieved July 27, 2015.
6. Donat, E., Brandtweiner, R. and Kerschbaum, J. Attitudes and the digital divide: Attitude measurement as instrument to predict Internet usage. *Informing Science* 12 (2009), 37–56.
7. Gurstain, M. Effective use: A community informatics strategy beyond the digital divide. *First Monday* 8, 12 (2003).
8. Gurstain, M. Open data: Empowering the empowered or effective data use for everyone? *First Monday* 16, 2 (2012).
9. Helsper, E.J. and Reisdorf, B.C. A quantitative examination of explanations for reasons for Internet nonuse. *Cyberpsychology, Behavior & Social Networking* 16, 2 (2013), 94–99.
10. Hilbert, M. Technological information inequality as an incessantly moving target: The redistribution of information and communication capacities between 1986 and 2010. *J. Assoc. Information Science & Technology* 65, 4 (2014), 821–835.
11. Popper, K.R. *The Logic of Scientific Discovery*. (2002), Routledge, London, U.K.
12. Rowley, J. The wisdom hierarchy: Representations of the DIKW hierarchy. *J. Information Science* 33, 2 (2007), 163–180.
13. Van Deursen, A.J.A.M. and Van Dijk, J.A.G.M. Toward a multifaceted model of Internet access for understanding digital divides: An empirical investigation. *The Information Society* 31, 5 (2015), 379–391.
14. Wei, K.-K., Teo, H.-H., Chan, H.C. and Tan, B.C.Y. Conceptualizing and testing a social cognitive model of the digital divide. *Information Systems Research* 22, 1 (2011), 170–187.

Carlo Gabriel Porto Bellini (carlo.bellini@pq.cnpq.br) is an associate professor of information systems in the Department of Management at the Federal University of Paraíba, Brazil.

© 2018 ACM 0001-0782/18/7 \$15.00.

^f It is *desired* if defined by the person himself/herself, and it is *needed* if defined by someone else.



s2018.siggraph.org

The 45th International Conference & Exhibition on
Computer Graphics & Interactive Techniques



Sponsored by ACM **SIGGRAPH**



GENERATIONS / **VANCOUVER** 12-16 AUGUST **SIGGRAPH2018**

REGISTER BY 22 JUNE TO SAVE

research highlights

P. 94

Technical Perspective The Rewards of Selfish Mining

By Sharon Goldberg
and Ethan Heilman

P. 95

Majority Is Not Enough: Bitcoin Mining Is Vulnerable

By Ittay Eyal and Emin Gün Sirer

Technical Perspective

The Rewards of Selfish Mining

By Sharon Goldberg and Ethan Heilman

AMIDST ALL THE hype surrounding blockchain and cryptocurrency, it is worth stepping back to understand how cryptocurrency technology broke through to the mainstream. While the community has been developing digital currency systems since the 1980s, Bitcoin was the first to see widespread use. Bitcoin has two crucial properties that set it apart from prior proposals. First, Bitcoin is decentralized; instead of passing transactions through a central bank, Bitcoin uses a decentralized network of *miners* and a distributed consensus algorithm to agree on a single public ledger of transactions called the *blockchain*. The blockchain is just a list of blocks, each of which contains a set of confirmed transactions. Second, Bitcoin has baked-in incentives: miners compete to add a block to the blockchain by solving a computational puzzle, and the winning miner claims several bitcoins as a *mining reward*. This incentive structure has given rise to a thriving industry of bitcoin miners.

When the Bitcoin protocol was first announced, the community assumed that each time a miner successfully solved a computational puzzle, she would immediately announce her solution to the network. After all, if our miner delayed announcing her solution to the network, some other miner might find his own solution and preemptively claim the mining reward. As such, the 2009 Bitcoin white paper makes an implicit assumption of perfect information—that all miners have the same view of the blockchain, and each time a miner solves a puzzle, the puzzle solution and corresponding block is immediately known to all other miners.

The following paper by Eyal and Sirer was the first to question this assumption.

The authors made two crucial observations. First, they noticed that while Bitcoin miners were assumed to act independently, in practice, miners organized themselves in pools of

collective cooperation. These *mining pools* collectively work toward solving each puzzle while sharing the resulting mining rewards. Second, they observed that strategic information propagation could be exploited to increase mining rewards. Specifically, the authors find that mining pools can increase their cumulative mining rewards by *selfish mining*, or strategically sharing puzzle solutions within their own mining pool, while delaying the announcement of those solutions to the Bitcoin network at large.

This surprising result, which was also one of the earliest academic analyses of the Bitcoin protocol, flew in the face of the conventional wisdom of the time. The authors showed that it is not incentive-compatible for miners to “honestly” follow the Bitcoin consensus algorithm, an observation that has serious implications on the security of Bitcoin’s consensus algorithm.

This paper was controversial when it was first made public. Sirer announced this paper on November 3, 2013 by tweeting “You heard it here first: now is a good time to sell your Bitcoins” because “[Bitcoin is] fundamentally broken at the protocol layer.” The authors took an unusual step of publicly disclosing their selfish-mining attack without first informing the Bitcoin developers. More typically, with responsible disclosure, a

security vulnerability is disclosed to the public only after a period of time that allows the vulnerability to be patched.

The authors argue that there were several barriers to this type of responsible disclosure. First, because Bitcoin developers are a distributed organization that eschews traditional forms of governance, there was no clear point of contact for securely disclosing protocol vulnerabilities. Second, because some of the Bitcoin developers could be involved in Bitcoin mining, the authors worried that the developers could leak the attack to certain Bitcoin miners, who might then begin to exploit it. Third, there was no clear way to resolve this issue, since making changes to Bitcoin’s consensus algorithm requires the consensus of all the Bitcoin miners, and arriving at this consensus is no easy task.^a The confusion following the announcement of this paper highlighted a key issue that remains open today: What is the right process for responsibly disclosing security vulnerabilities to decentralized blockchain projects?

Beyond this, Eyal and Sirer’s work triggered a burgeoning area of academic research on blockchain consensus algorithms and security. We are currently in the midst of an unusually fast-moving period where academic research results are quickly transitioned into production blockchain projects. What better way to learn about this exciting space than by reading one of the papers that started it all. 

Eyal and Sirer’s work triggered a burgeoning area of academic research on blockchain consensus algorithms and security.

^a Indeed, just last August, disagreements about changes to Bitcoin’s consensus algorithm caused Bitcoin to split into two blockchains—Bitcoin (BTC) and Bitcoin Cash (BCH).

Sharon Goldberg is an associate professor in the Department of Computer Science at Boston University, Boston, MA, USA, and co-founder/CEO of Commonwealth Crypto, Inc.

Ethan Heilman is a Ph.D. student in computer science at Boston University, Boston, MA, USA, and co-founder/CTO of Commonwealth Crypto, Inc.

Majority Is Not Enough: Bitcoin Mining Is Vulnerable

By Ittay Eyal and Emin Gün Sirer

Abstract

The Bitcoin cryptocurrency records its transactions in a public log called the blockchain. Its security rests critically on the distributed protocol that maintains the blockchain, run by participants called miners. Conventional wisdom asserts that the mining protocol is incentive-compatible and secure against colluding minority groups, that is, it incentivizes miners to follow the protocol as prescribed.

We show that the Bitcoin mining protocol is not incentive-compatible. We present an attack with which colluding miners' revenue is larger than their fair share. The attack can have significant consequences for Bitcoin: Rational miners will prefer to join the attackers, and the colluding group will increase in size until it becomes a majority. At this point, the Bitcoin system ceases to be a decentralized currency.

Unless certain assumptions are made, selfish mining may be feasible for any coalition size of colluding miners. We propose a practical modification to the Bitcoin protocol that protects Bitcoin in the general case. It prohibits selfish mining by a coalition that command less than 1/4 of the resources. This threshold is lower than the wrongly assumed 1/2 bound, but better than the current reality where a coalition of any size can compromise the system.

1. INTRODUCTION

Bitcoin¹⁵ is a cryptocurrency that has recently emerged as a popular medium of exchange, with a rich and extensive ecosystem. The Bitcoin network runs at over 42×10^{18} FLOPS,⁴ with a total market capitalization around 12bn US Dollars as of January 2014.⁵ Central to Bitcoin's operation is a global, public log, called the *blockchain*, that records all transactions between Bitcoin clients. The security of the blockchain is established by a chain of cryptographic puzzles, solved by a loosely-organized network of participants called *miners*. Each miner that successfully solves a cryptopuzzle is allowed to record a set of transactions, and to collect a reward in Bitcoins. The more *mining power* (resources) a miner applies, the better are its chances to solve the puzzle first. This reward structure provides an incentive for miners to contribute their resources to the system, and is essential to the currency's decentralized nature.

The Bitcoin protocol requires a majority of the miners to be *honest*; that is, follow the Bitcoin protocol as prescribed. By construction, if a set of colluding miners comes to command a majority of the mining power in the network, the currency stops being decentralized and becomes controlled by the colluding group. Such a group can, for example, prohibit certain transactions, or all of them. It is, therefore,

critical that the protocol be designed such that miners have no incentive to form such large colluding groups.

Empirical evidence shows that Bitcoin miners behave strategically. Specifically, because rewards are distributed at infrequent, random intervals, miners form *mining pools* in order to decrease the variance of their income rate. Within such pools, all members contribute to the solution of each cryptopuzzle, and share the rewards proportionally to their contributions. To the best of our knowledge, such pools have been benign and followed the protocol so far.

Indeed, conventional wisdom has long asserted that the Bitcoin mining protocol is equitable to its participants and secure against malfeasance by a non-majority attacker (Section 7). Barring recently-explored Sybil attacks on transaction propagation,² there were no known techniques by which a minority of colluding miners could earn disproportionate benefits by deviating from the protocol. Because the protocol was believed to reward miners in proportion to their ratio of the mining power, a miner in a large pool was believed to earn the same revenue as it would in a small pool. Consequently, if we ignore the fixed cost of pool operation and potential economies of scale, there is no advantage for colluding miners to organize into ever-increasing pools. Therefore, pool formation by honest rational miners poses no threat to the system.

In this paper, we show that the conventional wisdom is wrong: the Bitcoin mining protocol, as prescribed and implemented, is not incentive-compatible. We describe a strategy that can be used by a minority pool to obtain more revenue than the pool's fair share, that is, more than its ratio of the total mining power.

The key idea behind this strategy, called Selfish Mining, is for a pool to keep its discovered blocks private, thereby intentionally forking the chain. The honest nodes continue to mine on the public chain, while the pool mines on its own private branch. If the pool discovers more blocks, it develops a longer lead on the public chain, and continues to keep these new blocks private. When the public branch approaches the pool's private branch in length, the selfish miners reveal blocks from their private chain to the public.

This strategy leads honest miners that follow the Bitcoin protocol to waste resources on mining cryptopuzzles that end up serving no purpose. Our analysis demonstrates that, while both honest and selfish parties waste some resources, the honest miners waste proportionally more, and the

The original version of this paper was published in *Financial Cryptography and Data Security* (FC 2014). Lecture Notes in Computer Science 8437, Springer Berlin, Heidelberg, 436–454; https://link.springer.com/chapter/10.1007/978-3-662-45472-5_28.

This research was supported by the NSF Trust STC and by DARPA.

selfish pool's rewards exceed its share of the network's mining power, conferring it a competitive advantage and incentivizing rational miners to join the selfish mining pool.

We show that, above a certain threshold size, the revenue of a selfish pool rises superlinearly with pool size above its revenue with the honest strategy. This fact has critical implications for the resulting system dynamics. Once a selfish mining pool reaches the threshold, rational miners will preferentially join selfish miners to reap the higher revenues compared to other pools. Such a selfish mining pool can quickly grow towards a majority. If the pool tips the majority threshold (due to the addition of malicious actors aimed at undermining the system, rational actors wishing to usurp the currency, perhaps covertly, or due to momentum in pool popularity), it can switch to a modified protocol that ignores blocks generated outside the pool, to become the only creator of blocks and reap all the mining revenue. A majority pool wishing to remain covert may remain a benign monopolist, accepting blocks from third-parties on occasion to provide the illusion of decentralization, while retaining the ability to reap full revenue when needed, as well as the ability to launch double-expenditure attacks against merchants. Either way, the decentralized nature of the currency will have collapsed, and a single entity, the selfish pool manager, will control the system.

Since a selfish mining pool that exceeds threshold size poses a threat to the Bitcoin system, we characterize how the threshold varies as a function of message propagation speed in the network. We show that, for a mining pool with high connectivity and good control on information flow, the threshold is close to zero. This implies that, if less than 100% of the miners are honest, the system may not be incentive compatible: The first selfish miner will earn proportionally higher revenues than its honest counterparts, and the revenue of the selfish mining pool will increase superlinearly with pool size.

We further show that the Bitcoin mining protocol will never be safe against attacks by a selfish mining pool that commands more than 1/3 of the total mining power of the network. Such a pool will always be able to collect mining rewards that exceed its proportion of mining power, even if it loses every single block race in the network. The resulting bound of 2/3 for the fraction of Bitcoin mining power that needs to follow the honest protocol to ensure that the protocol remains resistant to being gamed is substantially lower than the 50% figure currently assumed, and difficult to achieve in practice. Finally, we suggest a simple modification to the Bitcoin protocol that achieves a threshold of 1/4. This change is backwards-compatible and *progressive*; that is, it can be adopted by current clients with modest changes, does not require full adoption to provide a benefit, and partial adoption will proportionally increase the threshold.

In summary, the contributions of this work are:

1. Introduction of the Selfish-Mine strategy, which demonstrates that Bitcoin mining is not incentive compatible (Section 3).
2. Analysis of Selfish-Mine, and when it can benefit a pool (Section 4).
3. Analysis of majority-pool formation in face of selfish mining (Section 5).

4. A simple backward-compatible progressive modification to the Bitcoin protocol that would raise the threshold from zero to 1/4 (Section 6).

We provide an overview of related work in Section 7, and discuss the implications of our results in Section 8.

2. PRELIMINARIES

Bitcoin is a distributed, decentralized cryptocurrency.¹⁵ The users of Bitcoin are called *clients*, each of whom can command accounts, known as *addresses*. A client can send Bitcoins to another client by forming a transaction and committing it into a global append-only log called the *blockchain*. The blockchain is maintained by a network of *miners*, which are compensated for their effort in Bitcoins. Bitcoin transactions are protected with cryptographic techniques that ensure only the rightful owner of a Bitcoin address can transfer funds from it.

The miners are in charge of recording the transactions in the blockchain, which determines the ownership of Bitcoins. A client owns x Bitcoins at time t if, in the prefix of the blockchain up to time t , the aggregate of transactions involving that client's address amounts to x . Miners only accept transactions if their inputs are unspent.

2.1. Blockchain and mining

The blockchain records the transactions in units of blocks. Each block includes a unique ID, and the ID of the preceding block. The first block, dubbed *the genesis block*, is defined as part of the protocol. A valid block contains a solution to a cryptopuzzle involving the hash of the previous block, the hash of the transactions in the current block, and a Bitcoin address which is to be credited with a reward for solving the cryptopuzzle. This process is called Bitcoin *mining*, and, by slight abuse of terminology, we refer to the creation of blocks as *block mining*. The specific cryptopuzzle is a double-hash whose result has to be smaller than a set value. The problem difficulty, set by this value, is dynamically adjusted such that blocks are generated at an average rate of one every ten minutes.

Any miner may add a valid block to the chain by simply publishing it over an overlay network to all other miners. If two miners create two blocks with the same preceding block, the chain is *forked* into two *branches*, forming a tree. Other miners may subsequently add new valid blocks to either branch. When a miner tries to add a new block after an existing block, we say it *mines on* the existing block. This existing block may be the head of a branch, in which case we say the miner mines on the head of the branch, or simply on the branch.

The formation of branches is undesirable since the miners have to maintain a globally-agreed totally ordered set of transactions. To resolve forks, the protocol prescribes miners to adopt and mine on the longest chain.^a All miners add

^a The criterion is actually the most difficult chain in the block tree, that is, the one that required (in expectancy) the most mining power to create. To simplify presentation, and because it is usually the case, we assume the set difficulty at the different branches is the same, and so the longest chain is also the most difficult one.

blocks to the longest chain they know of, or the first one they heard of if there are branches of equal length. This causes forked branches to be pruned; transactions in pruned blocks are ignored, and may be resubmitted by clients.

We note that block dissemination over the overlay network takes seconds, whereas the average mining interval is 10min. Accidental bifurcation is therefore rare, and occurs on average once about every 60 blocks.⁷

When a miner creates a block, it is compensated for its efforts with Bitcoins. This compensation includes a pertransaction fee paid by the users whose transactions are included, as well as an amount of new Bitcoins that did not exist before.^b

2.2. Pool formation

The probability of mining a block is proportional to the computational resources used for solving the associated cryptopuzzle. Due to the nature of the mining process, the interval between mining events exhibits high variance from the point of view of a single miner. A single home miner using a custom-made hardware is unlikely to mine a block for years.²² Consequently, miners typically organize themselves into mining *pools*. All members of a pool work together to mine each block, and share their revenues when one of them successfully mines a block. While joining a pool does not change a miner's expected revenue, it decreases the variance and makes the monthly revenues more predictable.

3. THE SELFISH-MINE STRATEGY

First, we formalize a model that captures the essentials of Bitcoin mining behavior and introduces notation for relevant system parameters. Then we detail the selfish mining algorithm.

3.1. Modeling miners and pools

The system is comprised of a set of miners $1, \dots, n$. Each miner i has mining power m_i , such that $\sum_{i=1}^n m_i = 1$. Each miner chooses a chain head to mine, and finds a subsequent block for that head after a time interval that is exponentially distributed with mean m_i^{-1} . We assume that miners are rational; that is, they try to maximize their revenue, and may deviate from the protocol to do so.

A group of miners can form a pool that behaves as single agent with a centralized coordinator, following some strategy. The mining power of a pool is the sum of mining power of its members, and its revenue is divided among its members according to their relative mining power.²¹ The *expected relative revenue*, or simply the *revenue* of a pool is the expected fraction of blocks that were mined by that pool out of the total number of blocks in the longest chain.

3.2. Selfish-mine

We now describe our strategy, called Selfish-Mine. As we show in Section 4, Selfish-Mine allows a pool of sufficient size to obtain a revenue larger than its ratio of mining power. For simplicity, and without loss of generality, we

assume that miners are divided into two groups, a colluding minority pool that follows the selfish mining strategy, and a majority that follows the honest mining strategy (others). It is immaterial whether the honest miners operate as a single group, as a collection of groups, or individually.

The key insight behind the selfish mining strategy is to force the honest miners into performing wasted computations on the stale public branch. Specifically, selfish mining forces the honest miners to spend their cycles on blocks that are destined to not be part of the blockchain.

Selfish miners achieve this goal by selectively revealing their mined blocks to invalidate the honest miners' work. Approximately speaking, the selfish mining pool keeps its mined blocks private, secretly bifurcating the blockchain and creating a private branch. Meanwhile, the honest miners continue mining on the shorter, public branch. Because the selfish miners command a relatively small portion of the total mining power, their private branch will not remain ahead of the public branch indefinitely. Consequently, selfish mining judiciously reveals blocks from the private branch to the public, such that the honest miners will switch to the recently revealed blocks, abandoning the shorter public branch. This renders their previous effort spent on the shorter public branch wasted, and enables the selfish pool to collect higher revenues by incorporating a higher fraction of its blocks into the blockchain.

Armed with this intuition, we can fully specify the selfish mining strategy. The strategy is driven by mining events by the selfish pool or by the others. Its decisions depend only on the relative lengths of the selfish pool's private branch versus the public branch. It is best to illustrate the operation of the selfish mining strategy by going through possible scenarios involving different public and private chain lengths.

When the public branch is longer than the private branch, the selfish mining pool is behind the public branch. Because of the power differential between the selfish miners and the others, the chances of the selfish miners mining on their own private branch and overtaking the main branch are small. Consequently, the selfish miner pool simply adopts the main branch whenever its private branch falls behind. As others find new blocks and publish them, the pool updates and mines at the current public head.

When the selfish miner pool finds a block, it is in an advantageous position with a single block lead on the public branch on which the honest miners operate. Instead of naively publishing this private block and notifying the rest of the miners of the newly discovered block, selfish miners keep this block private to the pool. There are two outcomes possible at this point: either the honest miners discover a new block on the public branch, nullifying the pool's lead, or else the pool mines a second block and extends its lead on the honest miners.

In the first scenario where the honest nodes succeed in finding a block on the public branch, nullifying the selfish pool's lead, the pool immediately publishes its private branch (of length 1). This yields a toss-up where either branch may win. The selfish miners unanimously adopt and extend the previously private branch, while the honest

^b The rate at which the new Bitcoins are generated is designed to slowly decrease towards zero, and will reach zero when almost 21 mn Bitcoins are created. Then, the miners' revenue will be only from transaction fees.

miners will choose to mine on either branch, depending on the propagation of the notifications. If the selfish pool manages to mine a subsequent block ahead of the honest miners that did not adopt the pool's recently revealed block, it publishes immediately to enjoy the revenue of both the first and the second blocks of its branch. If the honest miners mine a block after the pool's revealed block, the pool enjoys the revenue of its block, while the others get the revenue from their block. Finally, if the honest miners mine a block after their own block, they enjoy the revenue of their two blocks while the pool gets nothing.

In the second scenario, where the selfish pool succeeds in finding a second block, it develops a comfortable lead of two blocks that provide it with some cushion against discoveries by the honest miners. Once the pool reaches this point, it continues to mine at the head of its private branch. It publishes one block from its private branch for every block the others find. Since the selfish pool is a minority, its lead will, with high probability, eventually reduce to a single block. At this point, the pool publishes its private branch. Since the private branch is longer than the public branch by one block, it is adopted by all miners as the main branch, and the pool enjoys the revenue of all its blocks. This brings the system back to a state where there is just a single branch until the pool bifurcates it again.

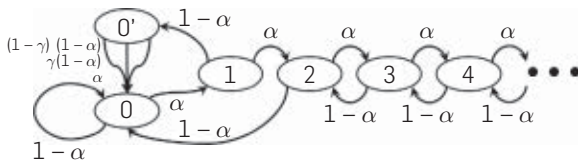
4. ANALYSIS

We can now analyze the expected rewards for a system where the selfish pool has mining power of α and the others of $(1 - \alpha)$.

Figure 1 illustrates the progress of the system as a state machine. The states of the system represent the lead of the selfish pool; that is, the difference between the number of unpublished blocks in the pool's private branch and the length of the public branch. Zero lead is separated to states 0 and 0'. State 0 is the state where there are no branches; that is, there is only a single, global, public longest chain. State 0' is the state where there are two public branches of length one: the main branch, and the branch that was private to the selfish miners, and published to match the main branch. The transitions in the figure correspond to mining events, either by the selfish pool or by the others. Recall that these events occur at exponential intervals with an average frequency of α and $(1 - \alpha)$, respectively.

We can analyze the expected rewards from selfish mining by taking into account the frequencies associated with each state transition of the state machine, and calculating the corresponding rewards. Let us go through the various cases and describe the associated events that trigger state transitions.

Figure 1. State machine with transition frequencies.



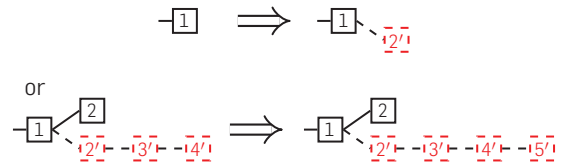
If the pool has a private branch of length 1 and the others mine one block, the pool publishes its branch immediately, which results in two public branches of length 1. Miners in the selfish pool all mine on the pool's branch, because a subsequent block discovery on this branch will yield a reward for the pool. The honest miners, following the standard Bitcoin protocol implementation, mine on the branch they heard of first. We denote by γ the ratio of honest miners that choose to mine on the pool's block, and the other $(1 - \gamma)$ of the non-pool miners mine on the other branch.

For state $s = 0, 1, 2, \dots$, with frequency α , the pool mines a block and the lead increases by one to $s + 1$. In states $s = 3, 4, \dots$, with frequency $(1 - \alpha)$, the honest miners mine a block and the lead decreases by one to $s - 1$. If the others mine a block when the lead is two, the pool publishes its private branch, and the system drops to a lead of 0. If the others mine a block with the lead is 1, we arrive at the aforementioned state 0'. From 0', there are three possible transitions, all leading to state 0 with total frequency 1: (1) the pool mines a block on its previously private branch (frequency α), (2) the others mine a block on the previously private branch (frequency $\gamma(1 - \alpha)$), and (3) the others mine a block on the public branch (frequency $(1 - \gamma)(1 - \alpha)$).

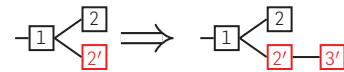
4.1. Revenue

We analyze the state machine and calculate the probabilities of the states $p_0, p_0', p_1, \dots$; the details are in our full report.⁹ The probability distribution over the state space provides the foundation for analyzing the revenue obtained by the selfish pool and by the honest miners. The revenue for finding a block belongs to its miner only if this block ends up in the main chain. We detail the revenues on each event below.

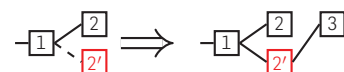
- (a) *Any state but two branches of length 1, pool finds a block.* The pool appends one block to its private branch, increasing its lead on the public branch by one. The revenue from this block will be determined later.



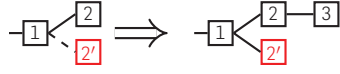
- (b) *Was two branches of length 1, pool finds a block.* The pool publishes its secret branch of length two, thus obtaining a revenue of two.



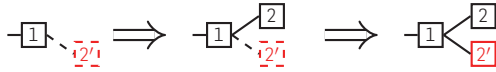
- (c) *Was two branches of length 1, others find a block after pool head.* The pool and the others obtain a revenue of one each—the others for the new head, the pool for its predecessor.



- (d) Was two branches of length 1, others find a block after others' head. The others obtain a revenue of two.

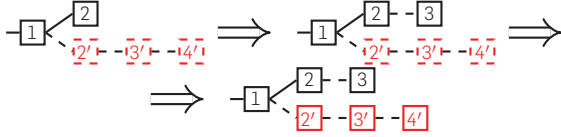


- (e) No private branch, others find a block. The others obtain a revenue of one, and both the pool and the others start mining on the new head.
- (f) Lead was 1, others find a block. Now there are two branches of length one, and the pool publishes its single secret block. The pool tries to mine on its previously private head, and the others split between the two heads.

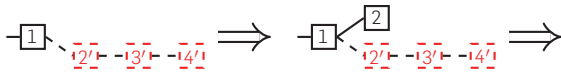


The revenues from both heads cannot be determined yet, because they depend on which branch will win. It will be counted later.

- (g) Lead was 2, others find a block. The others almost close the gap as the lead drops to 1. The pool publishes its secret blocks, causing everybody to start mining at the head of the previously private branch, since it is longer. The pool obtains a revenue of two.



- (h) Lead was more than 2, others find a block. The others decrease the lead, which remains at least two. The new block (say with number i) will end outside the chain once the pool publishes its entire branch, therefore the others obtain nothing. However, the pool now reveals its i 'th block, and obtains a revenue of one for its i 'th block.



We calculate the revenue of the pool and of the others from the state probabilities and transition frequencies:

$$r_{\text{others}} = \overbrace{p_{0'} \cdot \gamma(1-\alpha) \cdot 1}^{\text{Case (c)}} + \overbrace{p_{0'} \cdot (1-\gamma)(1-\alpha) \cdot 2}^{\text{Case (d)}} + \overbrace{p_0 \cdot (1-\alpha) \cdot 1}^{\text{Case (e)}} \quad (1)$$

$$r_{\text{pool}} = \overbrace{p_{0'} \cdot \alpha \cdot 2}^{\text{Case (b)}} + \overbrace{p_{0'} \cdot \gamma(1-\alpha) \cdot 1}^{\text{Case (c)}} + \overbrace{p_2 \cdot (1-\alpha) \cdot 2}^{\text{Case (g)}} + \overbrace{P[i > 2](1-\alpha) \cdot 1}^{\text{Case (h)}} \quad (2)$$

As expected, the intentional branching brought on by selfish mining leads the honest miners to work on blocks that end up outside the blockchain. This, in turn, leads to a drop in the total block generation rate with $r_{\text{pool}} + r_{\text{others}} < 1$. The protocol will adapt the mining difficulty such that the mining rate at the main chain becomes one block per 10min on average. Therefore, the actual revenue rate of each agent

is the *revenue rate ratio*; that is, the ratio of its blocks out of the blocks in the main chain. We substitute the probabilities from Reference⁹ in the revenue expressions of (1)-(2) to calculate the pool's revenue for $0 \leq \alpha \leq \frac{1}{2}$:

$$R_{\text{pool}} = \frac{r_{\text{pool}}}{r_{\text{pool}} + r_{\text{others}}} = \frac{\alpha(1-\alpha)^2(4\alpha + \gamma(1-2\alpha)) - \alpha^3}{1 - \alpha(1 + (2-\alpha)\alpha)} \quad (3)$$

4.2. Simulation

To validate our theoretical analysis, we compare its result with a Bitcoin protocol simulator. The simulator is constructed to capture all the salient Bitcoin mining protocol details described in previous sections, except for the cryptopuzzle module that has been replaced by a Monte Carlo simulator that simulates block discovery without actually computing a cryptopuzzle. In this experiment, we use the simulator to simulate 1000 miners mining at identical rates. A subset of 1000α miners form a pool running the Selfish-Mine algorithm. The other miners follow the Bitcoin protocol. We assume block propagation time is negligible compared to mining time, as is the case in reality. In the case of two branches of the same length, we artificially divide the non-pool miners such that a ratio of γ of them mine on the pool's branch and the rest mine on the other branch. Figure 2 shows that the simulation results match the theoretical analysis.

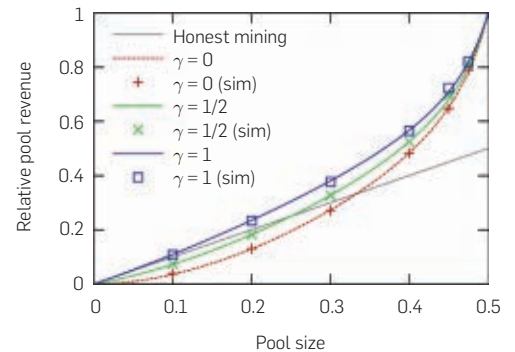
4.3. The effect of α and γ

When the pool's revenue given in Equation 3 is larger than α , the pool will earn more than its relative size by using the Selfish-Mine strategy. Its miners will therefore earn more than their relative mining power. Recall that the expression is valid only for $0 \leq \alpha \leq \frac{1}{2}$. We solve this inequality and phrase the result in the following observation:

OBSERVATION 1. For a given γ , a pool of size α obtains a revenue larger than its relative size for α in the range $\frac{1-\gamma}{3-2\gamma} < \alpha < \frac{1}{2}$.

We illustrate this in Figure 2, where we see the pool's revenue for different γ values with pool size ranging from 0 (very small pool) to 0.5 (half of the miners). Note that the pool is only at risk when it holds exactly one block secret, and the honest miners might publish a block that would compete with it. For $\gamma = 1$, the pool can quickly propagate its one-block branch if the others find their own branch, so all honest miners would

Figure 2. Revenue using the Selfish-Mine strategy for different propagation factors γ , compared to honest mining.



still mine on the pool's block. In this case, the pool takes no risk when following the Selfish-Mine strategy and its revenue is always better than when following the honest algorithm. The threshold is therefore zero, and a pool of any size can benefit by following Selfish-Mine. In the other extreme, $\gamma = 0$, the honest miners always publish and propagate their block first, and the threshold is at $1/3$. With $\gamma = 1/2$ the threshold is at $1/4$. Figure 3 shows the threshold as a function of γ .

We also note that the slope of the pool revenue, R_{pool} , as a function of the pool size is larger than one above the threshold. This implies the following observation:

OBSERVATION 2. *For a pool running the Selfish-Mine strategy, the revenue of each pool member increases with pool size for pools larger than the threshold.*

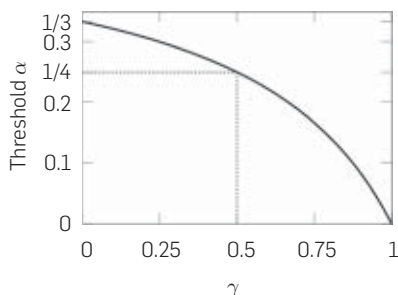
5. POOL FORMATION

We have shown that once a selfish pool's mining power exceeds the threshold, it can increase its revenue by running Selfish-Mine (Theorem 1). At this point, rational miners will preferentially join the selfish pool to increase their revenues. Moreover, the pool's members will want to accept new members, as this would increase their own revenue (Observation 2). The selfish pool would therefore increase in size, unopposed by any mechanism, towards a majority. Once a miner pool, selfish or otherwise, reaches a majority, it controls the blockchain. The Selfish-Mine strategy then becomes unnecessary, since the others are no longer faster than the pool. Instead, a majority pool can collect all the system's revenue by switching to a modified Bitcoin protocol that ignores blocks generated outside the pool; it also has no motivation to accept new members. At this point, the currency is not decentralized as originally envisioned.

6. HARDENING THE PROTOCOL

Ideally, a robust currency system would be designed to resist attacks by groups of colluding miners. Since selfish mining attacks yield positive outcomes for group sizes above the threshold, the protocol should be amended to set the threshold as high as possible. In this section, we argue that the current Bitcoin protocol has no measures to guarantee a low γ . This implies that the threshold may be as low as zero, and a pool of any size can benefit by running Selfish-Mine. We suggest a simple change to the protocol that, if adopted by all non-selfish miners, sets γ to $1/2$, and therefore the threshold to $1/4$.

Figure 3. Mining power α above which selfish mining trumps honest mining, function of the propagation factor γ .



This change is backward compatible: Any subset of the miners can adopt it without hindering the protocol. Moreover, it is progressive: any ratio of the miners that adopts it decreases γ , and therefore increases the threshold.

6.1. Problem

The Bitcoin protocol prescribes that when a miner knows of multiple branches of the same length, it mines and propagates only the first branch it received. Recall that a pool that runs the Selfish-Mine strategy and has a lead of 1 publishes its secret block P once it hears of a competing block X found by a non-pool block. If block P reaches a non-pool miner before block X , that miner will mine on P .

Because selfish mining is reactive, and it springs into action only after the honest nodes have discovered a block X , it may seem to be at a disadvantage. But a savvy pool operator can perform a sybil attack on honest miners by adding a significant number of zero-power miners to the Bitcoin miner network. These virtual miners act as advance sensors by participating in data dissemination, but do not mine new blocks. (Babaioff et al. also acknowledge the feasibility of such a sybil attack²). The virtual miners are managed by the pool, and once they hear of block X , they ignore it and start propagating block P . The random peer-to-peer structure of the Bitcoin overlay network will eventually propagate X to all miners, but the propagation of X under these conditions will be strictly slower than that of block P . By adding enough virtual nodes, the pool operator can thus increase γ . The result (see Observation 1), is a threshold close to zero.

6.2. Solution

We propose a simple, backwards-compatible change to the Bitcoin protocol to address this problem and raise the threshold. Specifically, when a miner learns of competing branches of the same length, it should propagate all of them, and choose which one to mine on uniformly at random. In the case of two branches of length 1, as discussed in Section 4, this would result in half the nodes (in expectancy) mining on the pool's branch and the other half mining on the other branch. This yields $\gamma = 1/2$, which in turn yields a threshold of $1/4$.

Each miner implementing our change decreases the selfish pool's ability to increase γ through control of data propagation. This improvement is independent of the adoption of the change at other miners, therefore it does not require a hard fork. This change to the protocol does not introduce new vulnerabilities to the protocol: Currently, when there are two branches of equal length, the choice of each miner is arbitrary, effectively determined by the network topology and latency. Our change explicitly randomizes this arbitrary choice, and therefore does not introduce new vulnerabilities.

7. RELATED WORK

Decentralized digital currencies have been proposed before Bitcoin, starting with eCash⁶ and followed by peer-to-peer currencies^{23,25}; see Ref. Barber et al. and Miers et al.^{3,14} for short surveys. None of these are centered around a global log; therefore, their techniques and challenges are unrelated to this work.

Several dozen cryptocurrencies have followed Bitcoin's success.²⁴ These currencies are based on a global log, which is extended by the users' efforts. We conjecture that the essential technique of withholding blocks for selfish mining can be directly applied to all such systems.

It was commonly believed that the Bitcoin system is sound as long as a majority of the participants honestly follow the protocol, and the "51% attack" was the chief concern.^{13, 15} The notion of soundness for a nascent, distributed, Internet-wide, decentralized system implies the presence of incentives for adoption of the prescribed protocol, for such incentives ensure a robust system comprised of participants other than enthusiastic and altruistic early adopters. Felten¹⁰ notes that "there was a folk theorem that the Bitcoin system was stable, in the sense that if everyone acted according to their incentives, the inevitable result would be that everyone followed the rules of Bitcoin as written." Others¹⁷ have claimed that "the well-known argument – never proven, but taken on intuitive faith – that a minority of miners can't control the network is a special case of a more general assumption: that a coalition of miners with $X\%$ of the network's hash power can make no more than $X\%$ of total mining revenues." A survey³ on the technical features responsible for Bitcoin's success notes that the Bitcoin design "addresses the incentive problems most expeditiously," while Bitcoin tutorials for the general public hint at incentives designed to align participants' and the system's goals.¹⁹ More formally, Kroll, Davey and Felten's work¹² provides a game-theoretic analysis of Bitcoin, without taking into account block withholding attacks such as selfish mining, and argues that the honest strategy constitutes a Nash equilibrium, implying incentive-compatibility.

Our work shows that the real Bitcoin protocol, which permits block withholding and thereby enables selfish mining-style attacks, does not constitute an equilibrium. It demonstrates that the Bitcoin mining system is not incentive compatible even in the presence of an honest majority. Over 2/3 of the participants need to be honest to protect against selfish mining, under the most optimistic of assumptions.

A distinct exception from this common wisdom is a discussion of maintaining a secret fork in the Bitcoin forums, mostly by users^c btchris, ByteCoin, mtgox, and RHorning.²⁰ The approach, dubbed the Mining Cartel Attack, is inferior to selfish mining in that the cartel publishes two blocks for every block published by the honest nodes. This discussion does not include an analysis of the attack (apart from a brief note on simulation results), does not explore the danger of the resulting pool dynamics, and does not suggest a solution to the problem.

The influential work of Rosenfeld²¹ addresses the behavior of miners in the presence of different pools with different rewards. Although it addresses revenue maximization for miners with a set mining power, this work is orthogonal to the discussion of Selfish Mining, as it centers around the pool reward system. Both selfish pools and honest pools should carefully choose their reward method. Since a large-enough selfish pool would earn more than its mining power, any fair reward method would provide more reward to its miners, so rational miners would choose it over an honest pool.

Recent work² addresses the lack of incentives for disseminating transactions between miners, since each of them prefers to collect the transaction fee himself. This is unrelated to the mining incentive mechanism we discuss.

The Bitcoin blockchain had one significant bifurcation in March 2013 due to a bug.¹ It was solved when the two largest pools at the time manually pruned one branch. This bug-induced fork, and the one-off mechanism used to resolve it, are fundamentally different from the intentional forks that Selfish-Mine exploits.

In a *blockwithholding attack*, a pool member decreases the pool revenue by never publishing blocks it finds. Although it sounds similar to the strategy of Selfish-Mine, the two are unrelated, as our work that deals with an attack by the pool on the system.

Various systems build services on top of the Bitcoin global log, for example, improved coin anonymity,¹⁴ namespace maintenance¹⁶ and virtual notaries.¹¹ These services that rely on Bitcoin are at risk in case of a Bitcoin collapse.

8. DISCUSSION

We briefly discuss below several points at the periphery of our scope.

8.1. System collapse

The Bitcoin protocol is designed explicitly to be decentralized. We therefore refer to a state in which a single entity controls the entire currency system as a collapse of Bitcoin.

Note that such a collapse does not immediately imply that the value of a Bitcoin drops to 0. The controlling entity will have an incentive to accept most transactions, if only to reap their fees, and because if it mines all Bitcoins, it has strong motivation that they maintain their value. It may also choose to remain covert, and hide the fact that it can control the entire currency. An analysis of a Bitcoin monopolist's behavior is beyond the scope of this paper, but we believe that a currency that is de facto or potentially controlled by a single entity may deter many of Bitcoin's clients.

8.2. Detecting selfish mining

There are two telltale network signatures of selfish mining that can be used to detect when selfish mining is taking place, but neither are easy to measure definitively.

The first and strongest sign is that of abandoned (orphaned) chains, where the block race that takes place as part of selfish mining leaves behind blocks that were not incorporated into the blockchain. Unfortunately, it is difficult to definitively account for abandoned blocks, as the current protocol prunes and discards such blocks inside the network. A measurement tool that connects to the network from a small number of vantage points may miss abandoned blocks.

The second indicator of selfish mining activity is the timing gap between successive blocks. A selfish miner who squelches an honest chain of length N with a chain of length $N+1$ will reveal a block very soon after its predecessor. Since normal mining events should be independent, one would expect block discovery times to be exponentially distributed. A deviation from this distribution would be suggestive of mining activity. The problems with this approach are that it detects only

c In alphabetical order.

a subset of the selfish miner's behavior (the transition from state 2 to state 0 in the state machine), the signature behavior occurs relatively rarely, and such a statistical detector may take a long time to accrue statistically significant data.

8.3. Measures and countermeasures

Although miners may choose to collude in a selfish mining effort, they may prefer to hide it in order to avoid public criticism and countermeasures. It is easy to hide Selfish-Mine behavior, and difficult to ban it. A selfish pool may never reveal its size by using different Bitcoin addresses and IP addresses, and by faking block creation times. The rest of the network would not even suspect that a pool is near a dangerous threshold.

Moreover, the honest protocol is public, so if a detection mechanism is set up, a selfish pool would know its parameters and use them to avoid detection. For instance, if the protocol was defined to reject blocks with creation time below a certain threshold, the pool could publish its secret blocks just before this threshold.

A possible line of defense against selfish mining pools is for counter-attackers to infiltrate selfish pools and expose their secret blocks for the honest miners. However, selfish pool managers can, in turn, selectively reveal blocks to subsets of the members in the pool, identify spy nodes through intersection, and expel nodes that leak information.

8.4. Thieves and snowballs

Selfish mining poses two kinds of danger to the Bitcoin ecosystem: selfish miners reap disproportionate rewards, and the dynamics favor the growth of selfish mining pools towards a majority, in a snowball effect. The system would be immune to selfish mining if there were no pools above the threshold size. Yet, since the current protocol has no guaranteed lower bound on this threshold, it cannot automatically protect against selfish miners.

Even with our proposed fix that raises the threshold to 25%, the system remains vulnerable: there already exist pools whose mining power exceeds the 25% threshold,¹⁸ and at times, even the 33% theoretical hard limit. Responsible miners should therefore break off from large pools until no pool exceeds the threshold size.

8.5. Responsible disclosure

Because of Bitcoin's decentralized nature, selfish mining can only be thwarted by collective, concerted action. There is no central repository, no push mechanism and no set of privileged developers; all protocol modifications require public discussion prior to adoption. In order to promote a swift solution and to avoid a scenario where some set of people had the benefit of selective access, we published a preliminary report⁹ and explained both the problem and our suggested solution in public forums.⁸

9. CONCLUSION

Bitcoin is the first widely popular cryptocurrency with a broad user base and a rich ecosystem, all hinging on the incentives in place to maintain the critical Bitcoin blockchain. Our results show that Bitcoin's mining protocol is not

incentive-compatible. We presented Selfish-Mine, a mining strategy that enables pools of colluding miners that adopt it to earn revenues in excess of their mining power. Higher revenues can lead new miners to join a selfish miner pool, a dangerous dynamic that enables the selfish mining pool to grow towards a majority. The Bitcoin system would be much more robust if it were to adopt an automated mechanism that can thwart selfish miners. We offer a backwards-compatible modification to Bitcoin that ensures that pools smaller than 1/4 of the total mining power cannot profitably engage selfish mining. We also show that at least 2/3 of the network needs to be honest to thwart selfish mining; a simple majority is not enough.

Acknowledgments

We are grateful to Raphael Rom, Fred B. Schneider, Eva Tardos, and Dror Kronstein for their valuable advice on drafts of this paper, as well as our shepherd Rainer Böhme. 

References

- Andresen, G. March 2013 chain fork post-mortem. BIP 50, en.bitcoin.it/wiki/BIP_50, retrieved Sep. 2013.
- Babaioff, M., Dobzinski, S., Oren, S., Zohar, A. On Bitcoin and red balloons. In *EC* (ACM, 2012).
- Barber, S., Boyen, X., Shi, E., Uzun, E. Bitter to better, how to make Bitcoin a better currency. In *FC* (2012).
- bitcoincharts.com. Bitcoin network. bitcoincharts.com/bitcoin/ (Nov. 2013).
- blockchain.info. Bitcoin market capitalization. blockchain.info/charts/market-cap (Jan. 2014).
- Chaum, D. Blind signatures for untraceable payments. In *Crypto 82* (1982), 199–203.
- Decker, C., Wattenhofer, R. Information propagation in the Bitcoin network. In *P2P* (IEEE, 2013).
- Eyal, I., Sirer, E.G. Bitcoin is broken. hackingdistributed.com/2013/11/04/bitcoin-is-broken/ (2013).
- Eyal, I., Sirer, E.G. Majority is not enough: Bitcoin mining is vulnerable. *arXiv preprint arXiv:1311.0243* (2013).
- Felten, E.W. Bitcoin research in Princeton CS. freedom-to-tinker.com/blog/felten/bitcoin-research-in-princeton-cs/ (2013).
- Kelkar, A., Bernard, J., Joshi, S., Premkumar, S., Sirer, E.G. Virtual notary. virtual-notary.org/ (Retrieved Sep. 2013).
- Kroll, J.A., Davey, I.C., Felten, E.W. The economics of Bitcoin mining or, Bitcoin in the presence of adversaries. In *Workshop on the Economics of Information Security* (2013).
- Lee, T.B. Four reasons Bitcoin is worth studying. forbes.com/sites/timothylee/2013/04/07/four-reasons-bitcoin-is-worth-studying/2/ (2013).
- Miers, I., Garman, C., Green, M., Rubin, A.D. Zerocoin: Anonymous distributed e-cash from Bitcoin. In *IEEE Symposium on Security and Privacy* (2013).
- Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system (2008).
- Namecoin Project. Namecoin DNS – DotBIT project. dot-bit.org (Retrieved Sep. 2013).
- Narayanan, A., Miller, A. Why the Cornell paper on Bitcoin mining is important. freedom-to-tinker.com/blog/randomwalker/why-the-cornell-paper-on-bitcoin-mining-is-important/ (2013).
- Neighborhood Pool Watch. October 27th 2013 weekly pool and network statistics. organoforti.blogspot.com/2013/10/october-27th-2013-weekly-pool-and.html (Retrieved Oct. 2013).
- Pacia, C. Bitcoin mining explained like you're five: Part 1 – incentives. chrispacia.wordpress.com/2013/09/02/bitcoin-mining-explained-like-youre-five-part-1-incentives/ (September 2013).
- RHorning, mtgox, btchris, and ByteCoin. Mining cartel attack. bitcointalk.org/index.php?topic=2227, December 2010.
- Rosenfeld, M. Analysis of Bitcoin pooled mining reward systems. *arXiv preprint arXiv:1112.4980* (2011).
- Swanson, E. Bitcoin mining calculator. alloscomp.com/bitcoin/calculator (Retrieved Sep. 2013).
- Vishnumurthy, V., Chandrakumar, S., Sirer, E.G. Karma: A secure economic framework for peer-to-peer resource sharing. In *Workshop on Economics of Peer-to-Peer Systems* (2003).
- Wikipedia. List of cryptocurrencies. en.wikipedia.org/wiki/List_of_cryptocurrencies (Oct. 2013).
- Yang, B., Garcia-Molina, H. PPay: Micropayments for peer-to-peer systems. In *CCS* (ACM, 2003).

Ittay Eyal (ittay@technion.ac.il), Electrical Engineering, Technion, Initiative for Cryptocurrencies and Contracts, Haifa, Israel.

Emin Gün Sirer (egs@systems.cs.cornell.edu), Computer Science, Cornell University, Initiative for Cryptocurrencies and Contracts, Ithaca, NY, USA.



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, Statistics, Applied Math, Artificial Intelligence, 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/NewsDetail.asp?id=373>

Deadline: The positions will be open until they are filled by appropriate candidates.

Southern University of Science and Technology (SUSTech) Tenure-Track Faculty Positions

The Department of Computer Science and Engineering (CSE, <http://cse.sustc.edu.cn/en/>), Southern University of Science and Technology (SUSTech) has multiple Tenure-track faculty openings at all ranks, including Professor/Associate Professor/Assistant Professor. We are looking for outstanding candidates with demonstrated research achievements and keen interest in teaching, in the following areas (but are not restricted to):

- Data Science
- Artificial Intelligence
- Computer Systems (including Networks, Cloud Computing, IoT, Software Engineering, etc.)
- Cognitive Robotics and Autonomous Systems
- Cybersecurity (including Cryptography)

Applicants should have an earned Ph.D. degree and demonstrated achievements in both research and teaching. The teaching language at SUSTech is bilingual, either English or Putonghua. It is perfectly acceptable to use English in all lectures, assignments, exams. In fact, our existing faculty members include several non-Chinese speaking professors.

Established in 2012, the Southern University of Science and Technology (SUSTech) is a public institution funded by the municipal of Shenzhen, a special economic zone city in China. Shenzhen is a major city located in South-

ern China, situated immediately north to Hong Kong Special Administrative Region. As one of China's major gateways to the world, Shenzhen is the country's fastest-growing city in the past two decades. The city is the high-tech and manufacturing hub of southern China. As a picturesque coastal city, Shenzhen is also a popular tourist destination and was named one of the world's 31 must-see tourist destinations in 2010 by The New York Times. Shenzhen ranks the 66th place on the 2017 Global City Competitiveness List, released by the National Academy of Economic Strategy, the Chinese Academy of Social Sciences and United Nations Habitat. By the end of 2016, there were around 20 million residents in Shenzhen.

SUSTech is a pioneer in higher education reform in China. The mission of the University is to become a globally recognized research university which emphasizes academic excellence and promotes innovation, creativity and entrepreneurship.

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.

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.

The Digital Engineering Faculty, established jointly by Hasso Plattner Institute and the University of Potsdam, is one of Germany's excellence centers in Computer Science and IT Systems Engineering.

Annually, HPI's Research School grants

10 Ph.D. and Postdoctoral Scholarships.

With its interdisciplinary and international structure, the Research School interconnects HPI's research groups as well as its branches at the University of Cape Town, Technion, and Nanjing University.

HPI RESEARCH GROUPS

- **Algorithm Engineering**, Prof. Dr. Tobias Friedrich
- **Business Process Technology**, Prof. Dr. Mathias Weske
- **Computer Graphics Systems**, Prof. Dr. Jürgen Döllner
- **Digital Health**, Prof. Dr. Erwin Böttinger
- **Enterprise Platform and Integration Concepts**, Prof. Dr. h.c. Hasso Plattner
- **Human Computer Interaction**, Prof. Dr. Patrick Baudisch
- **Information Systems**, Prof. Dr. Felix Naumann
- **Internet Technologies and Systems**, Prof. Dr. Christoph Meinel
- **Operating Systems and Middleware**, Prof. Dr. Andreas Polze
- **Software Architecture**, Prof. Dr. Robert Hirschfeld
- **System Analysis and Modeling**, Prof. Dr. Holger Giese

Applications must be submitted by August 15 of the respective year.
For more information on HPI's Research School please visit:
www.hpi.de/research-school





DOI:10.1145/3219818

Dennis Shasha

Upstart Puzzles

String Wars

SUPPOSE SOMEONE GIVES you two strings: X and Y . Your goal is to design a minimum-cost collection of smaller strings $\text{coll}(X|Y)$ that match and cover every character of string X with order independence without matching any substring of string Y .

Let us first break down that last sentence:

The collection of strings in $\text{coll}(X|Y)$ may have duplicates;

Matching and covering every character of string X means the strings in $\text{coll}(X|Y)$ should tile string X without overlaps or gaps, and every tile should exactly match an underlying substring of X ;

Not matching a substring of string Y means there should be no exact match of any string in $\text{coll}(X|Y)$ to any substring of string Y ; and

Order independence means no matter in which order the strings of $\text{coll}(X|Y)$ is introduced and where they match, X will be tiled once the last string in $\text{coll}(X|Y)$ is introduced.

When it satisfies all these properties, $\text{coll}(X|Y)$ is called a “proper covering of X with respect to Y .” The cost of $\text{coll}(X|Y)$ is the sum of the squares of each element in the collection, including the duplicates.

Here is a simple example to get started. If string X is `aaaaaaaaaa` and Y is `bbbbbbbbbbb`, then $\text{coll}(X|Y)$ consisting of 10 instances of “a” will be a proper covering. No instance of “a” will match any substring (letter, in this case) in Y . $\text{coll}(X|Y)$ is order-independent since the elements of $\text{coll}(X|Y)$ can be introduced in any order; all are just the single letter “a” after all. Further, the (total) cost is 10, because each “a” costs 1.

abaabaabaaba
bbabbbbaabba

A minimum-cost proper covering of the red string of characters with respect to the blue string is `aba, aba, aba, aba` for a cost of $4 \times 9 = 36$. A minimum-cost proper covering of the blue string with respect to the red string is `abba, bbba, bbab` for a cost of $3 \times 16 = 48$. The red string thus “beats” the blue string. Can you find a string that beats the red string?

Warm-Up 1. Continuing with this example, suppose X were `ababababab` and Y were `aaaaaaaaa`. What would be a proper covering of X with respect to Y ?

Solution to first warm-up. Five strings that are “ab” yielding a total cost of 20.

Warm-Up 2. Suppose X were `abaa-babab` and Y were `bbababba`. What would be a proper covering for X with respect to Y ?

Solution to second warm-up. $\text{coll}(X|Y) = \{\text{abaa}, \text{babab}\}$. Breaking up either of these strings into shorter strings would entail some matches with Y .

Challenge. Given the scenario of the first warm-up, what would be a minimum-cost collection $\text{coll}(Y|X)$ for Y that would cover Y with respect to X ?

Solution. Note that five instances of “aa” would *not* be an order-independent cover of Y with respect to X . The reason is that, for example, one “aa” might match the second and third letters of Y , thus preventing a tiling, because no element would cover the first letter of Y . In fact, only $\text{coll}(Y|X) = \{\text{aaaaaaaaa}\}$ would work. That would have a cost of $10 \times 10 = 100$.

We see that an inexpensive order-independent covering of X may not work when elements of the covering might match Y . This brings up the possibility

that an adversary—perhaps nature in the motivating use case of molecular biology—might create a Y that would greatly increase the cost of covering X .

String War Challenge. With respect to $X = \text{abaabaabaaba}$, the red string in the figure here, can you design a string Y of length 12 that can beat X ? That is, we seek a Y such that the minimum-cost proper covering of Y with respect to X costs less than the minimum-cost proper covering of X with respect to Y .

Solution. $Y = \text{bbbbbabbbaba}$. $\text{coll}(Y|X) = \{\text{bbbbba}, \text{bbaba}\}$ having cost $36 + 36 = 72$. $\text{coll}(X|Y) = \{\text{abaabaabaaba}\}$ having cost 144.

String War Upstart. Given an X , can you always design a Y of the same length as X such that Y beats X ? If so, design an algorithm to do so. Can you also design an algorithm to give a maximal difference in cost?

All are invited to submit their solutions to upstartpuzzles@cacm.acm.org; solutions to upstarts and discussion will be posted at <http://cs.nyu.edu/cs/faculty/shasha/papers/cacmpuzzles.html>

Dennis Shasha (dennishasha@yahoo.com) is a professor of computer science in the Computer Science Department of the Courant Institute at New York University, New York, USA, as well as the chronicler of his good friend the omniheurist Dr. Ecco.

Copyright held by the author.



LEARN
THE FUTURE
OF DATA
COMMUNICATION

SIGCOMM 2018

BUDAPEST

AUGUST 20-25

Mentoring
One-on-one meetings
Main conference
Industrial demos
Posters and demos
Student research competition
Tutorials Workshops
Topic preview Travel grants
Hackathon

conferences.sigcomm.org/sigcomm/2018



GENERAL CHAIRS

Sergey Gorinsky / IMDEA, Spain
János Tapolcai / BME, Hungary

TREASURER

Tilman Wolf / UMass, USA

PROGRAM COMMITTEE CHAIRS

Mark Handley / UCL, UK
Dina Katabi / MIT, USA

WORKSHOP CHAIRS

Xiaoming Fu / GAU, Germany
K. K. Ramakrishnan / UCR, USA



Association for
Computing Machinery



acm sigcomm



CROSSOVER

The 11th ACM SIGGRAPH Conference
and Exhibition on Computer Graphics
and Interactive Techniques in Asia

CONFERENCE 4 – 7 December 2018
EXHIBITION 5 – 7 December 2018
Tokyo International Forum, Japan

SA2018.SIGGRAPH.ORG

Sponsored by



Organized by

