

COMMUNICATIONS

CACM.ACM.ORG OF THE ACM 01/2018 VOL.61 NO.01

Decentralized Blockchain-Based Electronic Marketplaces

Information Hiding

The New Jobs

Computing Professionals
for Social Responsibility

Ask Not What Your PostDoc
Can Do For You

Halide

Association for
Computing Machinery



MobileHCI 2018

Beyond Mobile: The Next 20 Years

September 3rd - 6th
Barcelona, Spain



MobileHCI 2018, the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services, will take place in Barcelona (Spain). The conference chairs, Lynne Baillie and Nuria Oliver, invite researchers, academics, students, and industry leaders working in the field of Mobile Human-Computer Interaction, to submit their work and to join them for a rich, four-day program featuring paper presentations, poster presentations, demonstrations, tutorials, specialized workshops, industrial case studies, doctoral consortium, and social events.

We look forward to seeing you in Barcelona!

Deadlines:

February 9th:

Full and short papers

February 28th:

Workshop Proposals

Mid May:

Student Volunteer application

Mid April – Mid May:

Tutorials, Industrial perspectives, Demos, Late Breaking Work, Doctoral consortium

mobilehci.acm.org/2018



Inviting Young Scientists



Association for
Computing Machinery

Meet Great Minds in Computer Science and Mathematics

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

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

The Heidelberg Laureate Forum is **September 23–28, 2018** in Heidelberg, Germany.

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

Who can participate?

New and recent Ph.Ds, doctoral candidates, other graduate students pursuing research, and undergraduate students with solid research experience and a commitment to computing research

How to apply:

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

What is the schedule?

The application process is open between **November 6, 2017** and **February 9, 2018**.

We reserve the right to close the application website early depending on the volume

Successful applicants will be notified by **mid April 2018**.

More information available on Heidelberg social media



Departments

- 7 **Cerf's Up**
**The Role of Archives
in Digital Preservation**
By Vinton G. Cerf
-
- 9 **Vardi's Insights**
**Computer Professionals
for Social Responsibility**
By Moshe Y. Vardi
-
- 10 **Letters to the Editor**
A Leap from Artificial to Intelligence
-
- 12 **BLOG@CACM**
The Big IDEA and the PD Pipeline
Former Computer Science
Teachers Association executive
director Mark R. Nelson discusses
his work with the group to
overcome core challenges to
computer science education.
-
- 31 **Calendar**
-
- 116 **Careers**
-
- 121 **ACM Code of Ethics
and Professional Conduct**
**ACM Code of Ethics:
A Guide for Positive Action**
*By Don Gotterbarn, Amy Bruckman,
Catherine Flick, Keith Miller,
and Marty J. Wolf*

Last Byte

- 120 **Upstart Puzzles**
Polychromatic Choreography
By Dennis Shasha

News

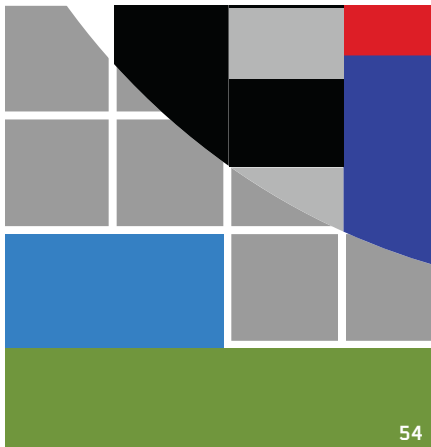


- 15 **Feeling Sounds, Hearing Sights**
A new wave of sensory substitution
devices work to assist people
who are blind or deaf.
By Gregory Mone
-
- 18 **Smartphone Science**
New portable scientific instruments
are taking shape, thanks to
mobile processors and innovative
data-gathering techniques.
By Alex Wright
-
- 21 **The New Jobs**
As automation takes on
more and more tasks, what will
human workers do?
By Marina Krakovsky

Viewpoints

- 26 **Technology Strategy and Management**
The Sharing Economy Meets Reality
Assessing the uncertainties
of the business models
driving the sharing economy.
By Michael A. Cusumano
-
- 29 **Law and Technology**
**How Law and Computer Science
Can Work Together to Improve
the Information Society**
Seeking to remedy bad legislation
with good science.
By Chris Marsden
-
- 32 **Historical Reflections**
**Defining American Greatness:
IBM from Watson to Trump**
Reflections on a firm that
encapsulated the American Century.
By Thomas Haigh
-
- 38 **Viewpoint**
**Technology and the Failure
of the University**
Considering the double-edged
sword of learning technologies
in various academic settings.
By Henry C. Lucas, Jr.
-
- 42 **Viewpoint**
**Ask Not What Your Postdoc
Can Do for You ...**
Seeking more effective strategies
for training and nurturing
CS postdocs to ensure their success.
*By Chitta Baral, Shih-Fu Chang,
Brian Curless, Partha Dasgupta,
Julia Hirschberg, and Anita Jones*

Practice



54

- 46 **Network Applications Are Interactive**
The network era requires new models, with interactions instead of algorithms.
By Antony Alappatt
- 54 **Abstracting the Geniuses Away from Failure Testing**
Ordinary users need tools that automate the selection of custom-tailored faults to inject.
By Peter Alvaro and Severine Tymon
- 62 **Cache Me If You Can**
Building a decentralized Web-delivery model.
By Jacob Loveless

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

About the Cover:



Blockchain technologies are positioned to transform the business world, connecting buyers and sellers and keeping records of their digital transactions clean and secure. This month's cover story (p. 78) explores the advantages of decentralized e-marketplaces using blockchain technology as the facilitator. Cover illustration by Spooky Pooka at Debut Art.

Contributed Articles



78

- 70 **Popularity Spikes Hurt Future Chances for Viral Propagation of Protomemes**
Once a meme gets popular, it will have to evolve to keep being popular.
By Michele Coscia
- 78 **Decentralized Blockchain-Based Electronic Marketplaces**
In a decentralized marketplace, buyers and sellers transact directly, without manipulation by intermediary platforms.
By Hemang Subramanian



Watch the author discuss his work in this exclusive *Communications* video.
<https://cacm.acm.org/videos/decentralized-blockchain-based-e-marketplaces>

Review Articles

- 86 **Information Hiding: Challenges for Forensic Experts**
The practice of hiding ill-gotten data in digital objects is rising among cyber thieves. New initiatives serve to educate, train, and thwart these activities.
By Wojciech Mazurczyk and Steffen Wendzel



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

Research Highlights

- 96 **Technical Perspective**
Moving Spectrum
By David C. Parkes
- 97 **Deep Optimization for Spectrum Repacking**
By Neil Newman, Alexandre Fréchet, and Kevin Leyton-Brown
- 105 **Technical Perspective**
Can High Performance Be Portable?
By Manuel Chakravarty
- 106 **Halide: Decoupling Algorithms from Schedules for High-Performance Image Processing**
By Jonathan Ragan-Kelley, Andrew Adams, Dillon Sharlet, Connelly Barnes, Sylvain Paris, Marc Levoy, Saman Amarasinghe, and Frédo Durand

Nominees for ACM's 2018 General Election

In accordance with the Constitution and Bylaws of the ACM, the Nominating Committee hereby submits the following slate of nominees for ACM's officers. In addition to the officers of the ACM, two Members at Large will be elected. The names of the candidates for each office are presented in random order below:

President (1 July 2018 – 30 June 2020):

Jack Davidson, University of Virginia

Cherri Pancake, Oregon State University

Vice President (1 July 2018 – 30 June 2020):

Moshe Y. Vardi, Rice University

Elizabeth Churchill, Google

Secretary/Treasurer (1 July 2018 – 30 June 2020):

Yannis Ioannidis, "Athena" Research Center and University of Athens

Mehran Sahami, Stanford University

Members at Large (1 July 2018 – 30 June 2022):

Claudia Bauzer Medeiros, University of Campinas, Brazil

Nenad Medvidović, University of Southern California

PJ Narayanan, IIIT Hyderabad

Theo Schlossnagle, Circonus

The Constitution and Bylaws provide that candidates for elected offices of the ACM may also be nominated by petition of one percent of the Members who as of **1 November 2017** are eligible to vote for the nominee. Such petitions must be accompanied by a written declaration that the nominee is willing to stand for election. The number of Member signatures required for the offices of President, Vice President, Secretary/Treasurer and Members at Large, is **653**.

The Bylaws provide that such petitions must reach the Elections Committee before **31 January 2018**.

Original petitions for ACM offices are to be submitted to the ACM Elections Committee, c/o Pat Ryan, COO, ACM Headquarters, 2 Penn Plaza, Suite 701, New York, NY 10121, USA, by **31 January 2018**. Duplicate copies of the petitions should also be sent to the Chair of the Elections Committee, Gerry Segal, c/o ACM Headquarters. Statements and biographical sketches of all candidates will appear in the May 2018 issue of *Communications of the ACM*.

The Nominating Committee would like to thank all those who helped us with their suggestions and advice.

Alexander L. Wolf, CHAIR,

Karin Breitman, **Judith Gal-Ezer**, **Satoshi Matsuoka**, **Rashmi Mohan**





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 Cappo
Director, Office of Publications
 Scott E. Delman

ACM COUNCIL

President
 Vicki L. Hanson
Vice-President
 Cherri M. Pancake
Secretary/Treasurer
 Elizabeth Churchill
Past President
 Alexander L. Wolf
Chair, SGB Board
 Jeanna Matthews
Co-Chairs, Publications Board
 Jack Davidson and Joseph Konstan
Members-at-Large
 Gabriele Anderst-Kotis; Susan Dumais;
 Elizabeth D. Mynatt; Pamela Samuelson;
 Eugene H. Spafford
SGB Council Representatives
 Paul Beame; Jenna 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
 Dame Professor Wendy Hall
ACM India Council
 Srinivas Padmanabhuni
ACM China Council
 Jianguang Sun

PUBLICATIONS BOARD

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

ACM U.S. Public Policy Office
 1701 Pennsylvania Ave NW, Suite 300,
 Washington, DC 20006 USA
 T (202) 659-9711; F (202) 667-1066

Computer Science Teachers Association
 Deborah Seehorn,
 Interim 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
 Andrii Borys

Associate Art Director
 Margaret Gray

Assistant Art Director
 Mia Angelica Balaquiot

Production Manager
 Bernadette Shade

Advertising Sales Account Manager
 Iliia Rodriguez

Columnists

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

CONTACT POINTS

Copyright permission
 permissions@hq.acm.org
Calendar items
 calendar@cacm.acm.org
Change of address
 acmhhelp@acm.org
Letters to the Editor
 letters@cacm.acm.org

WEBSITE
<http://cacm.acm.org>

AUTHOR GUIDELINES
<http://cacm.acm.org/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
 Iliia 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
 aic@cacm.acm.org

Deputy to the Editor-in-Chief
 Lihan Chen
 cacm.deputy.to.aic@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; Mark Guzdial;
 Richard Ladner; Carl Landwehr;
 Beng Chin Ooi; Loren Terveen;
 Marshall Van Alstyne; Jeannette Wing

PRACTICE

Chair
 Stephen Bourne and Theo Schlossnagle
Board Members
 Eric Allman; Samy Bahra; Peter Bailis;
 Terry Coatta; Stuart Feldman; Nicole Forsgren;
 Camille Fournier; Benjamin Fried;
 Pat Hanrahan; 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;
 Elisa Bertino; Gilles Brassard; 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;
 Ron Shamir; Alex Smola; Josep Torrellas;
 Michael Vitale; Hannes Werthner;
 Reinhard Wilhelm

RESEARCH HIGHLIGHTS

Co-Chairs
 Azer Bestavros and Gregory Morrisett
Board Members
 Martin Abadi; Amr El Abbadi; Sanjeev Arora;
 Michael Backes; Maria-Florina Balcan;
 Andrei Broder; Doug Burger; Stuart K. Card;
 Jeff Chase; Jon Crowcroft; Alexei Efros;
 Alon Halevy; Sven Koenig; Steve Marschner;
 Tim Roughgarden; Guy Steele, Jr.;
 Margaret H. Wright; Nikolai Zeldovich;
 Andreas Zeller

WEB

Chair
 James Landay
Board Members
 Marti Hearst; Jason I. Hong;
 Jeff Johnson; Wendy E. MacKay

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 acmhhelp@acm.org.

COMMUNICATIONS OF THE ACM

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

POSTMASTER

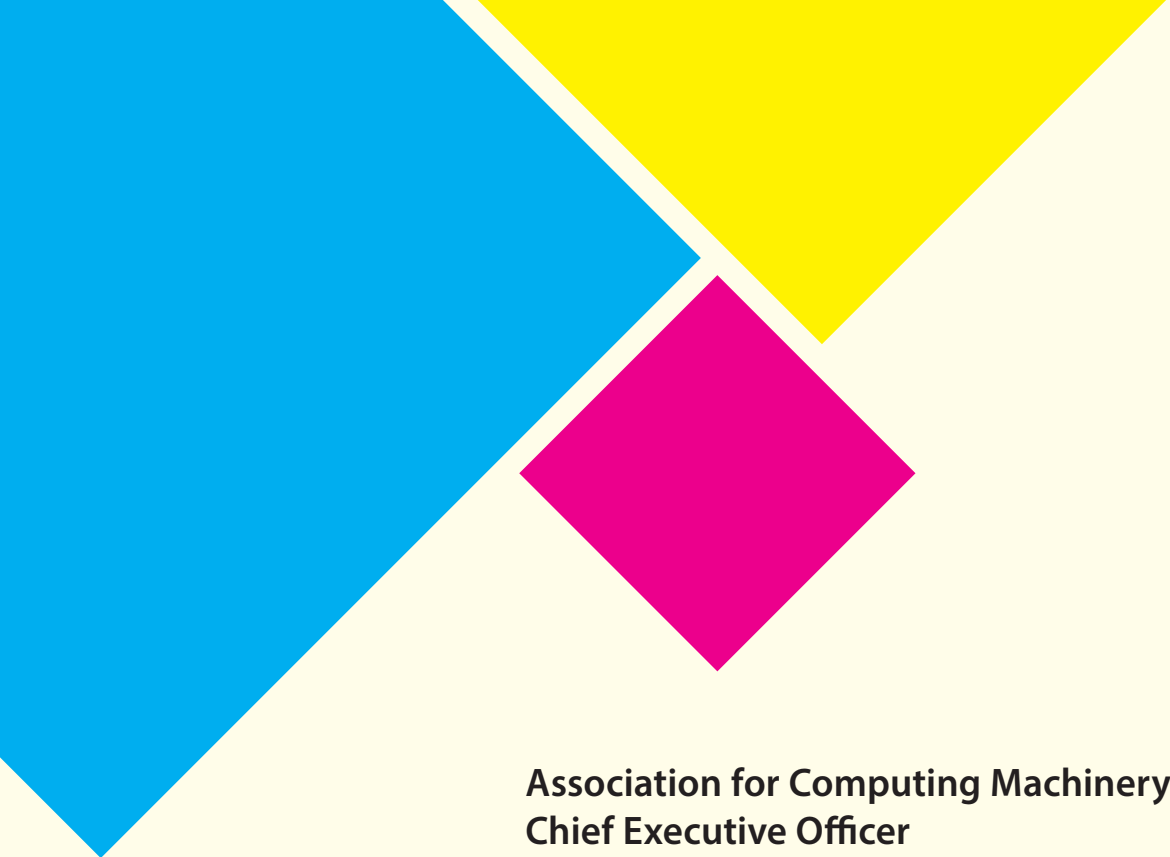
Please send address changes to *Communications of the ACM*
 2 Penn Plaza, Suite 701
 New York, NY 10121-0701 USA

Printed in the U.S.A.



Association for Computing Machinery





Association for Computing Machinery (ACM) Chief Executive Officer

ACM, the Association for Computing Machinery, invites applications for the position of Chief Executive Officer (CEO).

ACM is the oldest and largest educational and scientific computing society with nearly 100,000 members worldwide. The association has an annual budget of \$75 million, 75 full-time staff in New York and Washington D.C., a rich publications program that includes 90 periodicals in computing and hundreds of conference proceedings, a dynamic set of special interest groups (SIGs) that run nearly 300 conferences/symposia/workshops each year, initiatives in India, China, and Europe, and educational and public policy initiatives. ACM is the world's premiere computing society.

The ACM CEO serves as the primary executive responsible for the formulation and implementation of ACM strategic direction, for representing ACM in the worldwide computing community, and for overall management of the affairs of the association. The successful candidate will have high professional standing in the computing field, executive experience, leadership skills, and a vision of the future of professional societies and computing. The CEO reports to the ACM President. It is not a requirement that the CEO work from ACM's New York headquarters, but must be able to travel frequently to headquarters and other ACM meetings. The full job description and details on how to apply can be found at: ceosearch.acm.org

The ACM is an equal opportunity employer. All qualified applicants will receive consideration for employment without regard to race, color, religion, sex, national origin, age, protected veteran status or status as an individual with disability.



**Association for
Computing Machinery**



Vinton G. Cerf

DOI:10.1145/3169085

The Role of Archives in Digital Preservation

I HAD THE pleasure of spending a week in Mexico City to participate in the annual meeting of the International Council on Archives^a hosted by its Latin American branch, the Association of Latin American Archivists^b (ALA). This was a large and wide-ranging conference that stimulated many thoughts and ideas that affected my view of the digital archiving challenge. It is clear there are many archival institutions pursuing the problem out of necessity. They are receiving or collecting artifacts created or rendered in digital form. Many of these items are static in the sense they could be rendered and preserved using older techniques such as print or even microfilm. There is a concept of *variable fidelity* in which some aspect of the digital artifact is imperfectly captured. For example, a text file might be retained in readable but not editable form. This does not work well with objects such as spreadsheets whose value is in the interactive use of the computations captured in the spreadsheet. But one might be willing to give up some resolution if it meant the difference between capturing a portion of versus all of a collection of digital images.

The independent operation of archives may not produce the kind of mutual reinforcement that would arise from inventing and adopting standards for representation and transmission of content, metadata, and other information establishing the origins and history of the artifact. If such standards could be developed and implemented, it might make it possible for one archive to ingest the content of another, should circumstances demand it. Moreover,

such standards would make it possible for one archive to reference the content of another—making it possible to achieve a kind of global discovery of relevant content. Some readers of this column will know there are already a number of standards in place, one of the most important of which is the Open Archival Information System reference model^c (OAIS). This is a significant specification of the desirable functionality of any archive and has the potential to establish interoperability among archives. This property, interoperability, will require that the standards for transmission of digital archival information include descriptors of format and semantics of the nature and structure of the information, its metadata and provenance, intellectual property considerations among other things. Another powerful example is the Digital Object Architecture^d developed by the Corporation for National Research Initiatives.

Some readers may wonder whether the World Wide Web is already an example of an archive. It is not because there is no assurance of long-lasting storage or accessibility of the content. To be sure, the WWW content is a candidate for archiving and the Internet Archive^e is attempting to do exactly that.

A significant test of any specification is that multiple, independent implementations can be shown to interwork. That is, information from one archive can be successfully accessed and transferred to another archive without losing critical archival information. Establishing such standards and allowing archives to be networked could create an ecosystem with substantial resilience.

Multiple instances of content could be found in distinct archives. The impact of institutional failure of an archive (For example, lack of funding, physical destruction) could be mitigated by storing information in multiple locations; an instantiation of the LOCKSS (lots of copies keeps stuff safe) principle.

During this meeting it also became apparent that archivists of digital content will need to be prescriptive about the structure and encoding of digital artifacts they can process into the archive successfully. A proactive stance could improve the likelihood that digital content can be acquired and preserved. Again, standards can help and the software industry can assist by working toward the creation of applications that produce archive-ready content. Since some digital artifacts require software to be used or rendered, it stands to reason that software itself must also be archived and executable over the long term. That includes applications and operating systems and perhaps, detailed functional descriptions of hardware to the extent that the instruction set of the computer can be emulated at need.

It should be clear by now that archives will have to have privileges to acquire and execute applications and the necessary operating system(s) so that acquired artifacts can live for long time periods. Moreover, a long-lived archive must rely on a business model that matches the desired longevity that might extend 100 years or more. Think of archives housing 1,000-year-old vellum codices, for example. In addition to the extensible standardization of digital representation, then, we will need legal and financial frameworks that assist long-term preservation. We must pursue these objectives lest our digital history be lost in a mist of uninterpretable bits. □

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://www.ica.org/en>

b <http://bit.ly/2i4myzg>

c <http://bit.ly/2zSkXb6>

d <http://bit.ly/2ArYa5c>

e <https://archive.org/>

Code 2018: Updating the ACM Code of Ethics

The ACM Code of Ethics and Professional Conduct (The Code) outlines fundamental ethical considerations to which all ACM members are expected to adhere.

The Code consists of principles for personal responsibility and guidelines for dealing with many issues computing professionals are likely to face. The Code is intended to serve as a basis for ethical decision making in the conduct of professional work, and includes considerations for individuals in leadership roles.

The ACM Committee on Professional Ethics (COPE) is updating the ACM Code of Ethics and would like your input.

The current ACM Code of Ethics was adopted in 1992, and much has changed in the 25 years since then. We are updating The Code to reflect the shifts in both technology and society.

The 2018 Code is meant to be an update of The Code, not a wholesale revision. We are particularly concerned about possible blind spots or anachronisms that may have resulted from changes in technology or the profession since 1992.

You can help define what it means to be a good computing professional by contributing to the Code 2018 project.

We have completed three drafts of suggested updates to The Code, and we need your input as we begin to work on the final draft.

Get Involved! To review the drafts and to submit your comments, visit: <https://code2018.acm.org/discuss>

Members are encouraged to take an online survey about the specific principles of the code: <https://www.acm.org/code-2018-survey>



*Committee on
Professional Ethics*

<https://ethics.acm.org>



Moshe Y. Vardi

DOI:10.1145/3168007

Computer Professionals for Social Responsibility

I THINK OFTEN of *Ender's Game* these days. In this award-winning 1985 science-fiction novel by Orson Scott Card (based on a 1977 short story with the same title), Ender is being trained at Battle School, an institution designed to make young children into military commanders against an unspecified enemy (<http://bit.ly/2hYQMDF>). Ender's team engages in a series of computer-simulated battles, eventually destroying the enemy's planet, only to learn then that the battles were very real and a real planet has been destroyed.

I got involved in computing at age 16 because programming was fun. Later I discovered that developing algorithms was even more enjoyable. I found the combination of mathematical rigor and real-world applicability to be highly stimulating intellectually. The benefits of computing seemed intuitive to me then and now. I truly believe that computing yields tremendous societal benefits; for example, the life-saving potential of driverless cars is enormous!

Like Ender, however, I realized recently that computing is not a game—it is real—and it brings with it not only societal benefits, but also significant societal costs. Let me mention three examples. I have written previously on the automation's adverse impact on working-class people—an impact that has already had profound political consequences—with further such impact expected as driving gets automated (<http://bit.ly/2AdEv8A>). It has also become clear that “frictionless sharing” on social media has given rise to the fake-news phenomenon. It is now widely accepted that this had serious impact on both the 2016 U.K. Brexit referendum and the 2016 U.S. Presidential election. Finally, a 2017 paper in *Clinical Psychological Science*

attributes the recent rise in teen depression, suicide, and suicide attempts to the ascendance of the smartphone (<http://bit.ly/2zianG5>).

A dramatic drop in the public view of Tech, a term that I use to refer both to computing technology and the community that generates that technology, has accompanied the recent recognition of the adverse societal consequences of computing. This decline is well exemplified by Peggy Noonan, a *Wall Street Journal* columnist who wrote recently about trying to explain (dubiously, IMHO) why Americans own so many guns: “Because all of their personal and financial information got hacked in the latest breach, because our country's real overlords are in Silicon Valley and appear to be moral Martians who operate on some weird new postmodern ethical wavelength. And they'll be the ones programming the robots that'll soon take all the jobs!”

The question I'd like to pose to us in Tech is as follows: *We have created this technology; What is our social responsibility?* Of course, not all of us sit in Silicon Valley, and not all of us make product-deployment decisions. But much of the technology developed by high-tech corporations is based on academic research, by students educated in academic institutions. Whether you like it or not, if you are a computing professional, you are part of Tech!

Computer Professionals for Social Responsibility (CPSR), founded in the early 1980s, was an organization promoting the responsible use of computer technology. The triggering event was the Strategic Defense Initiative (SDI), a proposed missile-defense system intended to protect the U.S. from attack by ballistic strategic nuclear weapons. CPSR argued that we lack

the technology to develop software that would be reliable enough for the purpose of SDI. Later, CPSR expanded its scope to other tech-related issues. The organization was dissolved in 2013. (See Wikipedia <http://bit.ly/2zvZsZb>) With the benefit of hindsight, the issues that CPSR pursued in 1980s appear remarkably prescient today.

One could argue that CPSR is not needed any more; there are now numerous organizations and movements that are focused on various aspects of responsible use of technology. But our society is facing a plethora of new issues related to societal impact of technology, and we, the people who are creating the technology, lack a coherent voice. ACM is involved in many of these organizations and movements, by itself or with others, for example, ACM U.S. Public Policy Council, ACM Europe Policy Committee, the ACM Code of Professional Ethics, the Partnership on AI, and more. Yet, these efforts are dispersed and lack coordination.

I believe ACM must be more active in addressing social responsibility issues raised by computing technology. An effort that serves as a central organizing and leadership force within ACM would bring coherence to ACM's various activities in this sphere, and would establish ACM as a leading voice on this important topic. With great power comes great responsibility. Technology is now one of the most powerful forces shaping society, and we are responsible for it!

Follow me on Facebook, Google+, and Twitter. 

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. He is the former Editor-in-Chief of *Communications*.

Copyright held by author.

A Leap from Artificial to Intelligence

CARISSA SCHOENICK ET AL.'S article "Moving Beyond the Turing Test with the Allen AI Science Challenge" (Sept. 2017) got me thinking ... less about the article itself than about the many articles on artificial intelligence we see today. I am astonished that people who know what computers can do, and, especially, how they do it, still think we (humankind) will ever create a rational being, much less that the day is near.

I see no such sign. A program that can play winning chess or Go is not one. We all knew it would happen sooner or later. We are talking about a large but finite set of paths through a well-defined set. Clever algorithms? Sure. But such things are the work of engineers, not of the computer or its programs.

Siri is no more intelligent than a chess program. I was indeed surprised, the first time I tried it, by how my iPhone seemed to understand what I was saying, but it was illusory. Readers of *Communications* will have some notion of its basic components—something that parses sound waves from the microphone and something that looks up the resulting tokens—and if a sound token is within, say, 5% of the English word ... you know what must be happening. The code is clever, that is, cleverly designed, but just code.

Neither the chess program nor Siri has awareness or understanding. A game-playing program does not know what a "game" is, nor does it care if it wins or loses. Siri has no notion of what a "place" is or why anyone would want to go there.

By contrast, what we are doing—reading these words, asking maybe, "Hmm, what is intelligence?" is something no machine can do. We are considering ideas, asking, say, "Is this true?" and "Do I care?"

That which actually knows, cares, and chooses is the *spirit*, something every human being has. It is what distinguishes us from animals, and from computers. What makes us think we can create a being like ourselves? The

leap from artificial to intelligence could indeed be infinite.

Arthur Gardner, Scotts Summit, PA

Carissa Schoenick et al. (Sept. 2017) described an AI vs. eighth-grade science test competition run in 2015 by the Allen Institute for Artificial Intelligence. The top solution, as devised by Chaim Linhart, predicted the correctness of most answers through a combination of gradient-boosting models applied to a corpus. Schoenick et al. suggested that answering eighth-grade-level science-test questions would be more effective than Alan Turing's own historic approach as a test of machine intelligence. But could there be ways to extend the Turing Test even further, to, say, real-life scenarios, as in medicine?

Consider that electronic health records (EHRs) collected during a typical clinical-care scenario represent a largely untapped resource for studying diseases and associated co-occurring diseases at a national scale. EHRs are a rich data source, with thousands of data elements per patient, including structured elements and clinical notes, often spanning years. Because EHRs include sensitive personal information and are subject to confidentiality requirements, accessing such databases is a privilege research institutions grant only a few well-qualified medical experts.

All this motivated me to develop a simple computer program I call EM-RBots to generate a synthetic patient population of any size, including demographics, admissions, comorbidities, and laboratory values.² A synthetic patient has no confidentiality restrictions and thus can be used by anyone to practice machine learning algorithms. I was delighted to find one of my synthetic cohorts being used by other researchers to develop a novel neural network that performs better than the popular long short-term memory neural network.¹

In the EHR context, though a human physician can readily distinguish between synthetically generated and real live human patients, could a ma-

chine be given the intelligence to make such a determination on its own?

Health-care institutions increasingly must be able to identify authentically human patients' EHRs. In light of this trend, hackers might want to generate synthetic patient identities to exploit the data of real patients, hospitals, and insurance companies. For example, a hacker might want to falsely increase a particular hospital's congestive heart failure (CHF) 30-day readmission rate, a measure of quality. Medicare also uses it to evaluate U.S. hospitals—by posting into the hospital's database synthetic-patient identities associated with CHF readmissions.

Before synthetic patient identities become a public health problem, the legitimate EHR market might benefit from applying Turing Test-like techniques to ensure greater data reliability and diagnostic value. Any new techniques must thus consider patients' heterogeneity and are likely to have greater complexity than the Allen eighth-grade-science-test is able to grade.

References

1. Baytas, I., Xiao, C., Zhang, X., Wang, F., Jain, A., and Zhou, J. Patient subtyping via time-aware LSTM networks. In *Proceedings of the 23rd SIGKDD Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada, Aug. 13–17). ACM Press, New York, 2017, 65–74.
2. Kartoun, U. A methodology to generate virtual patient repositories. *arXiv Computing Research Repository*, 2016; <https://arxiv.org/ftp/arxiv/papers/1608/1608.00570.pdf>

Uri Kartoun, Cambridge MA

I Am Not a Number

Solon Barocas's and danah boyd's Viewpoint "Engaging the Ethics of Data Science in Practice" (Nov. 2017) did well to focus on ethics in data science, as data science is an increasingly important part of everyone's life. Data, ethics, and data scientists are not new, but today's computational power magnifies their scale and resulting social and economic influence. However, by focusing narrowly on data scientists, Barocas and boyd missed several important sides of the ethics story.

The disjoint they identified between

data scientists and researchers who critique data science is a strawman. For example, Cathy O’Neil, whose book *Weapons of Math Destruction* Barocas and boyd turned to for examples of unethical algorithms, is not just a researcher who happens to focus on data ethics, as they described her, but is herself a data scientist. Researchers like O’Neil would thus seem to be part of the solution to the problem Barocas and boyd identified.

Barocas and boyd also did not mention capitalism—the economic force behind much of today’s data science—with its behavioral prediction and manipulative advertising. What if the answer to unethical algorithms is to not create them in the first place? If an algorithm yields biased results based on, say, ethnicity, as Barocas and boyd mentioned, then surely the answer would be to not develop or use it to begin with; that is, do no harm. But such an approach also means the algorithm writer cannot sell the algorithm, an option Barocas and boyd ignored. They said data scientists try “to make machines learn something useful, valuable...” yet did not identify who might find it useful or what kind of value might be derived. Selling an algorithm that helps “wrongly incarcerate” people has financial value for the data scientist who wrote it but negative social value for, and does major financial harm to, those it might help put in jail. Data scientists must do what “maximizes the...models’ performance” but for whom and to what end? Often the answer is simply to earn a profit.

If one can learn data science, one can should be able to use it ethically. More than once Barocas and boyd mentioned how data scientists “struggle” with their work but expressed no empathy for those who have been hurt by algorithms. They did say data scientists choose “an acceptable error rate,” though for some scenarios there is no acceptable error rate.

If data scientists do not use data science ethically, as Barocas and boyd wrote and as O’Neil has shown, they are indeed doing it improperly. What Barocas and boyd failed to suggest is such data scientists should not be doing data science at all.

Nathaniel Poor, Cambridge, MA

Why Whole Foods for Amazon

Michael A. Cusumano’s Viewpoint “Amazon and Whole Foods: Follow the Strategy (and the Money)” (Oct. 2017) looked to identify a financial strategy for Amazon.com’s June 2017 acquisition of Whole Foods, which Amazon must have seen as strategically advantageous because it paid far more than Whole Food’s market valuation at the time. However, the column ignored the crucial potential for cross-subsidization to harm competition in the grocery industry. The trade press has since reported sales of high-margin Amazon products, including Echo devices, at Whole Foods, along with deals expected to come later (such as Whole Foods discounts for customers who also purchase Amazon Prime). Using revenue from such products and services to lower the price for commodity, low-margin products like groceries is a classic anti-competitive strategy with dubious legal or ethical basis.

Andrew Oram, Arlington, MA

Address the Slacker Before the Hacker

Esther Shein’s news story “Hacker-Proof Coding” (Aug. 2017) deserves a clarification and a warning about assumptions. Software developers should recognize that the techniques Shein explored will locate code faults but need not be solely manual, expensive, or tedious and thus error-prone. Contrary to the sources Shein quoted, much of the process of looking for faults can be automated. Automated diagnosis of errors can be done for approximately 20% of what it now costs in terms of programmer time and financial expenditure, yielding immediate and significant ROI.

Regarding assumptions, software developers should also be aware that the threat to the world’s software systems is not just from hackers but also from slackers often within their own midst. Too many software developers simply do not put enough effort into understanding the context in which their code will operate. For example, consider what Zachary Tatlock of the University of Washington said to Shein, “... as software that verifies the beam power has not become too high ... ” because software verifies only that some sensor output value is not too

high. Unfortunately, however, such verification software is not designed to verify first that the sensor is operating properly.

Moreover, software developers must learn to be more analytical about the specifications their software is being designed to obey. Making software conform to “some specification,” as Andrew Appel of Princeton University said to Shein, is foolish unless the specification is the right specification. In a large-system context, “right,” as Tatlock said, cannot be assured in advance. Developers must first create the code, then confirm it is consistent with all the other code with which it will eventually interoperate. Here, “confirm” means “consistent with the rules of logic, arithmetic, and semantics,” not just some code developer’s specification for only a piece of the ultimate system.

Jack Ring, Gilbert, AZ

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 0001-0782/18/1

Coming Next Month in **COMMUNICATIONS**

The Next Phase in the Digital Revolution: Platforms, Automation, Growth, Employment

Elements of the Theory of Dynamic Networks

Titus: Introducing Containers to the Netflix Cloud

Views from the Top

The Theory of Dynamic Networks

Plus the latest news about quantum encryption, the value of data, and the continuing education of software.

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:10.1145/3157073

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

The Big IDEA and the PD Pipeline

Former Computer Science Teachers Association executive director Mark R. Nelson discusses his work with the group to overcome core challenges to computer science education.



Mark R. Nelson
CSTA: Computer Science Education, Equity, and the Future Workforce
<http://bit.ly/2hGMSlt>
August 8, 2017

In 2015, when considering whether to apply for the position of executive director for the Computer Science Teachers Association (CSTA), I read the book *Stuck in the Shallow End* by Jane Margolis (<http://bit.ly/2zcEZeP>). Computer science (CS) remains one of the least diverse of the STEM disciplines, and Margolis' book opened with a compelling comparison between the racial divide in swimming and the divide we see today in CS. Understanding that teachers are critical to access, and how we teach can influence what field students might pursue, I saw that within CSTA there was opportunity to make a positive difference in the world.

Fast-forward to 2017, and thanks to Stephen Ibaraki, I had the opportunity to participate in the AI for Good Summit held in Geneva, Switzerland (<http://bit.ly/2h494Xi>). The Summit crystalized for me the scope, magnitude, and im-

portance of an organization like CSTA to change the world. The work of CSTA addresses core challenges for the future:

- ▶ How will we ensure all students are prepared to be citizens and workers in a different world than we knew in the past?

- ▶ How do we retrain adult workers, in this case education workers (teachers), to be ready for a rapidly changing world?

By working collaboratively, we *can* solve these challenges. Over the past two years, I saw the pathway to addressing these challenges within CSTA come together in a pair of initiatives: The Big IDEA and the PD Pipeline.

What Is the Big IDEA?

Building upon CSTA's historical interest in equity in CS, we began to change our language around equity. In 2015, our equity work focused mainly on a program with declining participation called "faces of computing," where students created videos to showcase some of the diversity within CS. In evaluating the program and looking at other successful initiatives, we recognized that focus on equity alone is insufficient to achieve equitable outcomes. Thus, we created the

Big IDEA as an umbrella for our current and future initiatives in this area.

IDEA stands for Inclusion, Diversity, Equity, and Access. While not the first group to utilize this acronym, the Big IDEA fit well within the needs of our community. By focusing almost exclusively in equity, we heard from many teachers that they did not feel included in the conversation because they identified themselves as "teachers of CS" rather than "CS teachers." That realization uncovered the diversity of perspectives that exist within our membership—which consists of nearly every teaching domain in K–12, and teachers who both come from and work with exceptionally diverse student populations. To achieve CS for All students, we realized the Big IDEA would be a more effective, holistic approach for focusing our efforts and building engagement in our community.

There is still much work to do. Take, for example, the challenges of accessibility in the context of disabilities. People with disabilities represent an estimated 15% of the world population. Designing for this group of learners can be challenging because disability is not a single construct. It can include physical challenges (related to areas such as vision, hearing, and mobility) as well as cognitive impairments (such as challenges related to reading, memory, and attention deficits).

Accessibility has implications for CS in education beyond the important category of children with disabilities. Accessibility challenges exist by race, gender, geography, community urbanization,

socioeconomic status, and many other dimensions. For example, in the U.S. there are entire states where not a single woman, African-American, or Latino takes the CS Advanced Placement (AP) exam each year. The Latino community, which represents 26% of K–12 students in the U.S., has been underserved by CS education largely due to access issues. We see similar challenges emerging for students in rural communities.

Limits to the Big IDEA, such as those highlighted here around access, become the basis for many forms of institutionalized inequity and can limit access to economic prosperity at multi-generational levels. As one speaker put it at the AI For Good Summit, the lack of diversity and inclusion in CS leads us to “solving mostly white male problems.” Over my two years at CSTA, I saw firsthand how teachers are critical for ensuring the Big IDEA becomes reality for children and society.

Organizations like CSTA have it within their mission and influence to make real change around the Big IDEA. Thanks to support and contributions from Google for Education, NCWIT, the Hispanic Heritage Foundation, and many volunteers, CSTA’s Big IDEA initiatives reached more than 3.5 million households over the past two years. In 2017, the Big IDEA in CS Education received a prestigious Silver Power of A Award recognizing CSTA and the contributions of both K–12 teachers and those who support them for “their extraordinary contributions and efforts to enrich lives, create a more competitive workforce, drive innovation, and make a better world.”

The PD Pipeline

In 2016, thanks to a grant from the Infosys Foundation USA, CSTA began development of the Professional Development (PD) Pipeline. The project emerged in response to a set of challenges from our members, which included:

- ▶ How do I know where to go to learn CS from quality providers?
- ▶ How do I track all of my PD and learning?
- ▶ How do I ensure all my PD contributes to certification or endorsement in my state?
- ▶ How do I connect with other teachers who took the same PD or came from the same background as me?
- ▶ How do I know what PD is the right

PD for me based on my current skill level and interests?

- ▶ How do chapters track and communicate PD taken or needed most by their members?

- ▶ How do I collect and share content and learning resources with other teachers like me?

Working with Degreed, last summer CSTA piloted an initial solution to these challenges via the PD Pipeline. The solution includes several elements. It begins with what will evolve into a developmental self-assessment to help teachers assess their knowledge of CS. That will be linked to developmental pathways to provide teachers with a roadmap to build their confidence and competence in teaching CS. We will provide digital badging and microcredentials to demonstrate teacher progress toward goals and outcomes. There is capacity to provide communities of practice and opportunities to share resources linked to a professional development (PD) experience, a developmental pathway, or other factors. The solution includes a digital portfolio, enabling a teacher to track all her PD in one place. Finally, nearly all these pieces link back to the K–12 CS Standards and Framework, or to a given state’s standards for CS, so a teacher can track progress against local requirements for licensure or endorsement.

The PD Pipeline recognizes that K–12 teachers of CS come from many different academic backgrounds and different exposure to CS. Our best estimate is that 1 in 9 teachers of CS have had a college-level course in CS, and fewer than 7% of our membership identify themselves as “CS teachers” first. Most are teachers of another subject who also teach CS. Thus, our 26,000 members include teachers of math, science, career and technical education, English, history, modern languages, art, music, physical education, special education, and many other domains.


We also learned that teachers came into CS with different areas of interest. We have teachers interested in robotics, cybersecurity, game design, artificial intelligence, data analytics, mobile applications, and many other domains of CS. We also recognized that some teachers want to become CS teachers, while others just want to integrate some CS in their classrooms.

The PD Pipeline is designed to provide a customized experience for an individual teacher with a goal to help her achieve the level of competence and confidence she desires or requires to be a successful teacher of CS in K–12. With technology changing quickly, and most CS teachers likely to come from the ranks of existing teachers, the PD Pipeline is an effort to address the question: How do we retrain the K–12 educational workforce (teachers) so they can prepare their students for the future?

As we look at the future displacement of jobs due to artificial intelligence, many individuals across a range of occupations and industries will need to be re-trained. The CSTA PD Pipeline initiative is one approach to solving such a large-scale workforce development problem.

Concluding Thoughts

The Big IDEA and the PD Pipeline are two examples of industry-changing initiatives under way at CSTA over the past two years. There is certainly much more that needs to be done in both these areas and others. As with any small non-profit, CSTA is dependent upon the support of organizations and individuals to fulfill its mission. While CSTA was founded by ACM more than a decade ago as part of its ongoing support to K–12 education, today more than ever, CSTA needs support from a broader range of stakeholders to successfully meet the challenges and needs of both the K–12 teacher community and the organizations and industries that depend on that community.

It was a distinct honor to serve as CSTA’s executive director for the past two years. By working collaboratively with many stakeholders, we made substantive progress on many fronts as we began the transition of CSTA from a “mom and pop” organization into a world-class professional association. Having accomplished many of the initial strategic goals set in 2015, it is now time for the next executive director to continue to build upon the transitional process that we began. I look forward to watching CSTA’s future success and encourage readers to support CSTA as the voice for K–12 teachers of CS. 

Mark R. Nelson was executive director of the Computer Science Teachers Association from June 2015 through July 2017.

© 2018 ACM 0001-0782/18/1 \$15.00

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

1966 A.J. Perlis
1967 Maurice Wilkes
1968 R.W. Hamming
1969 Marvin Minsky
1970 J.H. Wilkinson
1971 John McCarthy
1972 E.W. Dijkstra
1973 Charles Bachman
1974 Donald Knuth
1975 Allen Newell
1975 Herbert Simon
1976 Michael Rabin
1976 Dana Scott
1977 John Backus
1978 Robert Floyd
1979 Kenneth Iverson
1980 C.A.R Hoare
1981 Edgar Codd
1982 Stephen Cook
1983 Ken Thompson
1983 Dennis Ritchie
1984 Niklaus Wirth
1985 Richard Karp
1986 John Hopcroft
1986 Robert Tarjan
1987 John Cocke
1988 Ivan Sutherland
1989 William Kahan
1990 Fernando Corbató
1991 Robin Milner
1992 Butler Lampson
1993 Juris Hartmanis
1993 Richard Stearns
1994 Edward Feigenbaum
1994 Raj Reddy
1995 Manuel Blum
1996 Amir Pnueli
1997 Douglas Engelbart
1998 James Gray
1999 Frederick Brooks
2000 Andrew Yao
2001 Ole-Johan Dahl
2001 Kristen Nygaard
2002 Leonard Adleman
2002 Ronald Rivest
2002 Adi Shamir
2003 Alan Kay
2004 Vinton Cerf
2004 Robert Kahn
2005 Peter Naur
2006 Frances E. Allen
2007 Edmund Clarke
2007 E. Allen Emerson
2007 Joseph Sifakis
2008 Barbara Liskov
2009 Charles P. Thacker
2010 Leslie G. Valiant
2011 Judea Pearl
2012 Shafi Goldwasser
2012 Silvio Micali
2013 Leslie Lamport
2014 Michael Stonebraker
2015 Whitfield Diffie
2015 Martin Hellman
2016 Sir Tim Berners-Lee

ACM A.M. TURING AWARD NOMINATIONS SOLICITED

Nominations are invited for the 2017 ACM A.M. Turing Award. This is ACM's oldest and most prestigious award and is given to recognize contributions of a technical nature which are of lasting and major technical importance to the computing field. The award is accompanied by a prize of \$1,000,000. Financial support for the award is provided by Google Inc.

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

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

The deadline for nominations/endorsements is January 15, 2018.

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



Association for
Computing Machinery

Feeling Sounds, Hearing Sights

A new wave of sensory substitution devices work to assist people who are blind or deaf.

IN A 2016 video, Saqib Shaikh, a Microsoft Research software engineer, walks out of London's Clapham Station Underground stop, turns, and crosses a street, then stops suddenly when he hears an unexpected noise. Shaikh, who lost his sight when he was seven years old and walks with the aid of the standard white cane, reaches up and swipes the earpiece of his glasses.

The video then shifts to the view from his eyewear, a pair of smart glasses that capture high-quality still images and videos. That simple swipe instructed the glasses, an experimental prototype designed by a company called Pivothead, to snap a still photo. Microsoft software analyzed the picture, then translated the findings into auditory feedback. Through the smart glasses, which include a small speaker, Shaikh hears the results from an automated voice: "I think it's a man jumping in the air doing a trick on a skateboard."

The Pivothead smart glasses and Microsoft AI technology belong to a broader class of what have become known as sensory substitution technologies, apps and devices that collect visual, auditory, and in some cases



Chieko Asakawa, who is blind, uses the NavCog app, which she helped develop, to find her way on the campus of Carnegie Mellon University.

haptic stimuli, and feed the information to the user through another sensory channel. While the utility of these devices has long been debated in the vision- and hearing-impaired communities, recent advances suggest that sensory substitution technologies are finally starting to deliver on their promise.

A Rich History

The first sensory substitution devices originated long before computing machines and smartphone apps. The white cane widely used by people who are blind or have low vision alerts users to the presence of obstacles through tactile feedback. Similarly, Braille is a way of converting visual text to felt or tactile text. But a major technological shift has resulted from the efforts of neuroscientist Paul Bach-y-Rita, who developed a prototype device that converts video into tactile feedback.

Today, sensory substitution devices come in a variety of forms. The BrainPort, for example, translates visual information from a forehead-mounted camera into tactile feedback, delivering stimuli through 400 electrodes on a thumbprint-sized pad that users place on their tongue. Other aids include the vOICE, which translates camera scans of an environment into audible soundwaves, allowing users to hear obstacles they cannot see.

Although these devices use different approaches, they are capitalizing on the same general principle. “Most of the hard computing work is being done in the brain,” explains experimental psychologist Michael Proulx of the University of Bath. “What we’re doing is relying on the brain’s ability to take information in any sensory format and make sense of it independent of that format.”

Practical Uses

Neuroscientist David Eagleman and his colleagues at Neosensory, a Silicon Valley startup, are developing a new device, the Buzz, that translates ambient sounds such as sirens or smoke alarms into distinct patterns of vibrations that pulse through and across the device’s eight motors. A smartphone microphone picks up the sound, then passes it through an app that mimics the role of the inner ear.

“What we’re doing is relying on the brain’s ability to take information in any sensory format and make sense of it independent of that format.”

One algorithm separates the sound into its component frequencies (as our own ears do) while others cancel out unrelated noise, such as the hum of an air conditioner. The app then transforms this change in frequencies over time into a pattern of vibrations that alters every 20 milliseconds, rolling through or pulsing on the Buzz. “With a siren, you feel it going back and forth on your wrist because there are different frequencies involved,” Eagleman explains. “A knock on the door is easy. You feel the knock on your wrist.”

The Buzz and its predecessor, a more robust wearable vest, are also designed to be affordable: the projected price of the wrist-worn version should be less than \$400. Cost is a major concern because sensory substitution devices are not reimbursed through health insurance in the U.S., and studies have found that people with disabilities often have lower rates of employment and income, and may not be able to afford technologies like the BrainPort, which retails for \$10,000. “For people with sensory disabilities, none of these technologies are covered” by insurance, says Deborah Cook, Washington Assistive Technology Act Program technical advisor and director of the Older Blind Independent Living Program at the University of Washington. “You can get a wheelchair paid for, but you can’t get a new visual or auditory device reimbursed.”

Cook also argues that many sensory substitution devices are too focused on navigation. But IBM computer sci-

entist Chieko Asakawa believes there is still an unmet need in this space, and that such technologies have the potential to allow people who are blind to explore unfamiliar areas such as schools, train stations, airports, and more. “It’s not fun if I go to a shopping mall by myself, for example,” says Asakawa, who lost her sight at age 14. “If there are many people in the mall, it’s very difficult to move around with the white cane.”

Asakawa and her collaborators at IBM Tokyo and Carnegie Mellon University have developed a new system, NavCog, that deploys bluetooth beacons throughout interior spaces such as academic buildings and, in one case, a public shopping mall. The beacons connect to a smartphone app, which guides the user via voice assistance. “In the mall,” she explains, “I can find out which shop is next to me while I’m walking, such as a coffee shop on the left or a sushi restaurant on the right. That’s useful information.”

Siri’s Shortcomings

Devices that help individuals enhance their productivity in the workplace are also critical. Computer scientist Matt Huenerfauth of the Linguistic and Assistive Technologies Laboratory at the Rochester Institute of Technology (RIT) is working with researchers from the National Technical Institute for the Deaf (NTID) to see if Automatic Speech Recognition (ASR) technology of the sort that powers Siri, Alexa, and Cortana could be used to generate captions in real time during meetings. Often, people who are deaf or hard of hearing either skip business meetings and wait for summaries from other attendees, or sit through the conferences and miss numerous side conversations. However, ASR technology is imperfect, and a real-time captioning system with errors in the text can be confusing. Huenerfauth’s team is investigating whether highlighting words that the ASR is not confident it recognized correctly—using italicized fonts—will help users understand which fragments of a transcript they can trust.

Computer scientist Raja Kushalnagar of Gallaudet University, along with colleagues from the University of

Rochester, Carnegie Mellon University, and the University of Michigan, has pursued a slightly different approach to the same problem. Instead of attempting to build an automated system, Kushalnagar and his collaborators are developing a crowdsourcing technology that transcribes lectures or business meetings into reliable captions with only a five-second delay. Professional stenographers can cost \$100 per hour, so they are not a viable option for most users. The group's system recruits multiple untrained, lower-paid workers to remotely transcribe brief portions of a given meeting or lecture in real time. Their technology, Legion Scribe, then applies an algorithm that merges the three fragmented transcriptions into one sensible, readable text for people who are deaf or hard of hearing.

Feedback Loop

The incorporation of remote human assistants is not entirely unique in the sensory substitution field. The University of Washington's Cook touts a new service, Aira, which connects blind users with remote human agents through a pair of smart glasses. The agents, who act as assistants, see through the glasses, but they can also tap into GPS data and other information. If the user enters a Target store, for example, Aira software immediately brings up a map of that store's layout to help the agent guide the individual. Individuals can engage the service for a variety of tasks; one Aira user tapped into the service at a family funeral because he didn't want to bother relatives and ask for their help.

"Helping people identify objects and texts is good," says Aira co-founder and CEO Suman Kanuganti, "but we want to help them gain experiences in life, doing things faster and more efficiently."

Kanuganti says there is an additional advantage to keeping human agents in the loop. Since the service launched in early 2017, roughly 3,600 hours' worth of interactions between agents and users have been stored in the cloud. Aira has begun training machine learning algorithms on all that video, audio, GPS, and other data to develop automatic tools that provide some of the same functionality that re-

mote human agents deliver now, such as automatic facial recognition and reading.

While these technical advances are welcome, experimental psychologist Proulx, who tests and compares several sensory substitution devices, including the BrainPort, in his laboratory, believes the devices will also improve as researchers learn more about how the brain processes sensory information flowing in through different channels.

"There have been findings showing that as people learn to use these devices, the brain is starting to use the visual parts of the cortex to process the information it's receiving through sound or touch," Proulx says. "Over the next 10 years is probably when we'll see a nice feedback loop between improvements in the technology and greater findings about the human user and how the brain is actually functioning. As that comes together, I'm really excited to see what sort of advances we're able to make." ■

Further Reading

Lasecki, W. Kushalnagar, R., and Bigham, J. Legion Scribe: Real-time captioning by non-experts, *Proceedings of the 16th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS, 2014.*

Maidenbaum, S., Abboud, S., and Amedi, A. Sensory Substitution: Closing the gap between basic research and widespread practical visual rehabilitation, *Neuroscience & Biobehavioral Reviews, 41, 2014.*

S. Daisuke, O. Uran, N. Kakuya, T. Hironobu, K. Kris, and A. Chieko. NavCog3: An evaluation of a smartphone-based blind indoor navigation assistant with semantic features in a large-scale environment, *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS, 2017.*

Sadeghi, S., Minor, L., Cullen, K. Neural correlates of sensory substitution in vestibular pathways following complete vestibular loss, *The Journal of Neuroscience, Vol. 32, Issue 42, October 17, 2012.*

How new technology helps blind people explore the world; <http://bit.ly/1LRd2UD>

Gregory Mone is a Boston-based science writer and the author, with Bill Nye, of *Jack and the Geniuses: At the Bottom of the World.*

© 2018 ACM 0001-0782/18/1 \$15.00

ACM Member News

PURSUING COMBINATORICS WITH A MATH ARSENAL



Noga Alon, Baumritter Professor of Mathematics and Computer Science at Tel Aviv University,

Israel, earned all of his degrees in mathematics, in Israel. He completed his undergraduate studies at Technion-Israel Institute of Technology in Haifa, his master's degree at Tel Aviv University, and he received his Ph.D. from the Hebrew University of Jerusalem.

An ACM Fellow, Alon says his serious interest in computer science did not begin until he started post-doctoral studies at the Massachusetts Institute of Technology.

Lately, Alon notes, "I have been interested in understanding which types of combinatorial proofs lead to efficient procedures for solving corresponding algorithmic problems." The development of streaming algorithms, analyzing huge quantities of data on the fly to understand their statistical properties, has also received his attention.

His other research interests are mainly in combinatorics (a branch of math dealing with combinations of objects belonging to a finite set in accordance with certain constraints) and graph theory, and their applications in theoretical computer science.

While Alon, 61, believes his best work is still ahead of him, there are a number of challenges he would like to try to address, such as well-known problems in mathematics, combinatorics, and theoretical computer science. More specifically, "I would like to develop techniques to tackle problems in understanding the power of proof techniques and computation," he adds.

In 2018, Alon will join the faculty of Princeton University as a professor of mathematics and applied and computational mathematics, while maintaining his affiliation to Tel Aviv University.

—John Delaney

Smartphone Science

A new generation of portable scientific instruments is taking shape, thanks to mobile processors and innovative data-gathering techniques.

IN HIS 1998 book *The Invisible Computer*, user experience pioneer Don Norman predicted that general-purpose computers would one day give way to “information appliances”: compact tools designed for specialized computational purposes, capable of communicating across a wide range of platforms.

Thanks to recent innovations in mobile processors, sensors, and image capture tools, Norman’s vision now looks remarkably prescient—nowhere more so than in the worlds of science and medicine, where advances in mobile technology are starting to yield lightweight, low-cost scientific instruments that promise to democratize access to a wide range of powerful diagnostic techniques.

Over the past few years, researchers have created smartphone-powered tools capable of taking blood samples, identifying viruses, analyzing water safety, and even sequencing DNA. Many of these instruments are already allowing scientists and medical professionals to gather and analyze data in ways that, until recently, would have required bulky and expensive lab-based equipment.

“Over the past decade we’ve seen enormous advances in the capabilities of smartphones,” says Aydogan Ozcan, a professor at the University of California, Los Angeles (UCLA) who led an international team that created a smartphone-based microscope capable of analyzing DNA sequences in tissue samples with a level of fidelity comparable to that of a conventional microscope.

When Ozcan first started exploring the idea of a mobile phone-based microscope in 2006, his team created a specialized camera attachment capable of registering a red blood cell—about one-tenth the width of a hu-



Say Ah: An off-site doctor examines the throat of a child via smartphone during a digital medical consultation.

man hair. Today, thanks to advances in smartphone lenses and processor power, Ozcan’s team has been able to develop an Android-based system capable of registering a DNA molecule just two nanometers wide (about 500 times smaller than a red blood cell).

The device works by taking multiple different images of a tissue sample, which it then feeds into an algorithm that analyzes the images using fluorescent dyes to sequence the DNA bases and look for genetic mutations inside the tumor. The researchers claim the device can identify even miniscule signs of cancer among a large group of cells.

The team prototyped the device using a three-dimensional printer, and estimates the smartphone-based microscopes could ultimately be produced for less than \$500 per unit—a fraction of the \$10,000 to \$50,000 that a traditional high-end microscope would typically cost.

At that price point, the device could bring molecular diagnostics—

which are typically performed in centralized laboratories at major hospitals—within reach of remote doctors’ offices, and eventually consumers. The microscope could also have a wide range of other applications for biotechnology and nanotechnology research in remote environments.

“This transformation is really helping us become competitive with things that are normally done in advanced labs and professional settings,” says Ozcan. “This is about the democratization of measurement, science, and diagnostics.”

At Washington State University, a team of researchers led by Lei Li has created a smartphone-based spectrometer capable of analyzing blood samples for a specific biomarker known as human interleukin-6, commonly associated with lung, prostate, liver, and breast cancers. The device could also allow users to conduct spectrophotometric analysis for a variety of purposes including food safety, water quality, and other types of environmental analysis.

Spectrometers measure the light spectrum to gauge the amount and type of chemicals present in a given sample of material. But even today's powerful smartphone cameras were never intended for this kind of intensive optical analysis.

"The camera was never designed for sampling," says Li. "It was designed for taking pictures at a distance." To compensate for the limitations of tiny lenses, Li and his team fashioned a special prism array capable of dividing incoming light into waves for processing. Most of the technical challenges came in trying to fabricate the prism array itself; the software application was comparatively easy to develop using standard image processing routines.

In the past, large hospitals would have expected to pay up to \$500,000 for a top-of-the-line spectrometer; a more basic model might run as little as \$10,000. While Li's device won't fully supplant those kinds of machines, it could make spectrometer-based analysis available to smaller hospitals and clinics in rural areas or developing countries, thus bringing the technique within reach of patients who might otherwise never have access to this kind of diagnostic capability.

At the University of Washington, a team of doctors and engineers partnered to create a smartphone-based device called BiliCam that allows doctors and new parents to identify signs of jaundice, a common affliction of newborn babies.

The system consists of an app that invokes a standard smartphone camera and flash, along with a printed card used to calibrate color with a range of different lighting conditions and skin tones. By taking a photo of the baby's skin in proximity to the card, a parent or healthcare provider can then use the app to submit the photo to a cloud-based system that analyzes the data using machine-learning algorithms, then generates a report on the baby's bilirubin levels that it can send back almost instantly to a parent's smartphone.

Beyond image-processing applications, the sheer computing power of smartphones—the Samsung S7 handset now boasts far more processing power than a supercomputer from the mid-1990s—opens the door to a wide range of other pocket-sized scientific instruments.

At the University of Washington, a smartphone-based device allows doctors and new parents to identify signs of jaundice, an affliction common to newborn babies.

In the U.K., a company called Oxford Nanopore Technologies builds compact devices known as nanopore sequencers for genetic analysis; they utilize a highly cost-effective approach that allows a single molecule of DNA or RNA to be analyzed via electrical conductivity (as opposed to more resource-intensive solid-state methods). The method makes genetic sequencing more readily available to researchers for a wide range of purposes: identifying pathogens in food or water, monitoring environmental conditions for climate science, or even sequencing the human genome. Nanopore sequencing also holds promise for a wide range of other scientific applications including plant research, population genomics, and microbiology.

In 2014, the company released MinION, a nanopore sequencer that weighs under 100 grams and plugs into a PC or laptop via a USB cable. With prices starting at \$1,000, the device is making genetic sequencing widely available to a range of researchers who might not otherwise have access to sequencing technology. It has already been used for surveillance of Zika in Brazil and Ebola in Guinea, and has even found its way onto the International Space Station.

Oxford Nanopore is now taking this work a step further with its as-yet-unreleased SmidgION, a sequencer attachment that snaps onto a smartphone running specialized software.

The proliferation of mobile processing power will not only make high-powered scientific instruments more readily available to researchers,

Milestones

Labarta Recognized with ACM-IEEE Kennedy Award

ACM and IEEE recently named Jesús Labarta of the Barcelona Supercomputing Center and Universitat Politècnica de Catalunya as the recipient of the 2017 ACM-IEEE CS Ken Kennedy Award.

Labarta was recognized for his seminal contributions to programming models and performance analysis tools for high performance computing.

Throughout his career, Labarta has developed tools for scientists and engineers working in parallel programming. In the programming models area, he made fundamental contributions to the concept of asynchronous task-based models and intelligent runtime systems. In the performance tools area, Labarta's team develops and distributes Open Source Barcelona Supercomputer Center tools designed to analyze an application's behavior and identify issues that may impact performance.

Labarta is director of the Computer Science Department at the Barcelona Supercomputing Center and a professor of Computer Architecture at the Universitat Politècnica de Catalunya. From 1996 to 2004 he served as director of the European Center of Parallelism of Barcelona. He has been involved in research and cooperation with many leading companies on HPC-related topics. Currently, he is the leader of the Performance Optimization and Productivity EU Center of Excellence, where more than 100 users (both academic and SMEs) from a very wide range of application sectors receive performance assessments and suggestions for code refactoring efforts.

ACM and IEEE co-sponsor the Kennedy Award, which was established in 2009 to recognize substantial contributions to programmability and productivity in computing and significant community service or mentoring contributions. The Award carries a \$5,000 honorarium endowed by the SC Conference Steering Committee.

but should also eventually yield simpler diagnostic tools within reach of everyday consumers.

Dr. David Bello of Florida-based Orlando Health has developed a smartphone-based stethoscope that retails for less than \$50. Using a small sensor device that connects to a smartphone via its headphone jack, the system can capture a user's heartbeat and project a visualization of it onscreen—potentially offering consumers an easy-to-use tool for monitoring their cardiac health.

Since 2013, a team of researchers at Cornell University have been developing a portable tool that analyzes blood samples for nutritional deficiencies, with results sent to a mobile phone in about 10 minutes. The developers, who have moved the technology to global health startup VitaScan, hope the device could one day make this kind of testing available throughout the developing world.

“This is a severe, global, and frequently overlooked problem,” explains Li Jiang, VitaScan's CEO. Testing for vitamins and micronutrients poses far more complex diagnostic challenges than a simple yes/no result (like a pregnancy test). In developing countries, limited access to laboratory testing means nutritional problems too often go undiagnosed.

“The high cost and difficulty of assessing malnutrition has made it such that most people are not aware of it until they get sick,” he says.

The VitaScan system relies on a small standalone diagnostic unit that

Since 2013, Cornell University researchers have been working on a portable tool that analyzes blood samples for nutritional deficiencies.

can analyze a blood sample collected via a finger prick. The unit then communicates wirelessly over the Internet with a server-based application that can then transmit results to a user's smartphone. While in principle such a device could connect directly to a smartphone, FDA regulations preclude putting smartphones in close contact with biological fluids for safety reasons.

The company is planning to release its product initially to doctors' offices and clinics, before expanding into the consumer market, where Li sees an opportunity for VitaScan to take its place within the larger ecosystem of personal health and nutrition applications.

“It's more than just processing power,” he says. “The ubiquity and personalized nature of smartphones have

helped drive the growth of personalized nutrition.”

For a health-conscious consumer, a tool like VitaScan could form one component of a self-administered health and wellness regimen, with the user's smartphone corralling and synthesizing input from multiple sources like a diet tracker, fitness monitor, or any number of other information appliances.

With so many of these smartphone-based scientific instruments coming to market, we may be witnessing a maturation of the underlying technologies, like processors, optics, and fabrication methods. As is so often the case in computing, last year's innovations become this year's infrastructure.

Looking ahead, the next wave of experimentation may have less to do with the instruments themselves and more to do with finding the right pathways to market.

“There are still lots of improvements to come to the core technologies, but the next wave of real innovation will come with applications,” says Ozcan. “Now it's time to harness the power of what we've achieved.”

Further Reading

De Greef, L., Goel, M., Seo, M., Larson, E., Stout, J., Taylor, J., and Patel, S. *BiliCam*

Using mobile phones to monitor newborn jaundice. *UBICOMP '14 Adjunct*, Sept. 13–17, 2014.

Kühnemund, M., Wei, Q., Darai, E., Wang, Y., Hernández-Neuta, I., Yang, Z., Tseng, D., Ahlfjord, A., Mathot, L., Sjöblom, T., Ozcan, A., and Nilsson, M.

Targeted DNA sequencing and in situ mutation analysis using mobile phone microscopy. *Nature Communications* 8, Article number: 13913 (2017). doi:10.1038/ncomms13913

Loose, M.

The potential impact of nanopore sequencing on human genetics. *Human Molecular Genetics*, ddx287, doi:10.1093/hmg/ddx287

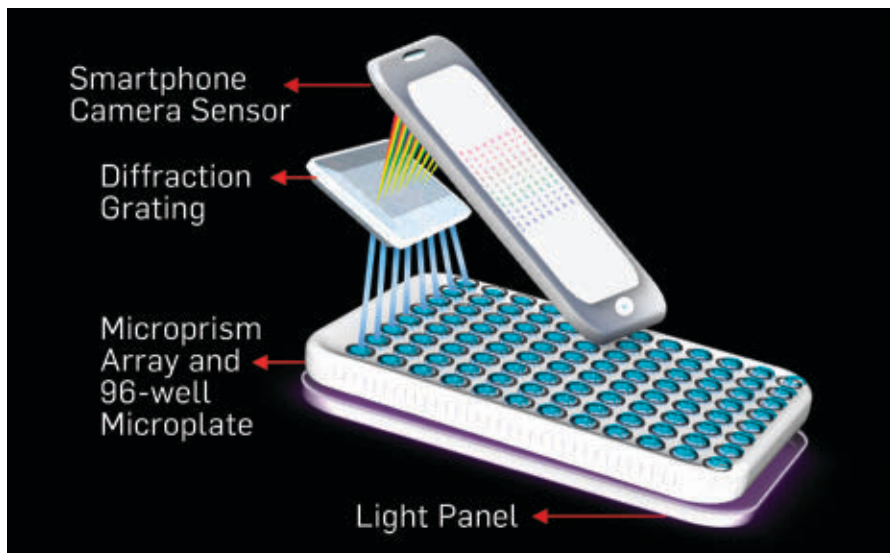
Norman, D.

The Invisible Computer. MIT Press, Cambridge, MA. 1998.

Wang, L., Chang, Y., Sun, R., and Li, L.

A multichannel smartphone optical biosensor for high-throughput point-of-care diagnostics. *Biosensors and Bioelectronics*, Volume 87, 15 January 2017, pp. 686–692. doi:10.1016/j.bios.2016.09.021

Alex Wright is a writer and researcher based in Brooklyn, NY.



Elements of Washington State University's smartphone-based spectrometer.

The New Jobs

As automation takes on more and more tasks, what will human workers do?

RARELY DOES A day go by without more news predicting the end of work. After all, autonomous vehicles are all but certain to replace truckers and taxi drivers in the coming decades, and robots have already taken over many jobs in factories and warehouses, and will continue to expand their reach beyond heavy industry as they become smarter and ever more affordable.

Perhaps most frighteningly, even professional services no longer seem safe from the encroachment of increasingly sophisticated artificial intelligence (AI). Law firms, for example, employ electronic-discovery software, which uses natural language processing to sift through reams of documents faster and more cheaply than the entry-level lawyers who used to do this tedious work. Deep-learning image recognition tools can flag and classify worrisome tumors in digital scans as well as, or better than, experienced radiologists. Websites like Wealthfront and Betterment, which algorithmically optimize investment portfolios, are giving financial advisors a run for their money.

No one doubts that automation is replacing many jobs, but will advances in technology also create plenty of new jobs? If so, what kind(s)?

These questions have taken on new urgency in the past year, as the startling rise of populism and the election of U.S. President Donald J. Trump have awakened previously complacent white-collar workers to the plight of the large numbers of Americans left behind by automation and other changes in the U.S. economy over the past several decades.

For example, according to a widely reported recent study by economists Daron Acemoglu and Pascual Restrepo, industrial robots alone have eliminated as many as 670,000 U.S. jobs between 1990 and 2007. As some



displaced workers stay unemployed, while others must settle for significantly lower-paying work in the retail and service sectors, society pays the price in the form of lower consumer spending, higher crime rates, and a lower tax base.

“Two-thirds of people in this country don’t have a four-year degree,” points out Moshe Vardi, senior editor of *Communications* and a professor of computational engineering at Rice University who writes and lectures on the impact of automation on work. “We have to make sure we have an economy that creates jobs for them—and not only jobs for the people who have a Ph.D. in computer science.”

What kinds of jobs will the tech-driven economy create? Although no one can peer into the future, experts make educated guesses by looking

to the effects of automation in the past, which have followed a pattern of initial disruptions followed by a period of adjustment that culminated in long-term gains.

A classic example is the Industrial Revolution, the massively disruptive transformation of the economy beginning in the 18th century. Industrialization created decades of problems, from changes in job structure and skills demanded to horrendous working conditions for those who did get the new factory jobs; eventually, though, industrialized economies adjusted, with enormous increases in prosperity. A major expansion of education and the rise of labor unions made sure that workers shared in these gains.

Similarly, in the first part of the 20th century, in what’s been called the

second Industrial Revolution, technological improvements increased job turnover but lowered unemployment, according to a study published last year in the *Review of Economics and Statistics* (<http://bit.ly/2wv9Jmj>).

The big question these days, though, is whether this time will be different. Lacking a crystal ball, scholars are again looking to history—analyzing data from recent years, as the rate of automation has accelerated.

That is the approach used by economists like Anna Salomons of the Utrecht University School of Economics and David Autor of the Massachusetts Institute of Technology, who analyzed 35 years' worth of data for 19 industrialized countries to study the relationship between technology-fueled productivity growth and employment. Across the board, the researchers found that although employment certainly falls in industries with rising productivity, overall employment actually goes up somewhat. "Job losses are occurring in some places," Salomons says, "but they are overcompensated by gains in others."

Those countervailing job gains come from two sources, according to Salomons, who holds her school's Chair of Applied Econometrics.

The first is higher incomes, which tend to increase consumption of products and services. Put simply, "we keep buying new stuff, and that keeps producing new jobs," she says. (In Silicon Valley, for example, the tremendous growth in the tech sector has fueled job growth in real estate, the restaurant industry, and personal services like massages.)

The other source of new jobs from automation is what economists call inter-industry demand linkages. "As one industry becomes more productive," Salomons explains, "other industries that use that [industry's products] as an input will also demand more, and as a result you'll also get more labor demand."

Whereas many of us see job titles—factory worker, doctor, engineer—experts on technology's economic impact find it more helpful to think in terms of *tasks* as the basic unit of work. This more granular view is use-

ful because most jobs don't consist of just one task. "It's a variety of skills and responsibilities," points out computer scientist and entrepreneur Jerry Kaplan, who teaches a Stanford University course on the social and economic impact of artificial intelligence. "So the way to think of it is which tasks to automate."

It is relatively easy to automate routine tasks, since these can be codified in an algorithm; non-routine tasks, on the other hand, have been resistant to even the most sophisticated AI, requiring complex capabilities such as analytical skills, creativity, and compassion. The more of a job's tasks that are non-routine, therefore, the more technology can serve as a complement, rather than a substitute, for the worker—helping them to be more effective, instead of threatening their livelihood.

That reasoning gives cause for optimism about the future of work. This is especially true if you look at it from the perspective of AI developers, for whom augmenting human capabilities makes economic sense, suggests Guruduth Banavar, chief technical

Technology

Sharing Secrets (Without Giving Them Away)

Secure multiparty computation enables two or more parties to share sensitive information and process it for a common purpose, without giving up privacy. A fairly simple example is online secret balloting, but now, advanced pilot projects in medicine and genetics are starting up.

Ronald Cramer, cryptographer at the Dutch Centrum voor Wiskunde en Informatica (CWI) research center, has received a €2.50 million (\$2.95 million) grant from the European Research Council to further develop (among other things) secure multiparty computation. He is now collaborating on several use-cases with electronics giant Philips, technical research hub TNO, and the University of Amsterdam.

In real-world applications, secure multiparty computation involves large amounts of data and multiple rounds of communication, making heavy demands on computer speed

and bandwidth. Improving the runtime of an algorithm by even 15% can be critical. "This is programming on a knife's edge," says Cramer.

The simplest textbook example of secure multiparty computing consists of three people who want to vote yes/no (1 or 0) on some topic, without revealing their individual votes, and without a trusted authority to count the votes. This can be accomplished by having each voter choose three random numbers. To every other voter, he then sends the sum (modulo some public number N) of his vote, plus two of the three random numbers (a different pair for each receiver). Every voter then adds the numbers he gets from the others, and makes this number public. The sum of these three numbers, modulo N , is the number of yes-votes.

This kind of privacy protection does not depend on encryption, but on general properties of random numbers,

so it cannot be broken (assuming certain limits as to how much the parties involved can eavesdrop and collude). For more than three parties and more than simply counting, the algorithms get much more complex, and can require many more rounds of communication.

The pilot project Cramer is working on with Philips Research and TNO aims to optimize the workflow in Dutch hospitals. In this project, every hospital employee and patient, and even the equipment, gets a locator tag, reporting its exact position every few seconds.

Analyzing this tracking data can reveal bottlenecks in the workflow, but obviously there is a privacy problem. To avoid that, the data is split between two parties: staff data is only accessible to the workers council, while only the hospital has access to patient data. In this way, secure multiparty computation can protect everybody's privacy,

while still distilling useful results on workflow.

Cramer and TNO's Thijs Veugen are working with Emiliano Mancini at the University of Amsterdam to apply secure multiparty computation to the improvement of the treatment of HIV patients. Every patient's virus has different mutations, with different levels of resistance to various medicines. By combining data from multiple patient databases, much can be learned about optimizing the treatment for individual patients.

Safeguarding the privacy of the patients using secure multiparty computation makes it a lot easier for institutes around the world to share data for joint research projects. Observes Veugen, "Without secure multiparty computation, you could derive who slept with whom."

—Arnout Jaspers is a freelance science writer based in Leiden, the Netherlands.

officer of Viome, a startup that develops AI for wellness using information from people's microbiomes. "The field of AI is going to take the route of complementing people much more than replicating people," says Banavar, who previously headed research and development for IBM's Watson.

He points out that although humans (like machines) have both strengths and weaknesses, for now "unfortunately we are making people do things that humans are bad at," Banavar says. Think of a radiologist sitting in a dimly lit room looking at hundreds of images each day, searching for anomalies. "It's a boring job, you can be error-prone, and you can get tired." Machines doing the same task make fewer errors—without getting bored or tired. A well-trained AI system, therefore, enables the radiologist to play to uniquely human strengths, such as the ability to make a differential diagnosis.

Differential diagnosis, Banavar explains, requires the doctor to mentally connect images to the patient's history, as well as to textbook knowledge and to the latest scientific findings. "You have to connect all those dots to make hypotheses about what is possibly going on, and then you have to analyze enough observations to decide among those possible hypotheses, and test some of them, then proceed based on what the outcomes of those experiments are."

The doctor also has to interpret the results for each patient and explain the patient's options. "That's a social skill, and those social skills are extremely complicated for machines to learn," Banavar says. Similarly, jobs in management require understanding a complex mix of human and social factors. "I think it's going to be a very long time before we can get machines to do those things," Banavar says. "For social skills, I think the demand will increase over time."

Recent research by Harvard economist David Deming on the growing importance of social skills in the labor market supports Banavar's prediction. "Between 1980 and 2012, social skill-intensive occupations grew by 11.8 percentage points as a share of all jobs in the U.S. economy," Deming writes, pointing to rapid growth in de-

"The field of AI is going to take the route of complementing people much more than replicating people."

mand for managers, teachers, nurses and therapists, and doctors and lawyers. These socially demanding jobs have also seen wage growth during this period, Deming adds, noting that jobs that require high levels of both cognitive and social skills have fared particularly well.

"Computer programmer" might well be one of those cognitive-and-social jobs of tomorrow, especially as programmers continue to boost their efficiency by working with others—seeking and giving answers on Stack Overflow, for example. Busting the myth of the lone coder will encourage more people prepare for tech jobs, believes Caleb Fristoe, the founder of a Tennessee nonprofit called CodeTN, which partners with Knoxville-area schools to teach basic software development skills to low-income students, such as the children of laid-off factory workers. These days, Fristoe says, coding doesn't require extraordinary smarts, or getting a job at the likes of Google.

Instead, CodeTN's goal is to prepare working-class children for generalist coding jobs consisting of unglamorous tasks like managing a firm's login page or setting up a customer database—work that requires only modest aptitude and little formal education.

"New frameworks are lowering the barrier to entry," Fristoe says; that's a far cry from the days when you had to learn the syntax of several programming languages to build useful software. "Rather than typing these seven lines of code to get a menu to pop down, you just download the framework from a code base that al-

lows you to do that in a simpler way," he explains. "Frameworks are taking the hard work that developers prided themselves on out of the equation."

In other words, programming itself has become more automated, potentially opening up job opportunities for more people within a widening circle of employers.

Not all experts are sanguine about the future of jobs, however. "There's a friction in economic systems that economists aren't thinking about," says Vardi, referring to the challenges of matching workers to available jobs. Training hundreds of programmers or plumbers, for example, doesn't guarantee that the local economy can absorb that many workers—nor is it simple for people to move to where the jobs actually are. Just think of Silicon Valley, which has created so many jobs outside the tech sector: the Bay Area is also, not coincidentally, one of the most expensive places to live. And just because there's high demand for nurses doesn't mean men who've lost their jobs in manufacturing will even want to become nurses, let alone retrain and try to get hired.

"You can argue that this is culture, this is bias, or whatever, but they're not moving to those jobs," Vardi says. ■

Further Reading

Deming, D.

The Growing Importance of Social Skills in the Labor Market, working paper, May 24, 2017
https://scholar.harvard.edu/files/ddeming/files/deming_socialskills_aug16.pdf

Kaplan, J.

Humans Need Not Apply: A Guide to Wealth and Work in the Age of Artificial Intelligence
Yale University Press, 2015

Autor, D. and Salomons, A.

Robocalypse Now—Does Productivity Growth Threaten Employment? (ECB Sintra Forum on Central Banking Conference Paper, June 19, 2017)
https://www.ecbforum.eu/uploads/originals/2017/speakers/papers/D_Autor_A_Salomons_Does_productivity_growth_threaten_employment_Final_Draft_20170619.pdf

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

© 2018 ACM 0001-0782/18/1 \$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:10.1145/3163905

Michael A. Cusumano

Technology Strategy and Management

The Sharing Economy Meets Reality

Assessing the uncertainties of the business models driving the sharing economy.

THE SHARING ECONOMY seems big and growing fast. In 2015, PricewaterhouseCoopers estimated that, from \$15 billion in global revenue in 2014, car and room sharing, crowd-funding, personal services, and video and audio streaming would reach \$335 billion by 2025.¹ These numbers extrapolate from the growth rates of Uber and Airbnb. Are these growth rates sustainable and is the economic activity really new? It is time to ask these questions.

The sharing economy depends on digital platforms that enable people who do not know each other to access underutilized assets (see my previous column “How Established Firms Must Compete in the Sharing Economy,” *Communications*, Jan. 2015). We often associate these ventures with “peer-to-peer” (P2P) technology, like Napster in the late 1990s. However, the recent estimates include two very different business models. Sometimes *individuals* own and share the assets, like extra

rooms, household tools, music files or movie videos, their own free time and skills, and even their pets. Prominent examples include Airbnb, HomeStay, Uber, Lyft, Rover, TrustedHousesitters, and Taskrabbit (recently purchased by Ikea). In other cases, *companies* own and lend out the assets, like automobiles, bicycles, scooters, or even basketballs and

The confluence of information technology, entrepreneurship, and venture capital has led to the creation of many new ventures.

umbrellas. Prime examples here are Zipcar, now owned by Avis, and Car2Go, owned by Daimler-Benz. Asset sharing that is P2P seems new but B2C (business-to-consumer) sharing is another version of the traditional rental business, and may be substituting for that activity. Even Airbnb may not be totally new economic activity, as it substitutes for traditional bed-and-breakfast revenue or hotel and vacation home rentals.

Nonetheless, the confluence of information technology, entrepreneurship, and venture capital has led to the creation of many new ventures, and that is generally good for any economy. Smartphone apps, cloud-based servers, and GPS technology have made various sharing services easy to launch and use. Many apps incorporate social media functions that allow users and providers to rate each other, which seems to increase trust among strangers. Venture capitalists have fueled growth by investing billions of dollars, especially in room sharing and ride sharing.



A cab driver mounted a sign over the cab light as part of a strike to protest against Uber in London in February 2016.

Demand is also there, especially in the U.S. A *Time* magazine poll published in January 2016 estimated that 42% of Americans had already used a sharing economy service, and that 22% had provided one. (By contrast, one observer in 2017 estimated that merely 1% of Japanese had used a sharing economy service.⁵) Most common among U.S. users were ride sharing (22%, led by Uber, Lyft, and Sidecar), room sharing (19%, led by Airbnb, VRBO, and Homeaway), and personal services (17%, led by Handy.com, Care.com, and Taskrabbit).⁷ Younger (ages 16–34) U.S. residents were the most common users and providers, though all ages participated.⁴

On the negative side, most sharing-economy ventures seem to be small, losing money, and surviving on venture capital. Airbnb is an exception. After raising \$3.1 billion in venture capital and reaching a valuation of about \$30 billion, it claimed to turn a profit in 2016. Uber is yet another story. After raising some \$10 billion in venture capital and achieving a recent valuation of nearly \$70 billion, it lost \$2.8 billion in 2016 on revenues of \$6.5 billion. Scan-

dals and management turmoil also led to the replacement of Uber's founder Travis Kalanik as CEO. How can a platform as popular as Uber be so valuable and lose so much money? Here is where the promise of the sharing economy meets reality.

First, are weak network effects (see “The Evolution of Platform Thinking,” *Communications*, January 2010). These are positive feedback loops whereby each additional user or complementary product or service should provide an increasing benefit to other platform users, resulting in geometric or even exponential growth in value. In Uber's case, the more drivers there are, the more convenient Uber becomes. The more riders Uber attracts, the more drivers are likely to want to work for Uber. More drivers make the service more convenient and should attract more users. Complementors are also moving in with third-party apps that integrate Uber with other Web sites and enhance the riding experience.⁶ However, Uber's apps ecosystem is still nascent. And most of Uber's network effects are local, not regional or

global. Adding more drivers in San Francisco adds some benefit to Uber users, but only when they travel.

Second, are limited economies of scale, apart from network effects. The new ventures often have huge expenses for sales and marketing, and R&D. Legal challenges also add costs and depress operations. Uber recently lost its license to operate in London, and it barely functions in Japan, where drivers must have taxi licenses. Airbnb has had to negotiate with several cities to pay hotel taxes and avoid being shut down. Hotels are fighting back with new services and pricing schemes.

Third, is “multi-homing.” Customers can use traditional services or competing platforms (their “homes” for the activities). There is not much to stop an Uber user from also using Lyft, Zipcar, Car2Go, or regular taxis.

Fourth, are subsidies. Platforms bring together users with providers, but one “side” of the market is usually more important to attract the other side. The platform company must identify and subsidize that all-important side. Airbnb must have rooms to rent in order to



Association for
Computing Machinery

ACM Transactions on Spatial Algorithms and Systems

ACM Transactions on Spatial Algorithms and Systems (TSAS) is a new scholarly journal that publishes high-quality papers on all aspects of spatial algorithms and systems and closely related disciplines. It has a multi-disciplinary perspective spanning a large number of areas where spatial data is manipulated or visualized (regardless of how it is specified - i.e., geometrically or textually), such as: geography, geographic information systems (GIS), geospatial and spatiotemporal databases, spatial and metric indexing, location-based services, web-based spatial applications, geographic information retrieval (GIR), spatial reasoning and mining, securing and privacy, as well as the related visual computing areas of computer graphics, computer vision, solid modeling, and visualization where the spatial, geospatial, and spatiotemporal data is central.

The journal is committed to the timely dissemination of research results in the area of spatial algorithms and systems.




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

Subscribe at www.acm.org/subscribe

attract renters. Uber must have drivers roaming about in order to attract riders. To list a room on Airbnb is free. When a customer rents a room, that person then pays the renter through Airbnb, which takes 6% to 12% from the guest's fee plus 3% to 5% from the property owner.³ By contrast, to attract and keep drivers, Uber pays an hourly wage in addition to a commission.² At the same time, Uber keeps prices artificially low to make the service cheaper than taxis or competitors. Thus, in contrast to Airbnb (which charges its two market sides), Uber subsidizes both of its market sides—drivers and riders.

Uber is also investing millions of dollars in autonomous vehicle research in the hope that, someday, it can eliminate drivers. If and when Uber does go driverless, it will then have to own or rent a fleet of cars—yet another enormous expense. The company may hope to drive competitors out of business and then raise prices, but Uber is likely to run out of venture capital long before that happens.

Most other sharing economy ventures are very small in comparison to Uber, but many seem to suffer from the same problem—they must spend more than their revenues on subsidies and marketing to get users and providers. We will know for sure how viable these ventures are for the long run as more time passes and when more of them release detailed financial information or simply run out of venture capital and shut down. Meanwhile, the bigger Uber gets, the more money it loses. Its business model does not bode well for the future of the sharing economy. 

References

1. Haworth, J. and Vaughan, R. *The Sharing Economy—Sizing the Revenue Opportunity*. PricewaterhouseCoopers, 2014.
2. Huet, E. Uber's newest tactic: Pay drivers more than they earn. *Forbes.com* (July 2, 2014).
3. Mitra, S. Here are the numbers behind Airbnb's staggering growth. *Inc.com* (Feb. 27, 2017); <http://bit.ly/2hwu1uf>
4. Quinones, A. and Augustine, A. Technology and trust: How the sharing economy is changing consumer behavior. *U.S. Banking Watch* (Nov. 19, 2015).
5. Romero, T. What you don't know about Japan's sharing economy. *Disrupting Japan Podcast* (Aug. 22, 2017).
6. Russell, J. and Lunden, I. Uber plans to turn its app into a content marketplace during rides. *TechCrunch* (Mar. 3, 2017).
7. Steinmetz, K. See how big the gig economy really is. *Time* (Jan. 6, 2016).

Michael A. Cusumano (cusumano@mit.edu) is a professor at the MIT Sloan School of Management and founding director of the Tokyo Entrepreneurship and Innovation Center at Tokyo University of Science.

Copyright held by author.

Law and Technology

How Law and Computer Science Can Work Together to Improve the Information Society

Seeking to remedy bad legislation with good science.

IN THIS COLUMN, I explore the various means by which lawyers can be helped by computer scientists to stop the (inevitable) collateral damage to innovation when the unstoppable force of legislation hits the irresistible innovation of the Internet.¹ I will explore some current controversies (fake news, Net neutrality, platform regulation) from an international perspective. The conclusion is familiar: lawyers and computer scientists need each other to prevent a disastrous retrenchment toward splintered national-regional intranets. To avoid that, we need to be intellectually pragmatic in pursuing what may be a mutually disagreeable aim: minimal legislative reform to achieve coregulation using the most independent expert advice. The alternatives are to allow libertarian advocates to so enrage politicians that severe overregulation results.

Regulation should first do no harm. That is easy to state, difficult to achieve, when legislation is the clumsiest version of the engineering principle of the ‘Birmingham Screwdriver’: to a legislator, every problem looks like a new bill will solve it, and worse, to an international lawyer every problem looks like a new Convention or Treaty is needed. Yet in reality, all that law can achieve is to enforce against a few bad actors to prevent the



most egregious overreaching by companies and users. More negatively, the worst law can do is overlegislate in the interests of monopolies old and new to prevent technological progress (one example: a man carrying a red flag in front of the first motor vehicles, which protected stagecoaches and railways from innovative competition).^{2,a}

a United Kingdom: Locomotives Act 1865 s.3 (An Act for farther regulating the use of Locomotives on Turnpike and other roads for Agricultural and other purposes: 28 & 29 Vic. Chapter lxxxiii).

Heroic policy interventions by government often fail, especially when aimed at industry self-regulation. Governments for the last 20 years have spent a great deal of time realizing how little governments can do effectively in steering toward better self-regulation by the industry, as well as the unintended consequences of badly constructed legal mechanisms. Experts have to continually advise in favor of independent scientific evidence gathering and against corporate and public safety advocates’ claims that the sky is falling. The

time-limited and dynamic ‘truth’ (or at least second-best solution) has always lain in the expert opinion that was not clamoring for attention, but reaching those experts requires patience and focus on the part of often distracted policymakers. Twenty years ago, it was pornography/Napster copytheft; now it is more likely ‘fake news’/‘Net neutrality’/‘platform regulation’. I will briefly sketch those three current controversies, as these case studies help us see what goes wrong in law enforcement and how to (partially) remedy that bad lawmaking with good science.

Fake News

‘Fake news’ is the heartfelt cry of politicians who feel wronged by the online media. Ad blocking and filter bubbles have made consumers and voters harder to reach. Industrial scale behavioral profiling and viral marketing via Twitter bots are a new method to so do. The expansion of social networking and smartphones means that new methods of communication are necessary, and consumers-voters are filtering out content they do not like.

That is not new—it applied to the tabloid newspapers methods of ‘yellow’ journalism, radio news and telegraph-supplied newswires 100 years ago. Unfortunately, the failure to adopt a universal independent public service model then meant the public was inflamed by irresponsible media into a series of wars for the first time made global by the same communications means (telegraphs, railways, radio, long-distance reliable air and maritime transport) that enabled the mass media. Today, the calls for fake news regulation pay no regard to both historians of technology and legal historians who can advise on public service media. Twitter and Facebook offer parallels to Hearst newspapers and radio broadcasts.

It is high time for an interdisciplinary project exploring how to avoid the same disastrous outcomes. Computer technology is a tool for the powerful; that insight is not new but politicians are ignoring the previous generations of transformative technology and our attempts to marshal them. More obviously, politicians are not fully using

the tools of behavioral insight to explore how to regulate fake news and social networking: evolutionary economics and behavioral neuroscience tells us how we become addicted to social media, yet legislators are only beginning to consult the experts to explore how social networking affects our behavior in fundamental ways.⁶ Social and economic sciences, as well as computer scientists, and neuroscientist, can help lawyers convince legislators not to be silly.

Net Neutrality

Net neutrality is a simple term that describes the complicated reality of highly complex engineering task: how to permit sufficient permission-free innovation in the network. The over-politicized doomsayers on both sides fail to mention what is becoming abundantly clear: policy can only partially steer traffic management practices. Net neutrality can do no more than prevent large telecoms companies continuing blocking Skype and WhatsApp or throttling back video traffic their subscribers want to see.

Net neutrality cannot stop innovation by telecoms companies (whose own corporate histories show a somewhat checkered relationship with Internet protocol network deployment). Regulators are simply not that competent, even if they had the resources and will to carry out laws to the letter, which they do not. More scientific exploration of the limited effects of Net neutrality policies would be rather useful. An example of regulators trying to do this in a non-confrontational manner is the extensive work produced by the Body of European

Heroic policy interventions by government often fail, especially when aimed at industry self-regulation.

Regulators of Electronic Communications.

How can legislators discuss complex laws when they do not know the difference between an Internet access provider (IAP) and an Internet service provider (ISP)? In European law, an access provider (telco) is an ‘Electronic Communications Service Provider’ (ECSP), distinct from an Information Society Service Provider (ISSP). ‘Information Society’ was Europe’s rhetorical counterpoint to Al Gore’s ‘Information Superhighway.’ Lawyers often fail to master these terms.

Minimal rules made sensibly by technically proficient people are achieving quietly what millions of email messages to regulators and legislators cannot: conduct rules to stop telecoms companies blocking legitimate content while giving them the latitude to experiment where not harmful to the public Internet. Note that common carriage was a rather successful way of delivering public (alongside private and business) communications services in previous technologies.

Platform Regulation

Politicians ask: What is the difference between platforms and networks? Journalists confuse their readers by referring to all those companies as ISPs—even though access networks perform fundamental and entirely different functions than social networks or search engines. This is the canard thrown into the Net neutrality debate by those telcos. If we are regulated, they argue, the same should also apply to the giant monopolies of Google, Apple, Facebook, and Amazon (known collectively as ‘GAFA’). The very high public profiles of Twitter and Snap cause issues, as they are by no means monopolies.

Google was fined €2.4billion by the European Commission in June 2017 for antitrust violations because of the links between its search engine and shopping platform. Google avoided an adverse outcome for eight years, a delay even longer than Microsoft under European investigation (a complaint in 1998 resulted in enforcement from 2004: Case T-201/04).

E-commerce dominated by the GAFA platforms is becoming a major

political issue especially in Europe, where mass youth unemployment and a rapidly ageing workforce means IT skills are in short supply, especially in Parliaments. Google- or Uber-sponsored promises of untold riches from autonomous vehicles fall on politicians' deaf ears: robots do not vote. This is a red flag to those advising governments as well as those legislating.²

If platform regulation signals a desire to slow down the pace of innovation by government, what rational answer can be sold to government? The first essential is to prevent platforms becoming liable as publishers, by whatever legitimate means necessary. That may mean fines for failure to take down fake content or revenge porn. It may mean a user ombudsman as suggested in new proposed English legislation. Recruiting more internal content checkers at Facebook and Google to remove content may be overdue. Global platforms need to conform to European rules on hate speech (for instance Nazi content), a legal battle lost by Yahoo! in the French *Tribunal de Grand Instance* 17 years ago.^b

Co-Regulation as a Hybrid Solution

What more can be done? Europe sets the global standards for regulation of content, notably in data protection and hate speech. The decisive power relationship in European law has swung to Germany and France. Regulation will increase, and Anglo-American companies increasingly recognize that and are embracing a French term: co-regulation. What that means is diluting government control of the Internet by ensuring a compromise based on industry self-regulation, but with oversight by users and by government regulators.³ Examples include global Top Level domain name oversight. Governments have sponsored industry standards not only in Europe but globally via hosting and supporting the World Wide Web Consortium with industry.

Co-regulation is the compromise computer scientists must live with. Totalitarian regimes want to use the

Co-regulation is the compromise computer scientists must live with.

threat of terrorism and cyber-crime to replace self-regulation with direct and often draconian control. Co-regulation is the best alternative.

Areas for cooperation between law and computer science can flourish in co-regulatory institutions, because the best of them engineer a deliberative evidence-driven expert-friendly process.⁵ It can curb the worst excesses of both corporate and government control.

If lawyers and computer scientists cooperate to make these social regulation processes work, it is the best chance to prevent a much worse system of direct government control emerging. **C**

References

1. Brown, I. and Marsden, C. *Regulating Code: Good Governance and Better Regulation in the Information Age*. MIT Press, Cambridge, MA, 2013.
2. Guadamuz A. and Marsden, C. Blockchains not Bitcoin: Distributed Ledger Technology, *Computers and Law* 2, 2016; <http://www.scl.org/site.aspx?i=ed46568>
3. Marsden, C. *Internet Co-regulation: European Law, Regulatory Governance and Legitimacy in Cyberspace*. Cambridge University Press, Cambridge, U.K., 2011.
4. Marsden, C. Technology and the law. *International Encyclopedia of Digital Communication & Society* Wiley-Blackwell, 2015; DOI: 10.1002/9781118767771.wbiedcs138
5. Marsden, C. et al. Deliverable 4.3: Final Report on Regulation, Governance, and Standard, 2015. European Internet Science Consortium; <http://www.internet-science.eu/groups/governance-regulation-and-standards>
6. Pollett, T.V., Roberts, S., and Dunbar, R.I.M. Use of social network sites and instant messaging does not lead to increased offline social network size, or to emotionally closer relationships with offline network members *Cyberpsychology, Behavior and Social Networking* 14 (2011), 253–258.

Chris Marsden (c.marsden@sussex.ac.uk) is Professor of Law at the University of Sussex, Brighton, U.K.

The argument in this column is further explored in the final chapters of author's recent book *Network Neutrality: From Policy to Law to Regulation*. Manchester University Press, 2017.

^b Confirmed in *Yahoo! Inc. v. La Ligue Contre Le Racisme et L'antisemitisme*. L'Union Des Etudiants Juifs De France, 433 F.3d 1199 (9th Cir. 2006); <http://bit.ly/2f8Oi59>

Calendar of Events

January

January 7–10

GROUP '18:
2018 ACM Conference on Supporting Groupwork, Sanibel Island, FL,
Sponsored: ACM/SIG,
Contact: Michael Prilla,
Email: prilla.michael@googlemail.com

January 8–13

POPL '18: The 45th Annual ACM SIGPLAN Symposium on Principles of Programming Languages, Los Angeles, CA,
Sponsored: ACM/SIG,
Contact: Ranjit Jhala,
Email: jhala@cs.ucsd.edu

January 22–25

ASPDAC '18: 23rd Asia and South Pacific Design Automation Conference Jeju, Republic of Korea,
Contact: Youngsoo Shin,
Email: youngsoo@ee.kaist.ac.kr

February

February 21–24

SIGCSE '18: The 49th ACM Technical Symposium on Computing Science Education, Baltimore, MD,
Sponsored: ACM/SIG,
Contact: Tiffany Barnes,
Email: tiffany.barnes@gmail.com

February 24–28

CGO '18: 16th Annual IEEE/ACM International Symposium on Code Generation and Optimization. Vösendorf/Vienna, Austria,
Contact: Jens Knoop,
Email: knoop@complang.tuwien.ac.at

February 24–28

PPoPP '18: 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Vienna, Austria,
Co-Sponsored: ACM/SIG,
Contact: Andreas Krall,
Email: andi@complang.tuwien.ac.at

Historical Reflections Defining American Greatness: IBM from Watson to Trump

Reflections on a firm that encapsulated the American Century.

WE HAVE BEEN hearing a lot lately on the topic of American greatness, where it went, and how to reclaim it. But greatness is complicated, and our ideas of what is great and what is not have changed over time. In this column, I ask what the history of one mighty corporation—IBM—can tell us about the rise and fall of a particular kind of American greatness.

Decade after decade, IBM has been one of the world's largest, most profitable, and most admired companies. Of all American businesses, only General Electric, Apple, Microsoft, and Exxon-Mobile have generated more wealth.^a Despite recent troubles, it has been ranked in the 2010s as the number one company for leaders (*Fortune*), the greenest company (*Newsweek*), the second most valuable global brand (*Interbrand*), the second most respected company (*Barron's*) and the fifth most admired (*Fortune*). IBM technical contributions to computing are second to none. Its researchers won six Turing awards and, more startling, four Nobel prizes. Its engineers produced the first hard disk drive, the first floppy disk drive, the first architecture implemented over a range of diverse but compatible machines, the first widely used high-level programming language, the

^a See <http://nyti.ms/2wESvrA>



Thomas J. Watson

relational database, the first scientific supercomputer, the first RISC designs, and the first DRAM chip.

As recently as 2014, IBM ranked ahead of its old adversary Microsoft on *Fortune's* lists of the largest U.S. companies (20th place) and of the firms most admired by managers (16th place).

In ways good and bad, IBM has been at the forefront of changes in American business and in America's relationships with the world.

Chasing Greatness

IBM's roots go back to the Tabulating Machine Company, founded by Her-

man Hollerith in 1896 to quickly tally data punched onto cards. The initial application was the U.S. Census, but punched card systems were also taken up for statistical applications in business. In 1911, Hollerith's firm was merged with several other producers of specialized business equipment to produce CTR, the Computing-Tabulating-Recording Company. The "Computing" part referred to weighing scales—an interesting example of how language evolves over time. Thomas J. Watson took over as manager in 1914. Ten years later CTR was renamed International Business Machines—less specific and more ambitious. It grew rapidly and consistently.

Watson was first and foremost a salesman, having learned his trade at National Cash Register, which was famous for its innovative sales practices. He loved public spectacles, gathering crowds at company rallies and picnics to listen to his speeches. The firm's strict dress code approximated a uniform: shirts were white, faces hairless.

Workers sang from the company songbook. Its anthem "Ever Onward: IBM Rally Song" promised "We're big, but bigger we will be, we can't fail, for all can see that to serve humanity has been our aim." Watson built a company with a sense of manifest destiny and an obsession with greatness. The word "great" appeared at least 45 times in the 1935 edition of *Songs of the IBM*, which began with "America" and "The Star Spangled Banner." The 86 new songs that followed celebrated every aspect of the company. They included such gems as "To Our I.B.M. Girls" and "To W.F. Titus, Assistant to the President, In Charge at Endicott."

To Watson, greatness meant investing in workers inside and outside work hours, to motivate them and see to the threats of labor unrest and unionization. As IBM grew it copied NCR by introducing what historians have termed "welfare capitalism." Such firms promoted themselves as showpieces of modern labor practices, providing benefits programs such as pensions and profit sharing, worker education programs (such as art classes), and sports leagues. It eventually built its own country club, and it also built career ladders so that employees starting out in manufacturing or clerical posi-

Watson was savvy politically.

tions could advance into technical or management jobs.

Watson was savvy politically. As other firms withered during the Great Depression of the 1930s, IBM bloomed, doubling its revenues from 1935 to 1940. Much business came from the New Deal's expansion of government bureaucracy, including the establishment of the Social Security Administration, which greatly increased demand for punched card equipment. When war followed depression, IBM supplied the tools to coordinate mobilization and diversified its production to create special machines for codebreaking and other military applications. Its revenues tripled from 1940 to 1945.

International Greatness

As chair of the International Chamber of Commerce Watson promoted "World Peace Through World Trade" during the turbulent 1930s, preaching the benefits of free trade. Watson always had global ambitions for his firm. The "I," after all, stood for International. Greatness meant engaging with the world to promote peace and stability, which were good for business. One of the firm's less well-known songs, "To Our Great I.B.M.," celebrated the firm's staff "overseas—in the heat of the tropics and northern cold," and praised "I.B.M. all glorious—on which the sun never does set." Watson loved arriving in far-flung places to open new subsidiaries and be acclaimed by local dignitaries. The song called him "our great president—king of the business world."

The largest and best-established foreign markets were in Europe, where IBM had a variety of distribution agreements, partnerships, and ownership stakes. When war came, Watson lost control over most of these, as its German subsidiary was expropriated by the Nazi regime and given control over operations in conquered countries. The end of the war

gave IBM a chance to rebuild its European operations on a new model. The national subsidiaries came under the firm control of a new group, the IBM World Trade Organization.

European politicians loved exports but hated imports. Their ruined economies were short of foreign currency. Unfortunately, one is not possible without the other. In research with Petri Paju, I explored the novel arrangement IBM came up with to rebuild international trade at a time when wartime devastation, shortages of currency, rampant import tariffs, and strong political opposition to trade deficits made free trade impossible. (See "IBM Rebuilds Europe" in the Further Reading section at the end of this column.)

This plan centered on a humble product: the electric typewriter. In comparison to punched card systems, they were affordable and easy to introduce one at a time into a business. This made them an ideal product for IBM's new European subsidiaries to sell, building trust and relationships with local businesses who might later adopt punched card systems or computers. IBM avoided the highest tariffs, on finished goods, by setting up typewriter assembly plants in its eight major European markets.

That was practical, if suboptimal, but it could never make the thousands of parts needed for an electric typewriter in a small country like the Netherlands. Neither, for political reasons, could it import all of them. IBM's ingenious solution was to make sure that some pieces of every typewriter were manufactured within each country, to be traded internally for the other parts needed. This let its European subsidiaries exchange goods without creating trade deficits.

IBM and the U.S. both deepened their engagement with Europe at around the same time, and for similar reasons. IBM devised its interchange plan in the early days of the Cold War, as the Soviet Union consolidated its control over Eastern Europe and seemed poised to extend its reach further West. A strong supporter of the U.S. Marshall Plan, Watson believed that restarting the European economy would sell capitalism to wavering populations, bolster the position of

the U.S., and rebuild trust between IBM staff in formerly hostile nations. Its public relations effort promoted the program in *Reader's Digest* as “a small but perhaps significant step towards a more unified Europe.” In recognition of these efforts French foreign minister Robert Schuman, an architect of what became the European Union, arranged for Watson to receive one of his country's highest honors, induction as a Grand Officer of the Legion of Honor. IBM had a vested interest in the stability, economic openness, and growth of its foreign markets, and suffered when, as in South America during the 1970s, these conditions were not met.^b

Watson died in 1956, shortly after passing control of IBM to his son, Thomas Watson Jr. As trade barriers came down and Europe's economies recovered, the firm's international operations expanded further. IBM's post-War decades, like those of the U.S. itself, were shaped both by the unifying tensions of the Cold War and the opportunities offered by a world in which major economies were recovering from devastation quickly enough to bolster its revenues but not yet to threaten its dominance.

IBM found ways to position its national subsidiaries as contributing both to the greatness of their host countries and to its own greatness as an increasingly global business. Many subsidiaries developed design and research capabilities as well as manufacturing, sales, and support. They were staffed and managed primarily by local employees, giving them dual status as locally rooted firms as well as agents of American economic power. Staff interacted at different levels. In IBM Finland, where our research has focused, staff interacted mostly with Finnish customers and colleagues. For training courses they worked with other Nordic people at the regional training center in Sweden. Senior or fast-tracked staff might travel further, to attend meetings at IBM's European headquarters in Paris, for a temporary assignment at a research facility in another country, or to attend a lavish

^b Medina, E. Big Blue in the bottomless pit: The early years of IBM Chile. *IEEE Annals of the History of Computing* 30, 4 (Apr. 2008), 26–41.

Greatness meant taking risks and making long-term investments in new technologies and platforms.

international meeting of IBM's 100% Club for its best salespeople.^c

Making America Great

Back in the U.S., IBM's rise to greatness in the emerging computer business was underwritten by a charter membership in what President Eisenhower called the “military-industrial-academic complex” of the early Cold War. Its first commercial computer model, the IBM 701, was branded the “defense calculator.” Most went to atomic labs and aerospace engineering firms working on government projects. Government officials managed the delivery queue, ranking customers by their contribution to national security.

In the late 1950s IBM had the same markets in mind when developing STRETCH, arguably the first supercomputer. STRETCH introduced new semiconductor packaging, new architectural features such as pipelining and memory protection, and new capabilities such as multitasking. A modified STRETCH was the core of the government's HARVEST system for message decryption and keyword searching, the ancestor of today's NSA data mining efforts. When first tested in 1961 STRETCH ran 30 times faster than IBM's previous flagship system, a leap forward that would be unthinkable today. But unfortunately IBM had promised pre-launch customers that STRETCH would be at least 60 times faster, and priced it accordingly. It was immediately pulled from the market, but in the decades since historians have highlighted the importance of the new technologies developed during the proj-

^c Paju, P. and Haigh, T. IBM's tiny peripheral: Finland and the tensions of transnationality. To be published in *Business History Review*, 2018.

ect for IBM's subsequent domination of the mainframe computer market.

Still more important to IBM's electronic transformation was SAGE, a collection of air defense centers hooked up to radar stations to direct fighters to intercept Soviet bombers. I often come across fanciful claims that early computers such as ENIAC or Colossus filled an entire building (or even a city block). The AN/FSQ-7 computers at the heart of SAGE were the only ones that came close. Two full installations, clustered redundantly, together filled one entire floor of each of the 23 SAGE control centers. SAGE tapped the urgent flow of government money to defense projects, raised through marginal income tax rates of up to 91% on wealthy Americans like Watson, to push the vacuum-tube technology of the day toward capabilities that would never have been viable for commercial applications. It pioneered computer networks, real time operation, and graphical displays.

Although SAGE itself targeted bombers rather than missiles, and so was strategically outmoded before it was fully deployed, it remained operational into the 1980s. Its true legacy was technological. While IBM's mainstream business evolved gradually from punched cards to small computers, its defense projects jumped straight into the future. America's Cold War strategy of containing Soviet expansion without fighting a large-scale war required clear superiority in military technology and industrial productivity. The technologies that IBM and its industrial and academic partners, such as MIT and the System Development Corporation, developed for SAGE and STRETCH helped to preserve those advantages.

Taking a Great Risk

Greatness meant taking risks and making massive long-term investments in new technologies and platforms. During the 1960s IBM, home of the dark suit and starched collar, made a gamble on a scale that would appall today's hoodie-clad, thrill seeking, Silicon Valley executives. While today's tech companies like to boast about their “moonshot” projects, firms such as Alphabet, Facebook, and Oracle derive the vast majority of their revenues from a hand-

ful of products and services that dominate their respective niches. Backward compatibility is more important than innovation, as Microsoft showed when turning Windows and Office products into lucrative monopolies. Most tech firms develop a platform as startups and grow it steadily over many years, defending it against threats from upstart competitors.

By the 1970s IBM held a similarly dominant position in the business-oriented mainframe market, but only because it launched a massive research and development effort during the early 1960s. In April 1964 when IBM introduced its System/360 range it rendered its entire installed product base obsolete. Two years earlier IBM had reportedly budgeted \$5 billion to develop the new range, twice its entire annual revenue. According to Watson Jr. the actual costs were so high that in 1965 the firm unexpectedly found itself just “a few weeks” from needing “emergency loans to meet the payroll.” (See “Father, Son & Co” in the Further Reading section.)

Before this IBM offered several incompatible ranges of computers for different markets and sizes of company. Customers increasingly expected to receive bundled with their machines an extensive collection of systems software packages, such as programming languages and operating systems. The cost of developing these tools across so many different platforms was rising. Incompatibility also threatened customer loyalty. Computers became obsolete every few years, but successful application programs could evolve over decades of use. Any computer upgrade that required customers to recode their applications was a chance for IBM’s competitors to steal an account away, particularly as the newly introduced COBOL language was intended to let users move applications between computers from different vendors.

It was this project that introduced the phrase “computer architecture,” to describe design features at a more abstract level than their implementation in a specific computer model. System/360 imposed a common machine instruction set, word length, character set, and peripheral interface over an industry-spanning range

of machines. For example, smaller machines implemented in micro-code instructions that were handled directly in hardware by the larger ones. From smallest to largest, the first wave of System/360 machines differed by a factor of 500 in memory capacity and a factor of approximately 20 in processor performance.

The machines were all supposed to run a single operating system, with the bigger ones loading more modules and turning on more features. That part of the plan didn’t work out so well. Fred Brooks’ classic study of the problems of OS/360, *The Mythical Man Month*, was a founding text of software engineering and helped win him an ACM A.M. Turing Award.

Despite these problems, IBM’s gamble paid off magnificently. IBM’s core platform evolved through countless new models over the next 50 years, though the 370, 380, and 390 ranges and on to today’s System z machines. These never dropped backward compatibility with the 360 range and IBM never lost its dominance of the mainframe market. Its most direct competitors of the 1960s, including General Electric, which was still a much larger firm, were unwilling to match this investment. Most exited the computer business over the next decade. IBM was dominant at home and abroad.

Supporting Great Design

For Watson Jr., greatness included a commitment to elegance in design and the finest in modern architecture. An IBM System 360 featured prominently in the recent TV show *Mad Men*, where its stylish complexity symbolized the rise of analytic approaches to advertising. Today the clean, confident design aesthetic of the 1950s and early 1960s is more popular than ever. Period houses, furniture, and consumer products sell for a premium. No company did more to popularize that aesthetic and bring it into the American mainstream than IBM.

As documented by John Harwood in his book *The Interface: IBM and the Transformation of Corporate Design, 1945–1976*, IBM’s design chief, Eliot Noyes, assembled one of the most influential teams in history. Their skills were applied not only to the firm’s computers and office products, which

received a unified and stylish industrial design language, but also to documentation, public exhibits, and architecture. IBM hired Charles and Ray Eames, probably best remembered today for their iconic chairs, to produce one of the earliest exhibits on the history of computing. Its landmark buildings, which hoisted the firm’s logo like a flag in the leading cities of the free world, were designed by star architects such as Mies van der Rohe.

IBM’s greatness also rested on its commitment to science. For its research headquarters, in Yorktown Heights, IBM turned to industrial architect Eero Saarinen, responsible for such futuristically iconic structures as the Gateway Arch in St. Louis and the TWA terminal at JFK airport. The lab curves in an oval shape, glowing in the dark like a flying saucer. Into the 1950s IBM retained a hands-on, product-centered engineering culture, long after Bell Labs and General Electric hired scientists and built up centralized research and development centers. By the 1960s, however, its international network of research facilities set the standard for corporate commitment to research. Its researchers, envisioned by Saarinen as “tweedy pipe-smoking men,” enjoyed the enviable conjunction of university-like research facilities with IBM’s generous pay and benefits and freedom from teaching duties. The firm’s Nobel prizes came from basic research into fields such as superconductivity and electron microscopy. IBM’s willingness to fund basic science reflected the many possibilities for payback across the huge range of products it developed and manufactured, from semiconductors and core memories to disk drives, keyboards, punched cards, printers, and dictating machines.

Threats to Greatness

The most successful computing firms of the 1970s and early-1980s targeted niche markets where IBM’s core architecture did not compete well. DEC established a huge new market for minicomputers, SDC and Cray targeted supercomputers, while Wang built a thriving business around word processing and office automation. IBM offered credible projects in most of

these areas, but it came to them after other firms were already established. Because it had to cover its heavy overheads for things like research and management, to procure parts from its own factories, and to avoid threatening the products of other divisions its smaller machines tended to be less powerful or more expensive than those of competitors. Even new technologies developed within IBM, such as RISC processor architectures and floppy disks, were deployed more aggressively by its competitors. Its strengths in build quality, sales, and support could sometimes offset these disadvantages.

The booming market for personal computers posed a particular challenge to IBM. These machines were much cheaper than traditional computers, had shortened design and production cycles, and were often purchased by individuals or departments rather than by corporate information systems departments. In 1981 IBM startled the world with its 5150 Personal Computer. Like competing models, but unlike other IBM products, it was designed quickly by a small team using standard parts. The IBM PC set a standard for personal computers that quickly dominated the market, and, as I explained in an earlier column, gradually evolved into the platform that dominates desktop, laptop, server, and super-computer markets today.^d

In the longer run, however, IBM was ill equipped to compete in the new market it had defined. IBM's efforts to fight the producers of "clone" machines by shifting the market to a new, proprietary, standard, the PS/2 range, failed miserably. In the early 1990s IBM still produced solid personal computers, and enjoyed great success with its business-oriented ThinkPad laptops, but faced a fiercely contested market in which it had no inherent competitive advantages.

Redefining Greatness

IBM's old factories in Endicott, its former headquarters, now lie in ruins. So do many of its other manufactur-

ing facilities around the world, such as a plant in Greenock, Scotland that once employed 13,000 people. Total IBM employment has fluctuated in recent years around 390,000, similar to its 1992 peak, but today many more of these people are believed to work in India than in the U.S.^e (In 2010, IBM stopped releasing counts of its employees by country). IBM is no longer a predominantly American firm, or a clear leader in its central areas of business. It has retreated from its old commitments. If today's IBM seems less monumental, less public-spirited, less consequential than the "Big Blue" of old, we should look not so much to the firm itself as to the evolution of global capitalism since the 1980s.

In 1993, IBM declared what was at that point the largest annual loss of any company in history: \$5 billion. Its new manager, Louis Gerstner, faced calls to break up the company and sell off the parts. Instead, he began a process of strategic reorientation away from mass-market hardware and toward software and business services. Gerstner's initial cull of 60,000 jobs, which to this day remains corporate America's largest single layoff announcement ever made, signaled the arrival of a new IBM culture in which the firm no longer made a lifetime commitment to its employees. To survive, IBM was going to have to move on from the legacy of the Watsons and find a new definition of greatness.

Turning around IBM was seen, with some justification, as a managerial triumph. Monumental tech corporations crumble quickly to sand. Few have faced a serious crisis and recovered. After stumbling, once-dominant firms such as Wang, Data General, Palm, DEC, Digital Research, Novell, Word Perfect, Lotus, and Compaq were quickly acquired and assimilated by more successful competitors. Most recently, the remains of Yahoo were acquired by Verizon, to be combined with the wreckage of AOL. Verizon is jettisoning the names, calculating that their brand history has a negative value.

Like newer tech firms such as Google, IBM deliberately fostered a distinctive corporate culture, shap-

ing its organizational structures as carefully as its products. IBM tended to be a reasonably early and very thorough adopter of organizational innovations rather than a pioneer. It turned to management committees and strong divisional structures only after the death of Watson Sr. As it modernized, IBM embraced both corporate philanthropy and, following the racial tensions of the late-1960s and the women's liberation movement, the idea that corporations should be committed to gender and racial diversity in their hiring and promotion practices. Its adoption of benefits programs, research labs, and industrial design followed a similar trajectory, as did its more recent embrace of fashionable green and urban initiatives. To keep its place as a model of modern capitalism IBM has consistently adopted ideas just as they enter the corporate mainstream.

If anything IBM was a little late to the 1980s shareholder revolution, now associated more than anything else with the phrase "greed is good," that stressed the idea that companies exist to maximize shareholder value, and that this outweighs responsibilities to employees, communities, or social causes. But as with its earlier shifts, IBM embraced the emerging orthodoxy with a vengeance. This has meant shifting out of lower margin businesses. As troubles first mounted, IBM sold off its printer business in 1991 to create Lexmark. The process continued even after IBM returned to financial health. In 2003, it sold its hard disk drive business to Hitachi, in 2005 the iconic ThinkPad laptop brand went to Lenovo, followed in 2014 by its low-end server business. Its point-of-sale business sold to Toshiba in 2012, and its semiconductor manufacturing business was acquired by GlobalFoundries in 2015.

The new definition of greatness was crystalized by CEO Sam Palmisano in 2010 when he promised investors that IBM's earnings per share would double from 2010 to 2015. IBM attempted to maximize this metric by slashing costs, such as employees and investment in new technology, and by using funds from selling off chunks of the company to repurchase shares. These steps hurt IBM's position in new battlegrounds

^d Haigh, T. The IBM PC: From beige box to industry standard. *Commun. ACM* 55, 1 (Jan. 2012), 35-37.

^e See <http://nyti.ms/2xMsmWT>

such as cloud computing. Despite recent efforts to change course, IBM's revenues have now fallen for 22 successive quarters. IBM has attempted to shrink its way back into to greatness.

Watson and Trump

IBM sold its landmark building on the Chicago River to a private equity group. Its giant logo has been taken down, but a massive TRUMP emblem now glows on the skyscraper next door. This seems somehow appropriate, as if the retreat of IBM from its mid-century version of American greatness opened the door to the rise of a rather different sense of America's place in the world. Global capitalism, driven above all by financial metrics, has withdrawn its implicit long-term commitments to workers and nations. The American public has lost faith in big business, free trade, and international engagement. Trump promised to make America great again by building walls, stepping back from its commitment to the defense of NATO allies, and tearing up trade deals.


From this viewpoint, Thomas Watson Sr. and Donald Trump are the bookends of the "American Century." That phrase was promoted during the war in anticipation of a post-war world in which the U.S. maintained a stable, rules-based international order wherein its allies and businesses both prospered.

The odd thing is that as individuals they have a lot in common. Both ran family businesses shaped in their own images. Watson Sr. was remembered, even by his own son, as a difficult, changeable, and emotionally distant man. Like Trump, he fed on the energy of crowds, and preferred rallies to committee meetings. Both worked through personal relationships rather than organizational structures. Both were teetotal workaholics who, as businessmen, coveted access to politicians from both parties. Trump has changed his party registration at least five times. Watson was driven more by an admiration for power than any ideological convictions as his allegiances shifted smoothly from boosting Roosevelt's New Deal to launching Eisenhower's political career. *Time* magazine remembered Watson, accurately, as the greatest salesman of his age, but his greatest product was always him-

Turning around IBM was seen, with some justification, as a managerial triumph.

self. Watson's early forays abroad were as much about the joy of planting his flag in another country as about rational economics, to the extent that during the 1930s many subsidiaries were branded as Watson companies rather than IBM. Trump remade himself as an international brand to be licensed to property developers and other businesses after his attempts in the 1980s to build a more conventional business empire collapsed under mountains of debt and multiple bankruptcies. Long before IBM grew into a top-tier corporation Watson had made himself a household name and secured one of the country's biggest compensation packages. Trump's gift for publicity and role as a reality television star gave him a public profile incomparably higher than those of developers with far larger empires (spot quiz: name the heads of Vornado Realty Trust and The Related Companies.)

The fact that, despite these shared traits, Watson and Trump held such diverging ideas of how to make America great points to some much deeper shifts in politics and capitalism. Watson's vision of American-led global prosperity was always easy to criticize, particularly from those he hoped to lead, but compared to the alternatives it had a lot to recommend it. IBM's post-War European investments served the firm's interests, and those of the U.S., but they only paid off many years later once tariffs and capital controls were lifted and customers were able to afford more expensive punched card and computer systems. Watson was firmly committed to the dismantling of trade barriers, and his belief that extending supply chains across national borders

would foster peace and prosperity was vindicated by the expansion of the European Union in the 1990s. I suspect that Watson shared the opinion expressed to Congress in 1953 by Charles Wilson, another politically engaged business leader, that "what was good for our country was good for General Motors, and vice versa." Watson was sure that what was good for IBM was good for the U.S., but also good for the broader causes of global prosperity and liberal democracy. That blinkered and self-interested faith in corporate virtue crumbled long ago. Recently, I have been feeling more fond of vain, pompous old Thomas Watson Sr., and the empire he created, than I ever thought possible. To quote a short poem by Leonard Cohen, who died the day before Trump was elected, "oh, and one more thing/you aren't going to like/what comes after America." 

Further Reading

Watson, T., Jr. and Petre, P. *Father, Son & Co: My Life at IBM and Beyond*. Bantam, NY, 1990. One of a handful of truly excellent business memoirs, this covers Watson Jr.'s impressions of his father as well as his own work as leader of IBM.

Maney, K.

The Maverick and His Machine: Thomas Watson, Sr. and the Making of IBM. Wiley, NY, 2003. Biography of the elder Watson, focused on his difficult personality and role as the "first celebrity CEO."

Usselman, S.

IBM and its Imitators: Organizational Capabilities and the Emergence of the International Computer Industry. *Business History Review* 22, 1 (Jan. 1993), 1–35. An award-winning analysis of IBM's early involvement in computers.

Pugh, E.W.

Memories That Shaped an Industry: Decision Leading to IBM System/360. MIT Press, Cambridge, MA, 1984.

Pugh, E.W.

Building IBM: Shaping an Industry and its Technologies. MIT Press, Cambridge, MA, 1994. An insider analysis of the firm.

Paju, P., and Haigh, T.

IBM rebuilds Europe: The curious case of the transnational typewriter. *Enterprise & Society* 17, 2 (Feb. 2016), 265–300.

Thomas Haigh (thaigh@computer.org) is an associate professor of history at the University of Wisconsin—Milwaukee and Comenius Visiting Professor at Siegen University. Learn more at <http://www.tomandmaria.com/tom>.

Copyright held by author.

Viewpoint

Technology and the Failure of the University

Considering the double-edged sword of learning technologies in various academic settings.

THERE ARE PREDICTIONS that half of U.S. universities will fail in the next 15 years.^a Will technology be responsible for some or all of these failures, or does technology have the potential to save the American university? The purpose of this Viewpoint is to examine the dual role of technology in the future of higher education. It argues that technology-enhanced teaching and learning can dramatically improve the quality and success of higher education, but learning technologies alone will not save the university. However, universities that lack the leadership, motivation and the resources to innovate with technology are good candidates for failure.

Technology is a double-edged sword. Schools that embrace technology and use it to improve the educational process are in a much better position than those that do not. Success requires the commitment of the administration, faculty, staff and students and it requires substantial resources invested in the technology. Schools that lack the will and the resources are the ones most likely to fail. Students are going to expect the variety, flexibility, and richness that learning technologies bring to their



programs. The most vulnerable institutions are small, private colleges with low enrollments, heavy reliance on tuition income, and that are well known only within a 200-hundred-mile radius of their campus. They are unlikely to be able to afford to invest in the people and skills to infuse their programs with technology-enhanced teaching and learning.

Technology-Enhanced Teaching and Learning

The rapid adoption of new teaching technologies in universities has happened concomitantly with an emphasis in education circles on active as opposed to passive learning. Fortunately, the two trends reinforce each other; new technology-based approaches to education can be used to encourage

^a See <http://bit.ly/2zVvLE1> and <http://bit.ly/P1JuxO>

active learning. Active learning means that students are heavily involved in their own education as opposed to passively sitting through a lecture. Discussing a case study, working on a simulation, playing a game, breaking a class into teams to work on a problem are all examples of active learning.

Learning technology can be divided into synchronous and asynchronous components. Asynchronous refers to course materials and exercises that students access at a time and place of their own choosing while synchronous interaction means that some number of students and an instructor are interacting at the same time, usually through video conferencing system of some kind. Frequently, students access asynchronous material like readings, videos, simulations, and games through a learning management system like Blackboard or Canvas.

Figure 1 depicts several different types of courses; online, blended and MOOCs all make use of teaching and learning technologies. The traditional course before the availability of the technology described here featured a physical classroom with an instructor and students. Typically, the instructor lectured and encouraged discussion depending on the size of the class. Course materials came in the form of textbooks and reading packets in hard copy. The first online courses were mostly asynchronous with interaction between the instructor and students occurring through email and discussion boards. With the advent of reliable video conferencing software and fast Internet connections, it is possible to have synchronous online classes where the instructor and students interact in real time on a computer or mobile device. A blended course features physical interaction between the instructor and students with course materials, including lectures, available asynchronously for the student to access at her convenience. A blended course stresses active learning when the instructor relegates all lectures to videos and uses the physical class meeting for interaction among and with students, for example, discussing current issues, case studies, student presentations on a course topic and similar activities.

A massive open online course is a different animal altogether; virtually

Figure 1. Traditional versus technology-enhanced courses.

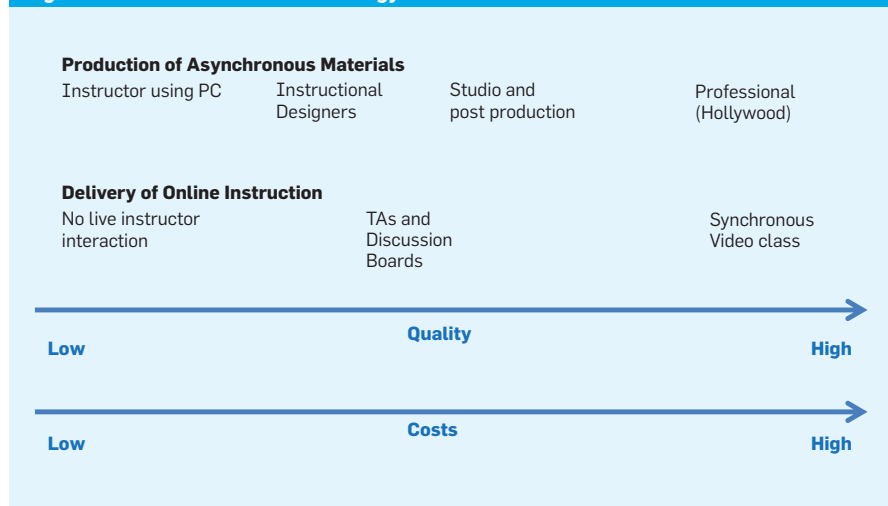
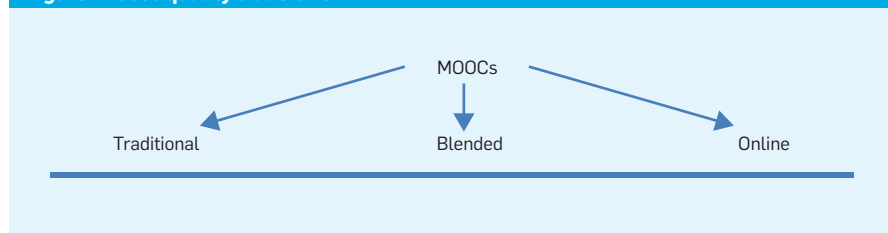


Figure 2. Cost-quality trade-offs.



all of the course occurs asynchronously without direct interaction with an instructor. MOOCs utilize videos viewed by students at their convenience. The major MOOC platforms provide the ability for students to submit homework some of which is graded automatically and some by peers. But MOOCs really are massive, with thousands of “learners” taking the courses, mostly for fun or for an inexpensive certificate for passing the course. Recently Coursera, the largest MOOC provider, has offered specializations by combining four courses or more in a particular area. A small number of colleges are using MOOCs to offer courses for credit including Georgia Tech with an MS in computer science for under \$7,000 and the University of Illinois with an MBA for about \$20,000. Students who want to take the MOOC-based degree programs have to apply to the universities and be accepted so that the schools can control the input and size of these specialized programs.

How Much Does All of This Cost?

Some have suggested that the technology will make it possible to dramatically reduce the cost of college through innovations like MOOCs, which scale

from a few students to hundreds of thousands with virtually no incremental costs. In this new world of learning technologies there are schools that are content producers and those that are content consumers. (Of course, the producers also consume content as well.) The content producer in general bears the full cost of developing technology-enhanced teaching and learning materials and systems.

The technology offers different options for creating content at different price points. An instructor might make 10–20 videos that replace lectures and post them to a learning management system. This instructor will also “curate” material from the Internet, adding YouTube videos and TED talks for example to her lectures. Some faculty will create the lectures using a PC with a touchscreen and pen, a Web cam, and video capture and editing software. Other faculty less comfortable with the technology will want to have a script and record the video in a studio; it takes considerable time to create a script and to undertake post-production of the video. The Smith School of Business, which has a number of blended courses, a MOOC and a completely online MBA program (with

two residence periods) as of this writing has seven instructional designers and an audiovisual studio with a staff of three; five years ago none of these positions existed.

What about a MOOC? The first MOOCs were prepared with a faculty member using a computer, but as with other developments, competition has driven the adoption of more costly production processes. Udacity, one of the MOOC platforms, supposedly budgets \$200,000 for each of its courses, and EdX, the other of the “big three” platforms charges \$250,000 to produce a MOOC.^b (It should be noted that many MOOCs are created for far less.)

Given potentially high investment costs, can the technology reduce the overall cost of college? One cannot talk about reducing the cost of education, and hence the price of education for students and their families, with learning technologies without talking about quality at the same time. Figure 2 illustrates the cost and quality trade-offs in the production and delivery of technology-enhanced courses. Quality is a subjective measure of the coverage, content, assignments, readings and student and peer evaluation of a course. On the production side the least cost option is for the instructor to create the course using a PC with limited editing; this approach is also likely to produce the least compelling asynchronous material. At the high end of the quality spectrum as well as the high cost side, the instructor creates and edits a script, records a video in a studio, and a post-production staff integrates illustrations with the video recording. The same trade-off in cost and quality exists for the delivery of courses; an online course can be completely asynchronous with little or no interaction between students and an instructor, or the course can feature interactive video classes held frequently during the time the course is running.

The MOOC-based course for credit has the potential to make dramatic changes in the cost of education as the two existing examples of Georgia Tech and Illinois illustrate. There is no question that the development costs

for MOOCs for programs like these can be fully amortized as the number of students scales up well beyond what a physical campus program could accommodate. Will the MOOC-based degree become the future of higher education? If so, a large number of schools that do not have the resources to offer these degrees will surely fail.

Threats to Technology

There are four major threats to the adoption of technology to enhance teaching and learning. The first of these and the most prominent is university faculty, many of whom oppose changing the system that has been in place for hundreds of years. The reward system in most schools and the inherent conservatism of faculty members in my experience create a huge barrier to adopting new technologies for education.

Assistant professors at research universities are rewarded for publishing scholarly articles and books, which they must do to be granted tenure. Learning and adopting new technologies takes time and is a huge risk for them.

Tenured faculty can largely do what they want, and they by the time of receiving tenure have fallen into a rhythm of research and teaching; once tenured they are expected to undertake more service to the school. What is the motivation to adopt a completely new approach to teaching?

Non-tenure track instructors are employed because they are good or at least adequate teachers. Adopting new technology in the classroom is risky and could result in lower student evaluations, which in turn could

In this new world of learning technologies there are schools that are content producers and those that are content consumers.

affect their employment status.

The second major threat to adopting the technology is the resources required. A colleague at a small, but well-known private college lamented that his school did not have nearly enough resources to even consider becoming a content producer or adopting new teaching and learning technology. Large public and well-endowed private schools can afford to experiment with technology and pedagogy where schools that are on the edge financially and highly dependent on tuition revenue do not have the resources needed to develop content or infrastructure.

The third threat to adoption comes from university administration; to overcome faculty resistance and to allocate resources to a new venture requires vision and leadership. Universities embrace bureaucracy as the natural way to organize and operate. Large staffs strive to grow larger still; an AAUP analysis based on the IPEDS database showed that from 1975 to 2011 professional staff grew at 16 times the growth of tenured and tenure track faculty. These large staffs treasure the status quo, making change very difficult to bring about in the university. Deans, department chairs and other academic administrators are pulled in many different directions and it can be hard to influence highly independent faculty to innovate.

Faculty governance is the fourth issue inhibiting progress; it exacerbates the difficulties for academic leadership and faculty adoption. The Smith School’s very successful online MBA program was implemented without a faculty vote, a fact that still disturbs many faculty members. A fearless dean decided that the online program was not a change in the curriculum, which requires a faculty vote, but rather a change in format that does not. It is unlikely the program would have survived a faculty vote. Why is that? When you ask the people who are most resistant to change to vote on change, the outcome is fairly predictable. As long as faculty governance procedures in universities require resisters to vote on change, moving ahead with technology-enhanced teaching and learning is going to take very special lead-

^b See <http://bit.ly/2AiXKyx> and <http://bit.ly/2A2nhco>

The biggest threat is to less well known schools with a local reputation lacking much of an endowment and a base of wealthy alumni.

ers who can make a compelling case for change.

Schools that do not overcome these obstacles to the adoption of learning technologies dramatically increase the probability that they will fail in the next 15 years.

Threats from Technology

What schools are threatened the least by technology-enhanced education? The selective privates and state flagship universities are not going to disappear. They could, if they fail to innovate, see declining applications and lower quality students, but such a change will take a long time to be noticed given how sticky academic reputations are. Small colleges with strong reputations, loyal alumni, and some endowment are probably all right, but they need to be careful. Potential students may lose interest if they perceive these schools as too resistant to new ways of teaching and learning.

Smaller colleges with low rankings, low endowments and declining enrollments are not in a position to invest in new technologies and will find themselves falling behind other schools.

These schools will have difficulty in meeting student expectations for technology-enhanced teaching and learning. While the colleges may be able to purchase content, they will be at a disadvantage in creating exciting new student experiences like a MOOC-based degree program. They will lack the knowledge, skilled staff, technology infrastructure, and faculty capabilities to make technology pervasive in the curriculum.

The biggest threat is to less well-known schools with a local reputation lacking much of an endowment and a base of wealthy alumni. These schools are often private and tend to have few resources and few, if any, star faculty. They do not have the resources to innovate and will always have to consume what others produce. With tuition probably higher than the state flagship university and fewer resources, what is their value proposition? They need to develop a niche whether in a particular group of subjects, study abroad programs, work-study options or stress the benefits of a small campus. Even that may not be enough if their enrollments drop.

Is Technology the Answer?

Earlier, technology was presented as a double-edged sword, and the preceding examples show how difficult it is for many schools to adopt it successfully, threatening their very existence. Are universities that become content producers and adopt new learning technologies guaranteed a healthy future? Unfortunately, the answer to this question is “no;” many schools are caught between declining applications and enrollments, declining state aid for publics, increases in tuition, insufficient endowments, increasing costs, and pressures from parents and the government to do something about constantly rising college costs. Schools need to consider many options for reform in addition to the implementation of learning technologies.

The opportunities for reform are many and a number may be found in Lucas.¹ Areas to consider include the tenure system, teaching loads, resources devoted to research, long summer vacations, departments with few students, bloated administrations and bureaucracies, staff sizes, faculty governance, expenditures on physical plant, and the distortions from high-cost varsity football and basketball programs. Universities are complex institutions with many actors and stakeholders. Saving schools that are on the edge will require concerted action across a number of variables to make the system work.

Strategies for Survival

A single strategy does not fit all types

of schools—the private elite and state flagship universities, the small regionals private or public school, and the start-up. A number of recommendations for these three groups may be found in Lucas¹ Exhibit 13-3. Suggested strategies for group 2, the most threatened, include:

- ▶ Building a brand in a niche like global study experiences
- ▶ Creating and joining a network of peer institutions
- ▶ Combining resources on campus and with peers to create technology-enhanced courses.
- ▶ Purchasing content from others
- ▶ Developing very high quality in a few programs
- ▶ Considering the elimination of most research to focus on teaching
- ▶ Increasing teaching loads
- ▶ Reducing staff and overhead
- ▶ Sharing facilities such as computer centers, labs, athletic facilities, and so forth

Concluding Thoughts

In 1873 Disraeli said in the English House of Commons that “A university should be a place of light, of liberty, and of learning.” Whether learning technologies enhance that mission or whether they threaten the university itself depends on how academic leaders and faculty members implement technology-enhanced teaching and learning. ■

Further Reading

Christensen, C., Horn, M., and Johnson, C. *Disrupting Class: How Disruptive Innovation Will Change the Way the World Learns*, N. McGraw-Hill, NY, 2008 and 2011.

Terwiesch, C. and Ulrich, K.T. *Will video kill the classroom star? The threat and opportunity of massively open online course for full-time MBA programs*. Mack Institute for Innovation Management at the Wharton School, University of Pennsylvania, 2014.

Reference

1. Lucas, H.C., Jr. *Technology and the Disruption of Higher Education*. World Scientific Press, Singapore, 2016.

Henry C. Lucas, Jr. (hlucas@rhsmith.umd.edu) is the Robert H. Smith Professor of Information at the Robert H. Smith School of Business, the University of Maryland, College Park, MD.

This article is based on Lucas¹ and the author's experiences teaching traditional, blended, and online courses and a MOOC on Coursera.

Copyright held by author.

Viewpoint

Ask Not What Your Postdoc Can Do for You ...

Seeking more effective strategies for training and nurturing CS postdocs to ensure their success.

THE NUMBER OF postdoctoral fellows in computer science (CS) has risen dramatically in recent years. Studies show that U.S. and Canadian Ph.D.'s taking postdoc positions have tripled since 2000,⁷ and the total number of postdocs in the U.S. rose to new, sustained highs over 2009–2014.⁶ It is now clear that postdocs are a substantial constituency in research-focused university departments.

The Computing Community Consortium (CCC), whose mission is to promote the vitality of computer science research, observed that postdocs in CS now play a much enhanced part in the conduct of research and education at universities. Postdocs are in training positions yet, anecdotally, most departments pay little heed to the training they receive. Certainly, support for postdocs is not on par with the education of graduate students. CCC asked: what are best practices in supporting the computer science postdoc population? An early document by Jones and Gianchandani³ analyzed such best practices and a 2013 *Communications Viewpoint*² raised issues for the CS community more broadly. In the broader context of science and engineering, efforts to improve postdoctoral experience are discussed by Davis.¹

Historically, postdocs have been “invisible”: they work closely with their faculty advisers and that adviser’s group, but have had little standing in the department or university. In essence they



have not been “first-class citizens.” This situation warrants investigating the best ways to support the career and contribution of postdocs. Concerns to address include the quality of training the postdoc receives, the quality of mentoring, the development of new skills, and the participation of postdocs in the community—both in their university department community and in the research community.

To address these concerns and learn more about the postdoc experience, CCC created a program, with NSF’s financial backing, to develop, implement, and institutionalize best practices for supporting postdocs in CS. Competitive awards were made in 2014 to three groups that by design are quite different from one another: a consortium of New York universities (Columbia University, New York University,

the City University of New York, and Cornell University, with 63 postdocs participating since the start of the program) led by Shih-Fu Chang and Julia Hirschberg; a consortium of the three research universities in Arizona led by Chitta Baral and Partha Dasgupta at Arizona State University and including the University of Arizona and Northern Arizona University (47 postdocs since the start of the program); and a single CS department program at the University of Washington (50 postdocs since start) led by Brian Curless. Each group includes evaluators to conduct surveys and focus groups, measuring the effect of the programs implemented. We will refer to the three sites by state name or university in this Viewpoint.

The purpose of this Viewpoint is twofold: to raise awareness right now about the need to serve the often neglected

community of postdocs in CS departments, and to provide insight into what the New York (NY), Arizona (AZ), and Washington (UW) sites are finding to be the best way to implement best practices for their particular situations. The implementation and evaluation of these practices is ongoing; in a future article, we intend to delve into the details of the implementations and the numerical assessments of their impacts. Though our efforts are a work in progress, we believe readers in CS departments will identify with elements they can implement sooner rather than later.

Here, we describe the challenges the three awardees decided were highest priorities to address and the rationales for choosing these challenges, and comment on aspects of the implementation of best practices they are pursuing.

Thoughtful Postdoc Evaluation of Career Goals

Postdoc positions in CS are typically short-term, lasting 1–3 years. The next step for a postdoc spans a wide range of options, including a tenure-track, research, or teaching faculty position, an industrial research lab, a startup. Thus, postdocs should use their training time to carefully evaluate career choices and to develop skills that will serve to further their career. To this end, all sites require postdocs to define and review their personal goals for both their current work and future positions. Both NY and AZ require the use of Individual Development Plans (IDPs) in which the postdoc sets concrete goals for development on a schedule. IDPs help frame conversations with postdoc advisers. NY requires the postdoc to discuss goals identified in his/her IDP with his/her advisor but leaves the submitted document confidential to the postdoc. In contrast, UW has created an online “Review of Progress” system similar to an IDP; the form must be filled out every six months, followed by required meetings with the postdoc’s adviser to discuss goals and progress toward them. The Review form is tracked electronically to ensure compliance, and to identify problems. NY, AZ and UW have all found the written plans to be very effective. NY adopted an existing IDP, designed within the natural science discipline,¹ and has found it to be useful but believes that the scope and

content should be further customized to meet the CS field, a task currently being undertaken by the NY team.

Quality Mentoring

Guidance from experienced mentors is critical to help postdocs stay on track to achieve their goals. Interestingly, each site has a different, complementary approach to encouraging this mentoring. UW emphasizes getting the most out of adviser mentoring using the postdoc’s plan to track and ensure that progress and career goals are discussed regularly with the research adviser. AZ has established non-adviser “Champions,” who are faculty members not directly advising a given postdoc, but who are available to discuss IDPs, postdocs’ concerns, career advice, and so forth. NY provides a mentor pool besides the postdoc’s advisor, comprised of other CS faculty members in the consortium, professionals in the local industry ecosystem, in addition to individual career counseling within their program.

Acquiring Skills

Postdoctoral appointments provide both an opportunity to improve postdocs’ research portfolios and to develop skills for their future careers; these skills will enable them to be more effective on day one of their next jobs. Each program promotes skill development for postdocs. Though in most cases the skill-development programs are aimed at academic careers (a common goal for many postdocs), a number of these skills are transferrable to industry careers. NY has a particularly strong program of frequently offered workshops and panels to develop leadership skills, academic writing, grant writing, teaching, public speaking, and entrepreneurship. They have also offered an extended teaching course comprised of 10 sessions. Leveraging the breadth of their consortium, they are able to utilize resources such as Cornell’s leadership program and Columbia’s writing programs. In two years, NY hosted an extraordinary 66 events, many of which are recorded and available online for review by members of the NY program. AZ offers panels, workshops, and mini-conferences that overlap with NY’s described above, but also include topics such as diversity and ethics. UW offers competitive awards

that fund undergraduates to work on the postdoc’s independent research. The award process offers a kind of academic, on-the-job training. In this program, postdocs define an independent project, write a short NSF-style proposal to fund an undergraduate to perform the research, receive critical reviews from a committee of faculty and past postdoc awardees, receive funding (if accepted) to perform the research, recruit an undergraduate, and mentor the undergraduate through the project. All of this is done independently of the postdoc’s faculty adviser.

Effective Networking

A key part of landing the ideal job and being effective in that job is making oneself known to others in the field—networking. Postdocs benefit from attending conferences and talking to other researchers in academia and industry. To support this goal, each site offers travel awards that supplement any travel (for example, for conference presentations) already supported by advisers. NY additionally offers an annual community networking event with postdoc presenters and guest speakers; these events rotate among the institutions. NY and AZ host well-attended industrial networking events. AZ emphasizes social events among postdocs and industry exposure workshops to encourage in-person interaction with industry leaders.

Belonging

Each site observed that, prior to beginning these projects, a postdoc was hired by a faculty member and vanished into a lab, interacting with few others in the department, before quietly leaving. Now, all three sites sponsor activities to enhance the sense of belonging with the postdoc community and the department community. This “belonging” is important for postdoc satisfaction, breaking down barriers to foster out-of-area interactions, and ultimately for career development. AZ has instituted weekly research presentations by faculty and invited speakers, and has a social time preceding events; it also holds monthly informal lunches with postdocs and faculty to discuss future events and build a greater sense of community. UW has established per postdoc on-boarding orien-

tations, email announcement of each postdoc arrival, department Web-page announcement of any major postdoc accomplishments, monthly postdoc-only lunches, an annual lunch with the department chair, inclusion in all email pertaining to researchers, and encouragement to attend the frequent social activities offered by the department. UW faculty also discuss each postdoc in annual review-of progress meetings, more broadly increasing faculty awareness of both the postdoc community in the department and the individual postdocs. NY hosts quarterly orientations for new postdocs, hosts community building events during National Postdoc Appreciation Week and community excursions (for example, a guided museum exhibit focused on computer science history in NY), in addition to the annual postdoc symposium rotated over consortium campuses for postdocs to present their research to other postdocs and faculty.

Institutional Support

Implementing and sustaining postdoc best practices requires a commitment from faculty and staff and support from the university as a whole; each of the sites funds personnel to run their programs. NY has a full-time staff member devoted to the effort, and AZ and UW each have a half-time staff member. NY's requirement has been greater due to the larger size of its multi-institutional effort and the greater organization and coordination required by the many workshops and other events it runs. UW and AZ also devote individual faculty time to engaging with the postdocs. Having faculty and staff support has been essential to developing sound programs to nurture postdocs appropriately, but such support is often difficult to fund outside of the programs described here, which are currently (but not indefinitely) supported by CCC. At AZ, support and participation from the graduate college and the Science Foundation of Arizona has helped in many events and has shown a path toward sustainability beyond the grant period. At the same time, taking inspiration from their CS program, AZ State University's graduate college is developing a general program for all its postdocs.

Conclusion ...

Our efforts are a work in progress, and we are continuing to evaluate the impact of our respective implementations of best practices. After completing their efforts, the three sites will collectively recommend a suite of effective strategies for training and nurturing CS postdocs, to ensure their success; these recommendations will appear in a follow-on article that provides greater depth about the programs, discussion of pros and cons of practices tried, and measurements of the impacts of the efforts. We do not expect a one-size-fits-all implementation of best practices. Each CS department has different strengths, challenges, and resources that will define the steps that they can realistically take. But it is clear to us that by taking a few actions, the support for postdocs can be enriched greatly. This observation is reinforced by the findings of a recent study by the National Academies of Sciences, Engineering and Medicine.⁵ Because postdocs now play a major role in the research activity of many departments, the quality of the work of those postdocs is a material determinant of the quality of the research of a department. Postdocs are not just another group to train; they are taking major responsibility in running labs, mentoring graduate and undergraduate students and even teaching.

... and a question

We conclude with an important question: How sustainable are these efforts? At a minimum, what we do to support graduate students can often support our postdocs as well. Both are at a somewhat similar stages in their careers. But such programs require dedicated staff, and such staff are expensive. Research awards and travel grants also have costs.

NY and AZ began their program by teaming with offices that support postdocs across the university. Indeed, much of the skill development support could be shared across postdocs in many science and engineering disciplines, yet still be of high quality for computer science postdocs. Individual development plans might likewise be common across "like" disciplines and their administration—to a great extent—supported by a school or the university, rather than by the department's lesser resources.

However, while some faculty advising and postdoc evaluation could be interdisciplinary, much of it needs to be specific to computer science. For example, computer science departments can take advantage of the vibrant research labs in industry to assemble a diverse mentor pool for CS postdocs, and topics covered in the IDP can be customized to meet the unique needs of the CS field. Creation of a faculty "postdoc committee" assignment is straightforward, but the faculty member(s) must be proactive in engaging the postdocs for the assignment to be of any value. Daily interaction with others in the department—students and faculty and postdocs—is also necessary for the postdoc to have a quality training experience.

Determining an effective division of labor and costs between the department and the larger university organizations will be key to creating sustainable programs for supporting a critical resource, our CS postdocs.

We urge all universities, departments, and faculty to consider: "Ask not what your postdoc can do for you, but what you can do for your postdoc ... to grow and advance toward a successful career." ■

References

1. Davis, G. Improving the postdoctoral experience: An empirical approach. *Science and Engineering Careers in the United States: An Analysis of Markets and Employment*. R.B. Freeman and D.L. Goroff, Eds., University of Chicago Press, 2009.
2. Jones, A. The explosive growth of postdocs in computer science. *Commun. ACM* 56, 2 (Feb. 2013), 37–39.
3. Jones, A. and Gianchandani, E. *Computer Science Postdocs—Best Practices*. 2012; <http://bit.ly/2mInIFH>
4. myIDP; <http://bit.ly/2jOziCB>
5. National Research Council. *The Postdoctoral Experience Revisited*. 2014.
6. NSF NCSSES data; <http://bit.ly/2j0m790>
7. The Taulbee Survey; <http://bit.ly/2ANbDRP>

Chitta Baral (chitta@asu.edu) is a Professor of Computer Science at Arizona State University.

Shih-Fu Chang (sfchang@cs.columbia.edu) is Senior Executive Vice Dean, School of Engineering and Applied Science, and Professor in the Electrical Engineering Department and the Computer Science Department at Columbia University.

Brian Curless (curless@cs.washington.edu) is a Professor in the Paul G. Allen School of Computer Science & Engineering at the University of Washington.

Partha Dasgupta (partha@asu.edu) is an Associate Professor of Computer Science at Arizona State University.

Julia Hirschberg (julia@cs.columbia.edu) is Percy K. and Vida L. W. Hudson Professor of Computer Science and Chair of the Computer Science Department at Columbia University.

Anita Jones (jones@virginia.edu) is University Professor Emerita at the University of Virginia.

Copyright held by authors.

Introducing *ACM Transactions on Human-Robot Interaction*

Now accepting submissions to ACM THRI

In January 2018, the *Journal of Human-Robot Interaction* (JHRI) will become an ACM publication and be rebranded as the *ACM Transactions on Human-Robot Interaction* (THRI).

Founded in 2012, the *Journal of HRI* has been serving as the premier peer-reviewed interdisciplinary journal in the field.

Since that time, the human-robot interaction field has experienced substantial growth. Research findings at the intersection of robotics, human-computer interaction, artificial intelligence, haptics, and natural language processing have been responsible for important discoveries and breakthrough technologies across many industries.

THRI now joins the ACM portfolio of highly respected journals. It will continue to be open access, fostering the widest possible readership of HRI research and information. All issues will be available on the ACM Digital Library.

Editors-in-Chief Odest Chadwicke Jenkins of the University of Michigan and Selma Šabanović of Indiana University plan to expand the scope of the publication, adding a new section on mechanical HRI to the existing sections on computational, social/behavioral, and design-related scholarship in HRI.

The inaugural issue of the rebranded *ACM Transactions on Human-Robot Interaction* is planned for March 2018.

To submit, go to <https://mc.manuscriptcentral.com/thri>



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

The network era requires new models, with interactions instead of algorithms.

BY ANTONY ALAPPATT

Network Applications Are Interactive

THE PROLIFERATION OF mobile devices and the interconnectivity between them has created new application opportunities. These new applications are no longer limited to a single system space but are spread across many system spaces. This shift of application from single system-spaced, host-based systems to multisystem-spaced solutions is being hampered by software toolsets that are stuck in the older sequential computational models.

This article defines this new requirement as a network application. It argues how current programming based on computation is not the right model and suggests that application development has to move from software based on a computational premise to a communication or interaction premise. It closes with a simple CRUD (create, read, update,

delete)-based application developed with interactions to illustrate the point, laying out the challenges and explaining how such a system will not have algorithms at its center but will work by using interactions.

Imagine a patient-doctor encounter in the future, where the patient, who has a personal electronic medical records (EMR) device, is identified to the doctor's office either by a near-field communication (NFC) wave or a physical scanning of the EMR device into a receptacle. The EMR device talks to the doctor's office system, providing all the necessary information. Prescriptions could be sent directly to the pharmacy, with a sync point given to the EMR. Thus, when the patient goes to the pharmacy, the medicine is delivered promptly.

This is the network era, requiring new approaches to software. Host-centric programming that limits itself to single system space will not be sufficient for the multisystem-space needs of this new digital era.

What Is a Network-Based Application?

Applications are of two types: stand-alone and networked.^{2,9} Applications such as word processing on PCs, multiuser programming on a mainframe, or distributed computing using Simple Object Access Protocol (SOAP) or object request broker (ORB) are stand-alone. Stand-alone applications provide a single machine view⁹ to the application programmer by having a single entry point (that is, the calling program hands over control to the application and takes back control after the application is complete). The application accesses data from the environment by executing I/O routines that runs parallel to the main routine. The program counter in the machine controls the program. The stand-alone application runs on top of the machine (virtual or real).

In the example EMR application, notice how the programmer does not view the application as one hierarchical pro-



gram but as interactions by different people that take place across a network of devices. Unlike a stand-alone application, a network application advances when the different agents interact with each other through their actions. These actions accept application state changes from agents on the network and also affect application state changes on other agents on the network. The network-based application runs on top of the network infrastructure.

Network Application Is Not Computation

The modern digital era requires people and devices all acting in unison to create the overall business experience. All these different computational agents sitting on different system spaces should communicate with each other. To put it simply, they should ask and tell each other to create the whole solution. This digital-era choreography of events happens simultaneously and, hence, is a concurrent problem.

Programming consists of statements separated by a sequential operator (as in Java) or an assignment operator. These operators assign the value of the expressions into a memory location and also instruct the compiler to move to the next instruction. These two fun-

damental constructs of the language make programming very sequential. Programming in a concurrent world, with sequential constraints, places a big challenge on the programmer.

Generally, there are two main parts that make a program work. One is the control of the program, and the other is what data is transferred when the control moves forward. The control is the cursor where the program is operating. In sequential programming, the control moves from top to bottom. The only ways to alter program control are by using go-to statements; exception statements; and iteration/looping statements. Programming consists of preparing the data so the processor can process the information. The creative part of software is established by manipulating program control. Manipulating hierarchical programs to represent all sorts of models is difficult to achieve and, hence, leads to bugs.

In sequential programs, program control is assumed to move forward. The language itself does not have any facilities to handle processing across system spaces. If part of the execution is sitting in another system space, then how is the issue of control handled? How will the language have enough concepts to handle the different issues

of multisystem-space computing such as (a) transferring control; (b) handling latency; and (c) dealing with exceptions? How will one piece of code tell another piece of code sitting in a different system space that it has continued successfully or has thrown an exception. Nothing in these languages supports such issues. They assume there is one system space and that the control in the processor will move forward to the next step.

Digital-era solutions must coordinate people and devices as peer-to-peer collaborating systems. Creating such collaborating systems from sequential programs is a big challenge. Is there a better way so that these concurrent systems can be expressed more directly to make digital solutions easier? Toward this end, the basic default sequential control of programs needs to be eliminated. The default sequential control makes expressing concurrent problems more difficult, as the programmer has to manipulate the sequential control to create concurrency.

One of the biggest issues is sharing state across system spaces. In the current programming paradigm, state is examined using functions and variables. These are available only within the constraints of the language and are not exposed outside of the operating language. To overcome this, the language itself should have facilities to share state (both program control and data) so that the client can examine it.

In some situations the application is aware of the location of the data but is not sure when all the data arrives. This makes the current deterministic nature of programs unsuitable. This nondeterministic nature of data in the digital era contributes to the inability of current programming to solve digital-era problems. Although fundamental programming does not have a way of dealing with nondeterministic systems, it is now achieved by following a new programming paradigm called reactive programming.^{3,8}

Once state and control of a program are shared, is it possible to ensure that only authorized people have access to both of these crucial elements? Current programming languages do not have the concept of hiding information. Ideally, programming languages should have facilities

Figure 1. Computation is communication.

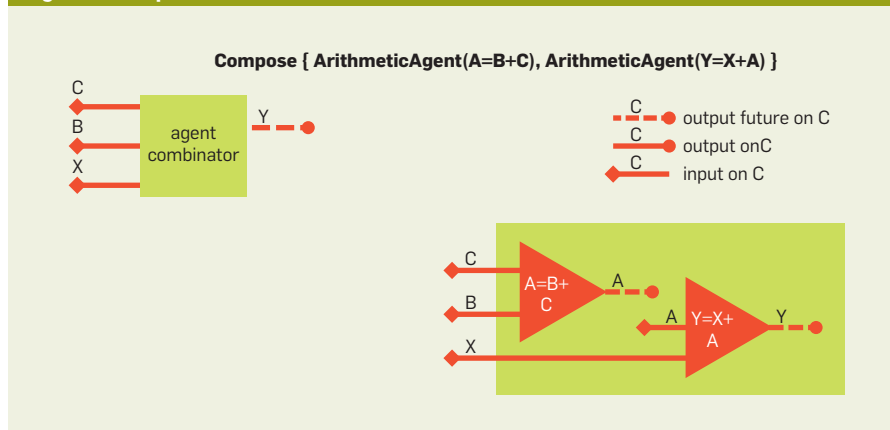


Figure 2. "Hello, World" application.

<code>#include <stdio.h></code>	include information about standard library
<code>main()</code>	define a function named main that receives no argument values
<code>{</code>	statements of main are enclosed in braces
<code>printf("hello, world\n");</code>	main calls library function <code>printf</code> to print this sequence of characters; <code>\n</code> represents the newline character
<code>}</code>	

to share state and provide controlled access to this selectively.

A network application that can work across system spaces needs a new way of looking at the computation problem. Rather than having *computation* ideas (procedure, functions, among others) as fundamental, it should be the *communication* ideas that are fundamental for a good network application programming paradigm.


Computation Is Communication

Consider two arithmetic expressions: $Y=X+A$ and $A=B+C$. With the sequential programs now in use, these are written sequentially as $\{ A=B+C; Y=X+A \}$. The programs operate sequentially to get the answer. If they are accidentally written as $\{ Y=X+A; A=B+C \}$, the program will continue to operate, but the answer will be wrong (i.e., a bug). Is there a way to eliminate this anomaly?


Imagine it a bit differently. In the same computation, the expressions run concurrently and combine when A is shared—that is, the composition consists of two agents, $A=B+C$ and $Y=X+A$, which operate concurrently (Figure 1). The right-hand side of an arithmetic expression is a *sink* of the value and the left-hand side of the expression is a source of information. So in the case of $A=B+C$, B and C are sinks and A is the source. $A=B+C$ is resolved when B and C arrive and raise A . Meanwhile, $Y=X+A$ is waiting on X and A . Assuming X has arrived, when the first computation raises A , it is consumed automatically by the second computation to raise the answer Y .

Notice in this case the agents run concurrently, but because the second agent is waiting on A , that is raised by the first agent. The two computations are concurrent. The two agents are combined but they expose the combination actions (i.e., they accept C , B , and X and produce Y). This eliminates the propensity of bugs caused by the out-of-order coding in sequential languages.

The whole computation is running in parallel and drives itself, depending on the arrival of the values. The computation is no longer about algorithms but is about communicating values between two active computation agents. The flow of data is accomplished by naming the two variables.



Digital-era solutions must coordinate people and devices as peer-to-peer collaborating systems. Creating such collaborating systems from sequential programs is a big challenge.



Thus, to flow Y to another agent, all that has to be done is to combine AgentCombinator with the arithmetic agent that is going to consume Y . To define this communication, the control is established by splitting the communication into two pieces: one representing the *giving* of information and the other representing the *receiving* of information.

Making the computation concurrent addresses the speed of the computation, eliminates the issue of bugs when the computation is coded out of order, and makes the computation networked and nondeterministic. Clearly, computation is better expressed as communication.

Network Application Is Communication

Unlike a stand-alone application, a network application does not have a single machine view. The network application is a swarm of many machines operating concurrently, also known as agents. These agents, residing in single or multiple system spaces, coordinate their work to create an experience. Each agent could be acting as a client in one instance and a server in another.

Whether stand-alone or networked, a coherent application has certain salient properties.

Application control. It is the application control that sets the rhythm of the program. Once the program is given control from the operating system (through the `main()` function in C) or from a Web server in the case of a service, the programmer manages program control by applying different control statements provided by the employed language. After the program is completed, control is handed over to the operating system or the service initiator.

The magic of software is accomplished by controlling how these instructions are organized according to the language rules. For the sake of illustration, consider the C language.⁴ The only control elements that are given to the programmer are statements of functions, assignments, and looping statements. The programmer uses these constructs to build the application flow (Figure 2).

Once a stand-alone application in C has received control, the program

Figure 3. “Hello, World” application as functions.

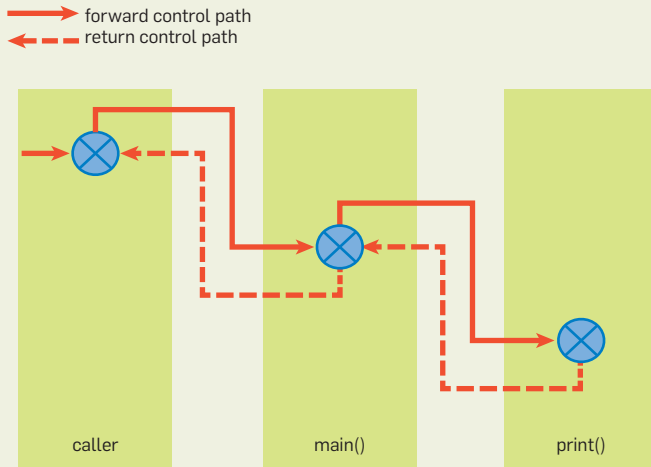
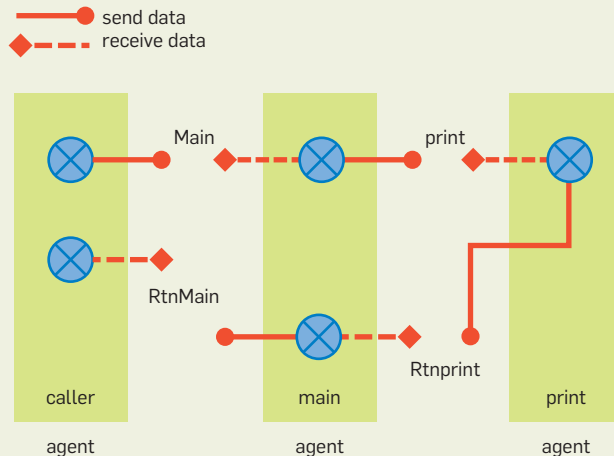


Figure 4. “Hello, World” application as communication.



interacts with the outside world by running functions to do the input/output (Figure 3). During the I/O statements, the program is blocked. This is becoming a multisystem-spaced world in which the expectation is that the state can be observed by another program. The program and its languages should have notation and concepts to share data dynamically at runtime, with no additional engineering.

A stand-alone application control has two elements: the forward and return movement of control, and the transfer of data during these movements. In the current programming model, because of the limitation of the state machine of the processor (with its current program counter), the forward and the return movements of control

and data are synchronous (that is, the caller halts until the call control is handed back to the caller).

Now fast forward to the futuristic patient-doctor encounter mentioned at the beginning of this article. In this instance, the different devices, also known as agents, talk to each other to move the process forward. It is not a single-host system but a collection of distributed devices talking to each other, making a complete, whole self. In this new world, there are no statements or assignments—only a series of interactions, or sync points. How do you express this new world with no single point of control? How do you represent this new world with no obvious program control in the traditional sense?

To make this possible, let’s explore

the same stand-alone application and try to achieve the same control flow by delivering it as a network application. To network the example application, the three components—`caller`, `main`, and `print`—operate as three independent agents (Figure 4). Connecting these agents allows them to deliver the same print function in a networked setting. The caller agent kicks off the computation by sending data on the main action and waiting on `rtnMain` action. The main agent consists of two independent agents, one listening on the main action and the other listening on `rtnPrint`. The first component agent that receives the main action turns on **print**. Meanwhile, the one print agent listens on print and after printing on screen, turns on `rtnPrint`. The other main agent responds to `rtnPrint` and turns on `rtnMain`. Notice that each agent operates independently but is coordinated by the different actions or sync points. As `rtnMain` is triggered only after the print is completed, the functionality is the same as in the first C program. The difference is how the functionality is achieved through coordination of the autonomous agents: `CallerAgent`, `MainAgent`, and `PrintAgent`. These agents could also work across multiple system spaces.

Moving the application control from computation to communication makes applications work coherently across multiple system spaces.

Latency. As mentioned earlier, the simple C program that was a stand-alone application is now expressed as a network application by moving it from a premise of computation to one of communication. The sync points such as `main`, `rtnMain`, `print`, and `rtnPrint` coordinate these agents to create a coherent whole. These coordination elements could sit either in a single system space or across multiple system spaces. If these sync points sit across address spaces, then this introduces a new constraint: the latency of the network.

This now identifies the speed of the whole application. In a typical network application, latency is reduced when the application does not use the network. By introducing caching, the network usage is reduced, thereby increasing the speed of the overall application.

Scoping. Information hiding is an important property of a computer system. Programming languages should support this very well. To illustrate, object-oriented programming lets the programmer define the visibility of information to be either private or public. When private, information is visible only to the object. If the variable is declared as public, the information is visible to the whole program. These declarations are instructions to the compiler to control information visibility. The runtime information visibility has to be done by the programmer during design and construction. Having these kinds of information hiding in memory is fine for stand-alone applications, but with network applications, the language also should support scoping across the network. How is this accomplished?

In a network application, state transitions are exposed as sync points. In addition, information is transferred through the sync points. Clients can influence the application by interacting with these sync points. In Figure 5, the server has two sync points: pa1, which is hidden so it cannot be observed by the clients, and pa2, which is open and can be observed by all clients. In this configuration, only client1 is aware of pa1 and, hence, the server can be influenced only by client1. In terms of object-oriented principles, pa1 is private and pa2 is public. This principle is used to control information visibility across the network. Scope is controlled by hiding the action from the public and then giving the sync name selectively to different agents. Each of these clients and servers could exist across many address spac-

es, creating controlled information security over the network.

Network applications bring new challenges such as multisystem spaces, latency, intermittent network availability, and security. Thinking of applications as communication rather than as functions overcomes these challenges.

Mathematics of Communication Systems

Network applications should be developed with toolsets and languages that have communication as their foundation. Software based on computation has a good theoretical foundation, such as λ -calculus, on which the whole concept of programming is built. An application based on λ -calculus shuts out the world when the program executes, whereas a network application, by its very nature, has many agents interacting with each other on a network. This difference in the nature of the applications calls for a new foundation for the toolsets and languages for building network applications.

π -calculus,^{6,7} the theory of communicating and mobile systems, is the algebraic representation of a Petri net. The focus in π -calculus is to define the communication between two agents that are operating on a network. It also can define how configurations of agents on the network change as they interact. This property of π -calculus—the ability to model interactions between many agents operating independently—makes it an ideal mathematical foundation for building toolsets and languages for network applications.

π -calculus achieves this by: (a) representing agent behavior as actions; (b)

configuring the different agents as the initial starting configuration; (c) representing shared transition between these agents; and (d) changing the configuration of the agent network as state transitions happen. The π -calculus agent actions can be observed. An action represented by x is the potential to receive information. The action represented by \bar{x} is the potential to be the source of information. The reaction of these complementary actions moves the process and the data from source to destination.

Figure 6 details initial configuration P. This is taken from page 88 of Milner’s *Communicating and Mobile Systems: the Pi-Calculus*.⁶ These terms run concurrently with each other. The parallel operation is identified by the operator “|”. The sequence operation is represented by “.”. The different actions are x and u . The action x carries through u . The action u transmits through v .

The π -calculus engine does interactions (that is, it looks at the action with the same name and executes the reaction). When the reaction happens, the value is communicated from one term to the other. In the case of the reaction between x and \bar{x} , it will send z through x . With the reaction, the second term becomes inert, represented by \emptyset —a term with no actions. Then, as shown in Figure 7, it proceeds to substitute u with z to raise the final configuration process state P1. Notice how the different term nodes collapse to create the final state.

It is possible to move programming from computation to communication—network-centric programming—by designing a language that is founded on π -calculus (such as, on interactions) and running it on a reaction engine that executes the language as reactions.

Figure 5. Scoping as communication.

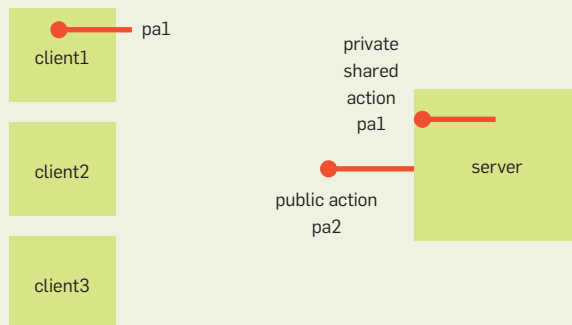


Figure 6. Initial processing configuration.

$$P = x(u).\bar{u} \langle v \rangle \mid x \langle z \rangle$$

Figure 7. Final process configuration.

$$P1 = \bar{z} \langle v \rangle \mid \emptyset$$

Figure 8. Interactions trace.

Current Interaction - Reaction between +ve and -ve	Active Ask Actions (-ve)	Active Tell Actions (+ve)
initial configuration	main, rtnPrint, print	–
on interacting with main	rtnPrint, print	print
on interacting with print	rtnPrint	rtnPrint
on interacting with rtnPrint	–	rtnMain

Figure 9. Action definition.

```
<Actions>
  <Action>
    <Element>FirstName</Element>
    <Element>LastName</Element>
    <Element>EmailId</Element>
    <ActionName>View</ActionName>
  </Action>
  <Action>
    <Element>FirstName</Element>
    <Element>LastName</Element>
    <Element>MobileNumber</Element>
    <Element>EmailId</Element>
    <ActionName>Delete</ActionName>
  </Action>
  <Action>
    <Element>FirstName</Element>
    <Element>LastName</Element>
    <Element>MobileNumber</Element>
    <Element>EmailId</Element>
    <ActionName>Edit</ActionName>
  </Action>
  <Action>
    <Element>FirstName</Element>
    <Element>LastName</Element>
    <Element>MobileNumber</Element>
    <Element>EmailId</Element>
    <ActionName>Add</ActionName>
  </Action>
</Actions>
```

Hello, World Application as Interaction

Network applications are best addressed when based on communication. Unlike the functional model of computation (which has functions to move data from one variable to another) in a network application the movement of data is done by communication. As in physics, the flow of current is defined by the potential. Similarly, network applications are defined by setting up communication potential and are then wired together to create the spark that ensures the flow of information from the source to the sink.

The first step in building a network application is to define what flows when this spark happens. To de-

fine this action then defines which elements flow through it. Once the action is defined, then the two potentials for this action are set up by creating the +ve and -ve. The +ve is the source of information (like x , in π -calculus), and the -ve (like x in π -calculus) is the sink of information. Tell sets up the +ve source of information, and Ask sets up the -ve sink of information.

The Hello, World network application has two agents that operate concurrently to deliver the functionality. MainAgent is composed of two sequences of two actions each. When this agent is activated, it activates main and rtnPrint simultaneously:

```
MainAgent = Compose [
  Sequence [ Ask(main);
  Tell(print)]; Sequence [
  Ask(rtnPrint);
  Tell(rtnMain)]
]
```

The second agent is PrintAgent:

```
PrintAgent = Sequence [
  Ask(print); Tell(rtnPrint) ]
```

The Hello, World application consists of these two agents operating concurrently. It is defined as

```
HelloWorldApplication =
  Compose [ MainAgent;
  PrintAgent]
```

Figure 8 shows the different actions that are on when each action is interacted with. To begin with, all agents become active and the actions main, rtnPrint, and print are exposed on the Ask side (-ve). The main action would be exposed, so it can be interacted with from a user interface. When main is interacted with, it activates print on the Tell side (+ve). The same name, print, is now active

on both the +ve and -ve sides, setting up the situation for a spark. Next the interaction happens on print opening up rtnPrint on the Tell side (+ve). This leads to interaction with rtnPrint, leading to the next set of interactions. The interactions' trace is shown in Figure 8.

These actions are represented as REST URIs (uniform resource identifiers), so they can be interacted with over the network. Network programming is as simple as bringing agents into configurations with the objective of precipitating these interactions.

A Simple CRUD Application as Interaction

MasterKube software technology⁵ is built with this network application in mind. The MasterKube software paradigm is used to build a simple CRUD application.

Define action. The first step is to define all actions that will be exposed to the outside system. These can be observed by any client UI that understands the MasterKube XML response. In this case there are actions to Add, Edit, View, and Delete, as illustrated in Figure 9.

Define behavior. Agents provide the dynamic part of the application. The agent AddContactAgent in the following code shows the action Add, which, when interacted with, creates a new agent. The AddContactAgent shows the Add UI. On interacting with the Add action, it creates the Contact, and AddContactAgent is repeated to show the same action (see Figure 10).

When there is interactions with Edit, it shows with all the elements that are in the Edit action. These are expected to be entered. On accepting these values into the system, EditContactAgent is repeated. Repeating the same agent gives the choice of the Edit, Delete, and View actions again, thereby exhibiting a persistent behavior.

In the Delete action the Choice action is empty. As there are no observations, interaction with Delete leaves no further observations and, hence, the contact is considered deleted.

All these actions are exposed as REST URLs, which can be interpreted by any device.

Figure 10. Agent definition.

```

<Agent>
<AgentName>AddContactAgent</AgentName>
<Sequence>
  <!--Triggers the contact agent when add action is pressed -->
  <AgentCommand> <AgentName>ContactAgent</AgentName>
  <Ask><action><actionName>Add</actionName></action></Ask>
  </AgentCommand>
  <!-- Because this agent is in a sequence operation,
  the same agent is called recursively -->
  <AgentCommand> <AgentName>AddContactAgent</AgentName> </AgentCommand>
</Sequence>
</Agent>

```

The ContactAgent accepts the FirstName, LastName, MobileNumber, and EmailId. It activates the EditContactAgent. The AvatarName, which is how the agent is identified, is the LastName.

```

<Agent>
<AgentName>ContactAgent</AgentName>
<ProcessName>Contacts</ProcessName>
<AvatarName>LastName</AvatarName>
<Element>FirstName</Element> <Element>LastName</
Element><Element>MobileNumber</Element><Element>EmailId</Element>
<Compose>
  <!-- Repeatedly show the edit contact agent till the user presses Delete
  -->
  <AgentCommand><AgentName>EditContactAgent</AgentName></AgentCommand>
  </Compose>
</Agent>

```

The EditContactAgent shows the different actions shown by the contact. The different actions that are enabled for each contact are Edit, Delete, and View. These actions are shown in a choice, as only one action can be taken. Upon interacting with one option, the other actions go away.

```

<Agent>
<AgentName>EditContactAgent</AgentName>
<Compose>
  <!--Two choices are presented. One is to Edit and the other is to De-
  lete-->
  <Choice> <ChoiceOption>
    <Ask><action><actionName>Edit</actionName></action></Ask>
    <OptionAction>
    <Compose>
      <!--When edit is interacted with the same agent
      is repeated to show persistence -->
      <AgentCommand> <AgentName>EditContactAgent</AgentName> </AgentCommand>
    </Compose> </OptionAction>
  </ChoiceOption>
  <ChoiceOption>
    <Ask><action><actionName>Delete</actionName></action></Ask>
    <OptionAction> <!--When delete is interacted with no further
    observations are exposed so this contact disappears. --> </OptionAc-
  tion>
  </ChoiceOption>
  <ChoiceOption>
    <Tell><action><actionName>View</actionName></action></Tell>
    <OptionAction>
      <!--When View is interacted with
      the same agent is repeated to show persistence -->
      <AgentCommand><AgentName>EditContactAgent</AgentName></AgentCommand></
    OptionAction>
  </ChoiceOption></Choice>
</Compose>
</Agent>

```

Conclusion

The miniaturization of devices and the prolific interconnectedness of these devices over high-speed wireless networks are completely changing how commerce is conducted. These changes (a.k.a. digital) will profoundly change how enterprises operate. Software is at the heart of this digital world, but the software toolsets and languages were conceived for the host-based era. The issues that already plague software practice (such as high defects, poor software productivity, information vulnerability, and poor software project success rates) will be more profound with such an approach. It is time for software to be made simpler, secure, and reliable.

Moving software from its computing foundation to a communication foundation will ensure the promise and benefits of digital are more widely shared. □

Related articles on queue.acm.org

A Guided Tour through Datacenter Networking

Dennis Abts and Bob Felderman
<http://queue.acm.org/detail.cfm?id=2208919>

You Don't Know Jack about Network Performance

Kevin Fall and Steve McCanne
<http://queue.acm.org/detail.cfm?id=1066069>

The Network's NEW Role

Taf Anthias and Krishna Sankar
<http://queue.acm.org/detail.cfm?id=1142069>

References

- Backus, J. Can programming be liberated from the von Neumann style? A functional style and its algebra of programs. *Commun. ACM* 21, 8 (Aug. 1978), 613–641; <http://dl.acm.org/citation.cfm?doi=359576.359579>.
- Fielding, R.T. Architectural styles and the design of network-based software architectures. Ph.D. dissertation, 2000, University of California, Irvine; <https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.
- Futures and promises. The Boost C++ Libraries; <http://theboostcpplibraries.com/boost.thread-futures-and-promises>.
- Kernighan, B., Ritchie, D.M. *The C Programming Language*. Prentice Hall, 1978.
- MasterKube Software Development Manual. 2014
- Milner, R. *Communicating and Mobile Systems: The Pi-Calculus*. Cambridge University Press, 1999.
- Milner, R. Elements of Interaction. *Commun. ACM* 36 1 (Jan. 1993), 78–89; <http://dl.acm.org/citation.cfm?id=1283948>.
- ReactiveX; <http://reactivex.io/intro.html>.
- Tanenbaum, A.S., van Renesse, R. Distributed operating systems. *ACM Computing Surveys* 17, 4 (1985), 419–470.

Antony Alappatt (antony.alappatt@masterkube.com) is the founder of MasterKube, whose goal is to provide path-breaking software technology and services to serve the new digital era.

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

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

Ordinary users need tools that automate the selection of custom-tailored faults to inject.

BY PETER ALVARO AND SEVERINE TYMON

Abstracting the Geniuses Away from Failure Testing

THE HETEROGENEITY, COMPLEXITY, and scale of cloud applications make verification of their fault tolerance properties challenging. Companies are moving away from formal methods and toward large-scale testing in which components are deliberately compromised to identify weaknesses in the software. For example, techniques such as Jepsen apply fault-injection testing to distributed data stores, and Chaos Engineering performs fault injection experiments on production systems, often on live traffic. Both approaches have captured the attention of industry and academia alike.

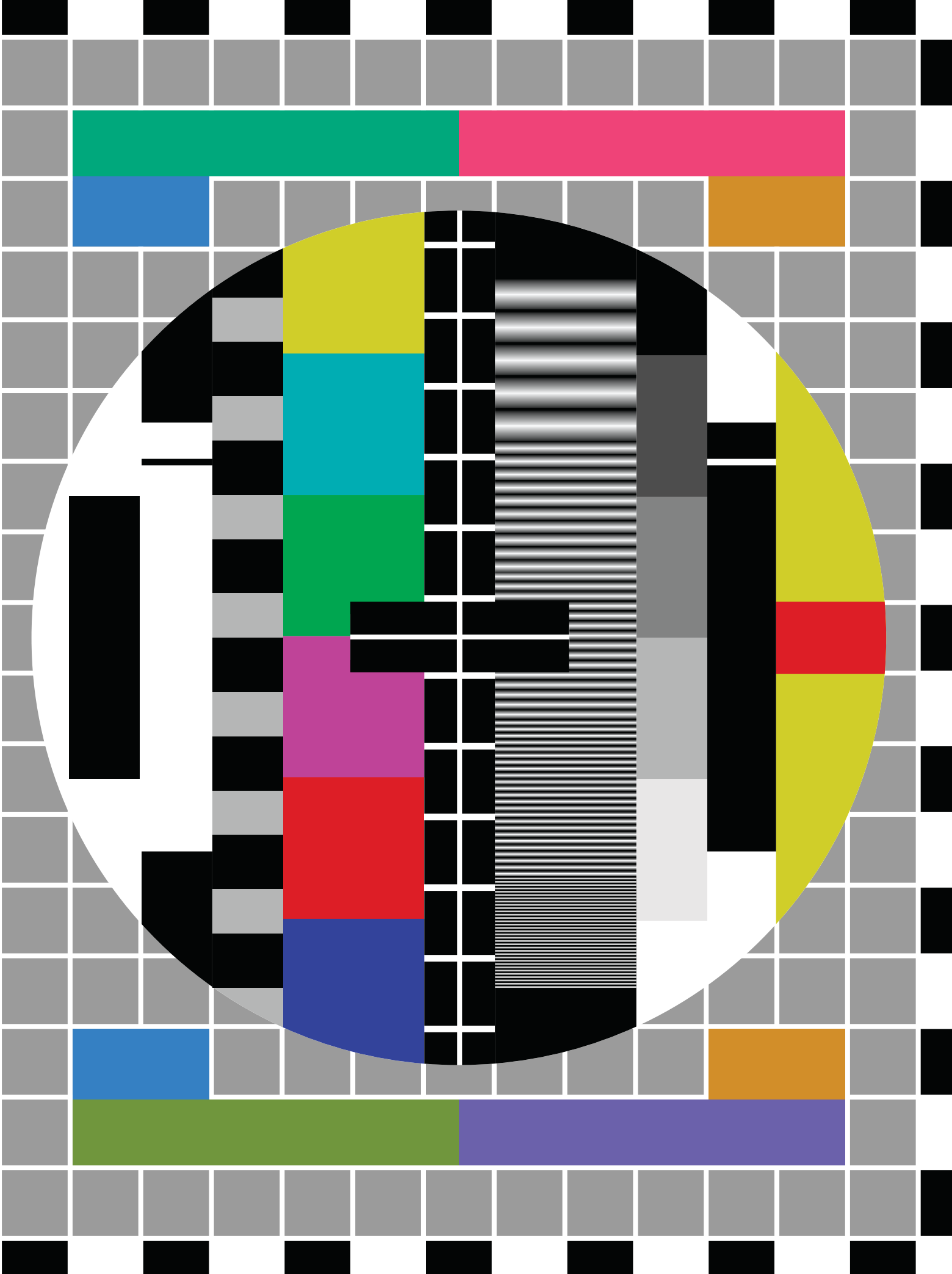
Unfortunately, the search space of distinct fault combinations that an infrastructure can test is intractable. Existing failure-testing solutions require skilled and intelligent users who can supply the faults to inject. These superusers, known as Chaos Engineers

and Jepsen experts, must study the systems under test, observe system executions, and then formulate hypotheses about which faults are most likely to expose real system-design flaws. This approach is fundamentally unscalable and unprincipled. It relies on the superuser's ability to interpret how a distributed system employs redundancy to mask or ameliorate faults and, moreover, the ability to recognize the insufficiencies in those redundancies—in other words, human genius.

This article presents a call to arms for the distributed systems research community to improve the state of the art in fault tolerance testing. Ordinary users need tools that automate the selection of custom-tailored faults to inject. We conjecture that the process by which superusers select experiments—observing executions, constructing models of system redundancy, and identifying weaknesses in the models—can be effectively modeled in software. The article describes a prototype validating this conjecture, presents early results from the lab and the field, and identifies new research directions that can make this vision a reality.

The Future Is Disorder

Providing an “always-on” experience for users and customers means that distributed software must be *fault tolerant*—that is to say, it must be written to anticipate, detect, and either mask or gracefully handle the effects of fault events such as hardware failures and network partitions. Writing fault-tolerant software—whether for distributed data management systems involving the interaction of a handful of physical machines, or for Web applications involving the cooperation of tens of thousands—remains extremely difficult. While the state of the art in verification and program analysis continues to evolve in the academic world, the industry is moving very much in the opposite direction: away from formal methods (however, with some noteworthy exceptions,⁴¹) and toward



approaches that combine testing with fault injection.

Here, we describe the underlying causes of this trend, why it has been successful so far, and why it is doomed to fail in its current practice.

The Old Gods. *The ancient myth: Leave it to the experts.* Once upon a time, distributed systems researchers and practitioners were confident that the responsibility for addressing the problem of fault tolerance could be relegated to a small priesthood of experts. Protocols for failure detection, recovery, reliable communication, consensus, and replication could be implemented once and hidden away in libraries, ready for use by the layfolk.

This has been a reasonable dream. After all, abstraction is the best tool for overcoming complexity in computer science, and composing reliable systems from unreliable components is fundamental to classical system design.³³ Reliability techniques such as process pairs¹⁸ and RAID⁴⁵ demonstrate that partial failure can, in certain cases, be handled at the lowest levels of a system and successfully masked from applications.

Unfortunately, these approaches rely on failure detection. Perfect failure detectors are impossible to implement in a distributed system,^{9,15} in which it is impossible to distinguish between delay and failure. Attempts to mask the fundamental uncertainty arising from partial failure in a distributed system—for example, RPC (remote procedure calls⁸) and NFS (network file system⁴⁹)—have met (famously) with difficulties. Despite the broad consensus that these attempts are failed abstractions,²⁸ in the absence of better abstractions, people continue to rely on them to the consternation of developers, operators, and users.

In a distributed system—that is, a system of loosely coupled components interacting via messages—the failure of a component is only ever manifested as the *absence of a message*. The only way to detect the absence of a message is via a timeout, an ambiguous signal that means either the message will never come or that it merely has not come yet. Timeouts are an end-to-end concern^{28,48} that must ultimately be managed by the application. Hence, partial failures in distributed systems bubble



While the state of the art in verification and program analysis continues to evolve in the academic world, the industry is moving in the opposite direction: away from formal methods and toward approaches that combine testing with fault injection.



up the stack and frustrate any attempts at abstraction.

The Old Guard. *The modern myth: Formally verified distributed components.* If we cannot rely on geniuses to hide the specter of partial failure, the next best hope is to face it head on, armed with tools. Until quite recently, many of us (academics in particular) looked to formal methods such as model checking^{16,20,29,39,40,53,54} to assist “mere mortal” programmers in writing distributed code that upholds its guarantees despite pervasive uncertainty in distributed executions. It is not reasonable to exhaustively search the state space of large-scale systems (one cannot, for example, model check Netflix), but the hope is that modularity and composition (the next best tools for conquering complexity) can be brought to bear. If individual distributed components could be formally verified and combined into systems in a way that preserved their guarantees, then global fault tolerance could be obtained via composition of local fault tolerance.

Unfortunately, this, too, is a pipe dream. Most model checkers require a formal specification; most real-world systems have none (or have not had one since the design phase, many versions ago). Software model checkers and other program-analysis tools require the source code of the system under study. The accessibility of source code is also an increasingly tenuous assumption. Many of the data stores targeted by tools such as Jepsen are closed source; large-scale architectures, while typically built from open source components, are increasingly *polyglot* (written in a wide variety of languages).

Finally, even if you assume that specifications or source code are available, techniques such as model checking are not a viable strategy for ensuring that applications are fault tolerant because, as mentioned, in the context of timeouts, fault tolerance itself is an end-to-end property that does not necessarily hold under composition. Even if you are lucky enough to build a system out of individually verified components, it does not follow the system is fault tolerant—you may have made a critical error in the glue that binds them.

The Vanguard. *The emerging ethos: YOLO.* Modern distributed systems

are simply too large, too heterogeneous, and too dynamic for these classic approaches to software quality to take root. In reaction, practitioners increasingly rely on resiliency techniques based on *testing* and *fault injection*.^{6,14,19,23,27,35} These “black box” approaches (which perturb and observe the complete system, rather than its components) are (arguably) better suited for testing an end-to-end property such as fault tolerance. Instead of deriving guarantees from understanding how a system works on the *inside*, testers of the system observe its behavior from the *outside*, building confidence that it functions correctly under stress.

Two giants have recently emerged in this space: Chaos Engineering⁶ and Jepsen testing.²⁴ Chaos Engineering, the practice of actively perturbing production systems to increase overall site resiliency, was pioneered by Netflix,⁶ but since then LinkedIn,⁵² Microsoft,³⁸ Uber,⁴⁷ and PagerDuty⁵ have developed Chaos-based infrastructures. Jepsen performs black box testing and fault injection on unmodified distributed data management systems, in search of correctness violations (for example, counterexamples that show an execution was not linearizable).

Both approaches are pragmatic and empirical. Each builds an understanding of how a system operates under faults by *running the system* and observing its behavior. Both approaches offer a pay-as-you-go method to resiliency: the initial cost of integration is low, and the more experiments that are performed, the higher the confidence that the system under test is robust. Because these approaches represent a straightforward enrichment of existing best practices in testing with well-understood fault injection techniques, they are easy to adopt. Finally, and perhaps most importantly, both approaches have been shown to be effective at identifying bugs.

Unfortunately, both techniques also have a fatal flaw: they are manual processes that require an *extremely* sophisticated operator. Chaos Engineers are a highly specialized subclass of site reliability engineers. To devise a custom fault injection strategy, a Chaos Engineer typically meets with different service teams to build an

understanding of the idiosyncrasies of various components and their interactions. The Chaos Engineer then targets those services and interactions that seem likely to have latent fault tolerance weaknesses. Not only is this approach difficult to scale since it must be repeated for every new composition of services, but its critical currency—a mental model of the system under study—is hidden away in a person’s brain. These points are reminiscent of a bigger (and more worrying) trend in industry toward reliability priesthoods,⁷ complete with icons (dashboards) and rituals (playbooks).

Jepsen is in principle a framework that anyone can use, but to the best of our knowledge all of the reported bugs discovered by Jepsen to date were discovered by its inventor, Kyle Kingsbury, who currently operates a “distributed systems safety research” consultancy.²⁴ Applying Jepsen to a storage system requires the superuser carefully read the system documentation, generate workloads, and observe the externally visible behaviors of the system under test. It is then up to the operator to choose—from the massive combinatorial space of “nemeses,” including machine crashes and network partitions—those fault schedules that are likely to drive the system into returning incorrect responses.

A human in the loop is the kiss of death for systems that need to keep up with software evolution. Human attention should always be targeted at tasks that computers cannot do! Moreover, the specialists that Chaos and Jepsen testing require are expensive and rare. Here, we show how geniuses can be abstracted away from the process of failure testing.

We Don’t Need Another Hero

Rapidly changing assumptions about our visibility into distributed system internals have made obsolete many if not all of the classic approaches to software quality, while emerging “chaos-based” approaches are fragile and unscalable because of their genius-in-the-loop requirement.

We present our vision of automated failure testing by looking at how the same changing environments that hastened the demise of time-tested resiliency techniques can enable new ones.

We argue the best way to automate the experts out of the failure-testing loop is to imitate their best practices in software and show how the emergence of sophisticated observability infrastructure makes this possible.

The order is rapidly fading. For large-scale distributed systems, the three fundamental assumptions of traditional approaches to software quality are quickly fading in the rearview mirror. The first to go was the belief that you could rely on experts to solve the hardest problems in the domain. Second was the assumption that a formal specification of the system is available. Finally, any program analysis (broadly defined) that requires that source code is available must be taken off the table. The erosion of these assumptions helps explain the move away from classic academic approaches to resiliency in favor of the black box approaches described earlier.

What hope is there of understanding the behavior of complex systems in this new reality? Luckily, the fact that it is more difficult than ever to understand distributed systems from the inside has led to the rapid evolution of tools that allow us to understand them from the *outside*. Call-graph logging was first described by Google,⁵¹ similar systems are in use at Twitter,⁴ Netflix,¹ and Uber,⁵⁰ and the technique has since been standardized.⁴³ It is reasonable to assume that a modern microservice-based Internet enterprise will already have instrumented its systems to collect call-graph traces. A number of startups that focus on observability have recently emerged.^{21,34} Meanwhile, provenance collection techniques for data processing systems^{11,22,42} are becoming mature, as are operating system-level provenance tools.⁴⁴ Recent work^{12,55} has attempted to infer causal and communication structure of distributed computations from raw logs, bringing high-level explanations of outcomes within reach even for uninstrumented systems.

Regarding testing distributed systems. Chaos Monkey, like they mention, is awesome, and I also highly recommend getting Kyle to run Jepsen tests.

—Commentator on HackerRumor

Away from the experts. While this quote is anecdotal, it is difficult to imagine a better example of the fundamental unscalability of the current state of the art. A single person cannot possibly keep pace with the explosion of distributed system implementations. If we can take the human out of this critical loop, we must; if we cannot, we should probably throw in the towel.

The first step to understanding how to automate any process is to comprehend the human component that we would like to abstract away. How do Chaos Engineers and Jepsen superusers apply their unique genius in practice? Here is the three-step recipe common to both approaches.

Step 1: Observe the system in action. The human element of the Chaos and Jepsen processes begins with principled observation, broadly defined.

A Chaos Engineer will, after studying the external API of services relevant to a given class of interactions, meet with the engineering teams to better understand the details of the implementations of the individual

services.²⁵ To understand the high-level interactions among services, the engineer will then peruse call-graph traces in a trace repository.³

A Jepsen superuser typically begins by reviewing the product documentation, both to determine the guarantees that the system should uphold and to learn something about the mechanisms by which it does so. From there, the superuser builds a model of the behavior of the system based on interaction with the system's external API. Since the systems under study are typically data management and storage, these interactions involve generating histories of reads and writes.³¹

The first step to understanding what can go wrong in a distributed system is watching things go right: observing the system in the common case.

Step 2. Build a mental model of how the system tolerates faults. The common next step in both approaches is the most subtle and subjective. Once there is a mental model of how a distributed system behaves (at least in the common case), how is it used to help choose the appropriate faults to inject? At this point we are forced to dabble in conjecture: bear with us.

Fault tolerance is redundancy. Given some fixed set of faults, we say that a system is “fault tolerant” exactly if it operates correctly in all executions in which those faults occur. What does it mean to “operate correctly”? Correctness is a system-specific notion, but, broadly speaking, is expressed in terms

of properties that are either *maintained* throughout the system's execution (for example, system invariants or safety properties) or *established* during execution (for example, liveness properties). Most distributed systems with which we interact, though their executions may be unbounded, nevertheless provide finite, bounded interactions that have *outcomes*. For example, a broadcast protocol may run “forever” in a reactive system, but each broadcast delivered to *all* group members constitutes a successful execution.

By viewing distributed systems in this way, we can revise the definition: A system is fault tolerant if it provides sufficient mechanisms to achieve its successful outcomes despite the given class of faults.

Step 3: Formulate experiments that target weaknesses in the façade. If we could understand all of the ways in which a system can obtain its good outcomes, we could understand which faults it can tolerate (or which faults it could be sensitive to). We assert that (whether they realize it or not!) the process by which Chaos Engineers and Jepsen superusers determine, on a system-by-system basis, which faults to inject uses precisely this kind of reasoning. A target experiment of faults that knocks out *all* of the supports for an expected outcome.

Carrying out the experiments turns out to be the easy part. Fault injection infrastructure, much like observability infrastructure, has evolved rapidly in recent years. In contrast to random, coarse-grained approaches to distributed fault injection such as Chaos Monkey,²³ approaches such as FIT (failure injection testing)¹⁷ and Grem-lin³² allow faults to be injected at the granularity of individual requests with high precision.

Step 4. Profit! This process can be effectively automated. The emergence of sophisticated tracing tools described earlier makes it easier than ever to build redundancy models even from the executions of black box systems. The rapid evolution of fault injection infrastructure makes it easier than ever to test fault hypotheses on large-scale systems. Figure 1 illustrates how the automation described in this here fits neatly between existing observ-

Figure 1. Our vision of automated failure testing.

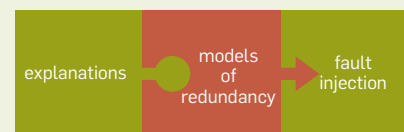
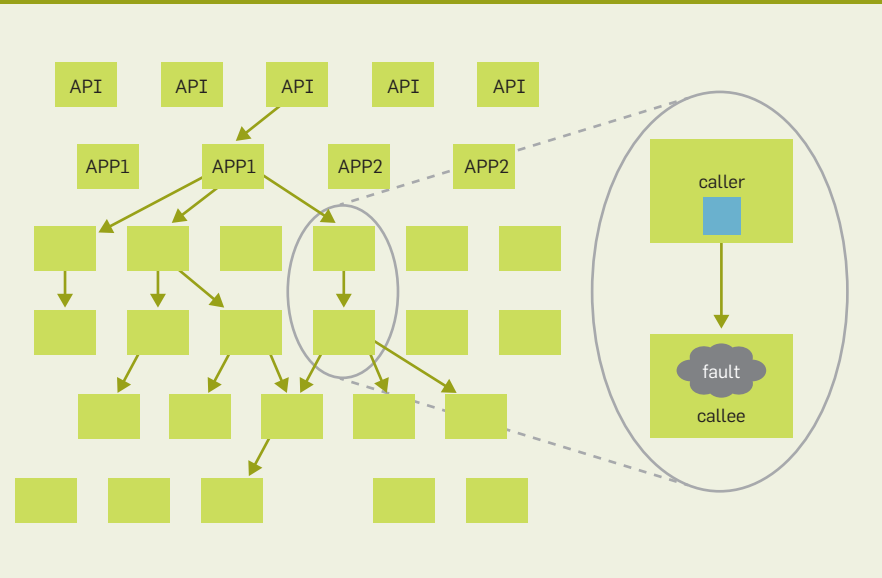


Figure 2. Fault injection and fault-tolerant code.




ability infrastructure and fault injection infrastructure, consuming the former, maintaining a model of system redundancy, and using it to parameterize the latter. Explanations of system outcomes and fault injection infrastructures are already available. In the current state of the art, the puzzle piece that fits them together (*models of redundancy*) is a manual process. LDFI (as we will explain) shows that automation of this component is possible.

A Blast from the Past


In previous work, we introduced a bug-finding tool called LDFI (lineage-driven fault injection).² LDFI uses data provenance collected during simulations of distributed executions to build *derivation graphs* for system outcomes. These graphs function much like the models of system redundancy described earlier. LDFI then converts the derivation graphs into a Boolean formula whose satisfying assignments correspond to combinations of faults that invalidate all derivations of the outcome. An experiment targeting those faults will then either expose a bug (that is, the expected outcome fails to occur) or reveal additional derivations (for example, after a timeout, the system fails over to a backup) that can be used to enrich the model and constrain future solutions.

At its heart, LDFI reapplies well-understood techniques from data management systems, treating fault tolerance as a materialized view maintenance problem.^{2,13} It models a distributed system as a query, its expected outcomes as query outcomes, and critical facts such as “replica *A* is up at time *t*” and “there is connectivity between nodes *X* and *Y* during the interval *i* . . . *j*” as base facts. It can then ask a how-to query:³⁷ What changes to base data will cause changes to the derived data in the view? The answers to this query are the faults that could, according to the current model, invalidate the expected outcomes.

The idea seems far-fetched, but the LDFI approach shows a great deal of promise. The initial prototype demonstrated the efficacy of the approach at the level of protocols, identifying bugs in replication, broadcast, and commit protocols.^{2,46} Notably, LDFI reproduced a bug in the replication protocol used by the Kafka distributed log²⁶ that was first



The rapid evolution of fault injection infrastructure makes it easier than ever to test fault hypotheses on large-scale systems.



(manually) identified by Kingsbury.³⁰ A later iteration of LDFI is deployed at Netflix,¹ where (much like the illustration in Figure 1) it was implemented as a microservice that consumes traces from a call-graph repository service and provides inputs for a fault injection service. Since its deployment, LDFI has identified 11 critical bugs in user-facing applications at Netflix.¹

Rumors from the Future

The prior research presented earlier is only the tip of the iceberg. Much work still needs to be undertaken to realize the vision of fully automated failure testing for distributed systems. Here, we highlight nascent research that shows promise and identifies new directions that will help realize our vision.

Don’t overthink fault injection. In the context of resiliency testing for distributed systems, attempting to enumerate and faithfully simulate every possible kind of fault is a tempting but distracting path. The problem of understanding all the *causes* of faults is not directly relevant to the target, which is to ensure that code (along with its configuration) intended to detect and mitigate faults performs as expected.

Consider Figure 2: The diagram on the left shows a microservice-based architecture; arrows represent calls generated by a client request. The right-hand side zooms in on a pair of interacting services. The shaded box in the caller service represents the fault tolerance logic that is intended to detect and handle faults of the callee. Failure testing targets bugs in this logic. The fault tolerance logic targeted in this bug search is represented as the shaded box in the caller service, while the injected faults affect the callee.

The common *effect* of all faults, from the perspective of the caller, is explicit error returns, corrupted responses, and (possibly infinite) delay. Of these manifestations, the first two can be adequately tested with unit tests. The last is difficult to test, leading to branches of code that are infrequently executed. If we inject *only* delay, and only at component boundaries, we conjecture that we can address the majority of bugs related to fault tolerance.

Explanations everywhere. If we can provide better explanations of system outcomes, we can build better models


of redundancy. Unfortunately, a barrier to entry for systems such as LDFI is the unwillingness of software developers and operators to instrument their systems for tracing or provenance collection. Fortunately, operating system-level provenance-collection techniques are mature and can be applied to uninstrumented systems.

Moreover, the container revolution makes simulating distributed executions of black box software within a single hypervisor easier than ever. We are actively exploring the collection of system call-level provenance from unmodified distributed software in order to select a custom-tailored fault injection schedule. Doing so requires extrapolating application-level causal structure from low-level traces, identifying appropriate *cut points* in an observed execution, and finally synchronizing the execution with fault injection actions.


We are also interested in the possibility of inferring high-level explanations from even noisier signals, such as raw logs. This would allow us to relax the assumption that the systems under study have been instrumented to collect execution traces. While this is a difficult problem, work such as the Mystery Machine¹² developed at Facebook shows great promise.

Toward better models. The LDFI system represents system redundancy using derivation graphs and treats the task of identifying possible bugs as a materialized-view maintenance problem. LDFI was hence able to exploit well-understood theory and mechanisms from the history of data management systems research. But this is just one of many ways to represent how a system provides alternative computations to achieve its expected outcomes.

A shortcoming of the LDFI approach is its reliance on assumptions of determinism. In particular, it assumes that if it has witnessed a computation that, under a particular contingency (that is, given certain inputs and in the presence of certain faults), produces a successful outcome, then any future computation under that contingency will produce the same outcome. That is to say, it ignores the uncertainty in timing that is fundamental to distributed systems. A more appropriate way to model system redundancy would be



The container revolution makes simulating distributed executions of black-box software within a single hypervisor easier than ever.



to embrace (rather than abstracting away) this uncertainty.

Distributed systems are probabilistic by nature and are arguably better modeled probabilistically. Future directions of work include the probabilistic representation of system redundancy and an exploration of how this representation can be exploited to guide the search of fault experiments. We encourage the research community to join in exploring alternative internal representations of system redundancy.

Turning the explanations inside out. Most of the classic work on data provenance in database research has focused on aspects related to human-computer interaction. Explanations of why a query returned a particular result can be used to debug both the query and the initial database—given an unexpected result, what changes could be made to the query or the database to fix it? By contrast, in the class of systems we envision (and for LDFI concretely), explanations are part of the internal language of the reasoner, used to construct models of redundancy in order to drive the search through faults.

Ideally, explanations should play a role in both worlds. After all, when a bug-finding tool such as LDFI identifies a counterexample to a correctness property, the job of the programmers has only just begun—now they must undertake the onerous job of distributed debugging. Tooling around debugging has not kept up with the explosive pace of distributed systems development. We continue to use tools that were designed for a single site, a uniform memory, and a single clock. While we are not certain what an ideal distributed debugger should look like, we are quite certain that it does not look like GDB (GNU Project debugger).³⁶ The derivation graphs used by LDFI show how provenance can also serve a role in debugging by providing a concise, visual explanation of how the system reached a bad state.

This line of research can be pushed further. To understand the root causes of a bug in LDFI, a human operator must review the provenance graphs of the good and bad executions and then examine the ways in which they differ. Intuitively, if you could abstractly *subtract* the (incomplete by assumption) explanations of the bad outcomes from the explanations of the good out-

comes,¹⁰ then the root cause of the discrepancy would be likely to be near the “frontier” of the difference.

Conclusion

A sea change is occurring in the techniques used to determine whether distributed systems are fault tolerant. The emergence of fault injection approaches such as Chaos Engineering and Jepsen is a reaction to the erosion of the availability of expert programmers, formal specifications, and uniform source code. For all of their promise, these new approaches are crippled by their reliance on superusers who decide which faults to inject.

To address this critical shortcoming, we propose a way of modeling and ultimately automating the process carried out by these superusers. The enabling technologies for this vision are the rapidly improving observability and fault injection infrastructures that are becoming commonplace in the industry. While LDFI provides constructive proof that this approach is possible and profitable, it is only the beginning. Much work remains to be done in targeting faults at a finer grain, constructing more accurate models of system redundancy, and providing better explanations to end users of *exactly what went wrong* when bugs are identified. The distributed systems research community is invited to join in exploring this new and promising domain. **□**

Related articles on queue.acm.org

Fault Injection in Production

John Allspaw

<http://queue.acm.org/detail.cfm?id=2353017>

The Verification of a Distributed System

Caitie McCaffrey

<http://queue.acm.org/detail.cfm?id=2889274>

Injecting Errors for Fun and Profit

Steve Chessin

<http://queue.acm.org/detail.cfm?id=1839574>

References

- Alvaro, P. et al. Automating failure-testing research at Internet scale. In *Proceedings of the 7th ACM Symposium on Cloud Computing* (2016), 17–28.
- Alvaro, P., Rosen, J., Hellerstein, J.M. Lineage-driven fault injection. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2015), 331–346.
- Andrus, K. Personal communication, 2016.
- Aniszczuk, C. Distributed systems tracing with Zipkin. Twitter Engineering; <https://blog.twitter.com/2012/distributed-systems-tracing-with-zipkin>.
- Barth, D. Inject failure to make your systems more reliable. DevOps.com; <http://devops.com/2014/06/03/inject-failure/>.
- Basiri, A. et al. Chaos Engineering. *IEEE Software* 33, 3 (2016), 35–41.
- Beyer, B., Jones, C., Petoff, J., Murphy, N.R. *Site Reliability Engineering*. O'Reilly, 2016.
- Birrell, A.D., Nelson, B.J. Implementing remote procedure calls. *ACM Trans. Computer Systems* 2, 1 (1984), 39–59.
- Chandra, T.D., Hadzilacos, V., Toueg, S. The weakest failure detector for solving consensus. *J.ACM* 43, 4 (1996), 685–722.
- Chen, A. et al. The good, the bad, and the differences: better network diagnostics with differential provenance. In *Proceedings of the ACM SIGCOMM Conference* (2016), 115–128.
- Chothia, Z., Liagouris, J., McSherry, F., Roscoe, T. Explaining outputs in modern data analytics. In *Proceedings of the VLDB Endowment* 9, 12 (2016): 1137–1148.
- Chow, M. et al. The Mystery Machine: End-to-end performance analysis of large-scale Internet services. In *Proceedings of the 11th Usenix Conference on Operating Systems Design and Implementation* (2014), 217–231.
- Cui, Y., Widom, J., Wiener, J.L. Tracing the lineage of view data in a warehousing environment. *ACM Trans. Database Systems* 25, 2 (2000), 179–227.
- Dawson, S., Jahanian, F., Mitton, T. ORCHESTRA: A Fault Injection Environment for Distributed Systems. In *Proceedings of the 26th International Symposium on Fault-tolerant Computing*, (1996).
- Fischer, M.J., Lynch, N.A., Paterson, M.S. Impossibility of distributed consensus with one faulty process. *J. ACM* 32, 2 (1985): 374–382; <https://groups.csail.mit.edu/tds/papers/Lynch/jacm85.pdf>.
- Fisman, D., Kupferman, O., Lustig, Y. On verifying fault tolerance of distributed protocols. In *Tools and Algorithms for the Construction and Analysis of Systems, Lecture Notes in Computer Science 4963*, Springer Verlag (2008). 315–331.
- Gopalani, N., Andrus, K., Schmaus, B. FIT: Failure injection testing. Netflix Technology Blog; <http://techblog.netflix.com/2014/10/fat-failure-injection-testing.html>.
- Gray, J. Why do computers stop and what can be done about it? Tandem Technical Report 85.7 (1985); <http://www.hpl.hp.com/techreports/tandem/TR-85.7.pdf>.
- Gunawi, H.S. et al. FATE and DESTINI: A framework for cloud recovery testing. In *Proceedings of the 8th Usenix Conference on Networked Systems Design and Implementation* (2011), 238–252; <http://db.cs.berkeley.edu/papers/nsdi11-fate-destini.pdf>.
- Holzmann, G. *The SPIN Model Checker: Primer and Reference Manual*. Addison-Wesley Professional, 2003.
- Honeycomb. 2016; <https://honeycomb.io/>.
- Interlandi, M. et al. Titian: Data provenance support in Spark. In *Proceedings of the VLDB Endowment* 9, 33 (2015), 216–227.
- Izrailevsky, Y., Tseitlin, A. The Netflix Simian Army. Netflix Technology Blog; <http://techblog.netflix.com/2011/07/netflix-simian-army.html>.
- Jepsen. Distributed systems safety research, 2016; <http://jepsen.io/>.
- Jones, N. Personal communication, 2016.
- Kafka 0.8.0. Apache, 2013; <https://kafka.apache.org/08/documentation.html>.
- Kanawati, G.A., Kanawati, N.A., Abraham, J.A. Ferrari: A flexible software-based fault and error injection system. *IEEE Trans. Computers* 44, 2 (1995): 248–260.
- Kendall, S.C., Waldo, J., Wollrath, A., Wyant, G. A note on distributed computing. Technical Report, 1994. Sun Microsystems Laboratories.
- Killian, C.E., Anderson, J.W., Jhala, R., Vahdat, A. Life, death, and the critical transition: Finding liveness bugs in systems code. *Networked System Design and Implementation*, (2007); 243–256.
- Kingsbury, K. Call me maybe: Kafka, 2013; <http://aphyr.com/posts/293-call-me-maybe-kafka>.
- Kingsbury, K. Personal communication, 2016.
- Lafeldt, M. The discipline of Chaos Engineering. Gremlin Inc., 2017; <https://blog.gremlininc.com/the-discipline-of-chaos-engineering-e39d2383c459>.
- Lampson, B.W. Atomic transactions. In *Distributed Systems—Architecture and Implementation, An Advanced Course*: (1980), 246–265; https://link.springer.com/chapter/10.1007%2F3-540-10571-9_11.
- LightStep. 2016; <http://lightstep.com/>.
- Marinescu, P.D., Canda, G. LFI: A practical and general library-level fault injector. In *IEEE/IFIP International Conference on Dependable Systems and Networks* (2009).
- Matloff, N., Salzman, P.J. *The Art of Debugging with GDB, DDD, and Eclipse*. No Starch Press, 2008.
- Meliou, A., Suciu, D., Tiresias: The database oracle for how-to queries. *Proceedings of the ACM SIGMOD International Conference on the Management of Data* (2012), 337–348.
- Microsoft Azure Documentation. Introduction to the fault analysis service, 2016; <https://azure.microsoft.com/en-us/documentation/articles/service-fabric-testability-overview/>.
- Musuvathi, M. et al. CMC: A pragmatic approach to model checking real code. *ACM SIGOPS Operating Systems Review*. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation* 36 (2002), 75–88.
- Musuvathi, M. et al. Finding and reproducing Heisenbugs in concurrent programs. In *Proceedings of the 8th Usenix Conference on Operating Systems Design and Implementation* (2008), 267–280.
- Newcombe, C. et al. Use of formal methods at Amazon Web Services. Technical Report, 2014; <http://lampart.azurewebsites.net/ta/formal-methods-amazon.pdf>.
- Olston, C., Reed, B. Inspector Gadget: A framework for custom monitoring and debugging of distributed data flows. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data* (2011), 1221–1224.
- OpenTracing. 2016; <http://opentracing.io/>.
- Pasquier, T.F. J.-M., Singh, J., Eyers, D.M., Bacon, J. CamFlow: Managed data-sharing for cloud services, 2015; <https://arxiv.org/pdf/1506.04391.pdf>.
- Patterson, D.A., Gibson, G., Katz, R.H. A case for redundant arrays of inexpensive disks (RAID). In *Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data*, 109–116; <http://web.mit.edu/6.033/2015/wwwdocs/papers/Patterson88.pdf>.
- Ramasubramanian, K. et al. Growing a protocol. In *Proceedings of the 9th Usenix Workshop on Hot Topics in Cloud Computing* (2017).
- Reinhold, E. Rewriting Uber engineering: The opportunities microservices provide. *Uber Engineering*, 2016; <https://eng.uber.com/building-tincup/>.
- Saltzer, J. H., Reed, D.P., Clark, D.D. End-to-end arguments in system design. *ACM Trans. Computing Systems* 2, 4 (1984): 277–288.
- Sandberg, R. The Sun network file system: design, implementation and experience. Technical report, Sun Microsystems. In *Proceedings of the Summer 1986 Usenix Technical Conference and Exhibition*.
- Shkuro, Y. Jaeger: Uber's distributed tracing system. *Uber Engineering*, 2017; <https://uber.github.io/jaeger/>.
- Sigelman, B.H. et al. Dapper, a large-scale distributed systems tracing infrastructure. Technical report. Research at Google, 2010; <https://research.google.com/pubs/pub36356.html>.
- Shenoy, A. A deep dive into Simoorg: Our open source failure induction framework. *LinkedIn Engineering*, 2016; <https://engineering.linkedin.com/blog/2016/03/deep-dive-Simoorg-open-source-failure-induction-framework>.
- Yang, J. et al. L., Zhou, L. MODIST: Transparent model checking of unmodified distributed systems. In *Proceedings of the 6th Usenix Symposium on Networked Systems Design and Implementation* (2009), 213–228.
- Yu, Y., Manolios, P., Lamport, L. Model checking TLA+ specifications. In *Proceedings of the 10th IFIP WG 10.5 Advanced Research Working Conference on Correct Hardware Design and Verification Methods* (1999), 54–66.
- Zhao, X. et al. Lprof: A non-intrusive request flow profiler for distributed systems. In *Proceedings of the 11th Usenix Conference on Operating Systems Design and Implementation* (2014), 629–644.

Peter Alvaro is an assistant professor of computer science at the University of California Santa Cruz, where he leads the Disorderly Labs research group (disorderlylabs.github.io).

Severine Tymon is a technical writer who has written documentation for both internal and external users of enterprise and open source software, including for Microsoft, CNET, VMware, and Oracle.

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

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

Building a decentralized Web-delivery model.

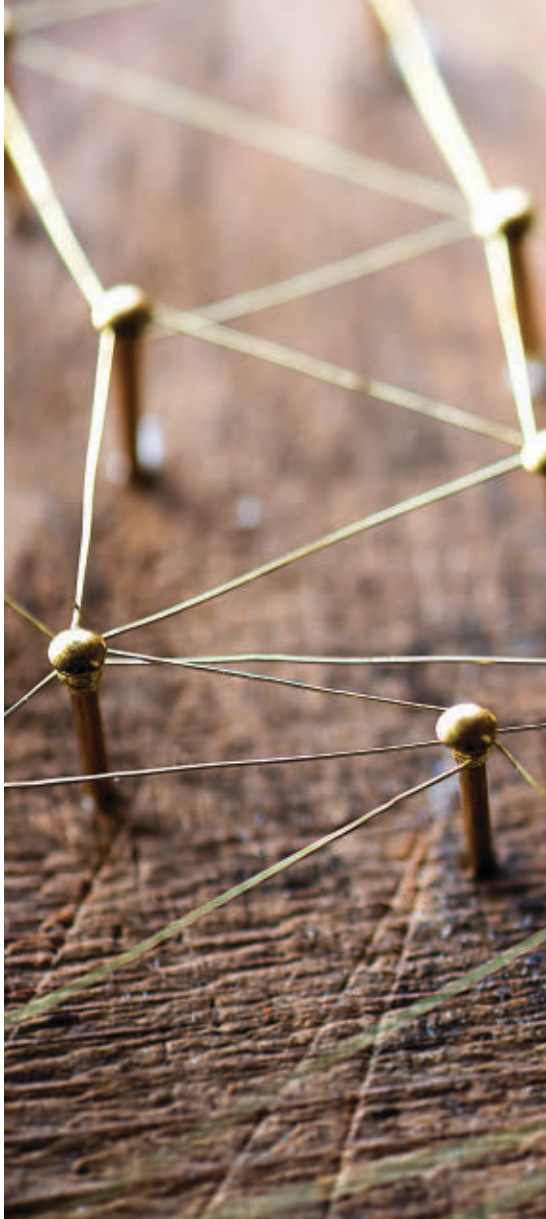
BY JACOB LOVELESS

Cache Me If You Can

YEARS AGO I squandered most of my summer break locked inside my apartment, tackling an obscure problem in network theory—the bidirectional channel capacity problem. I was convinced that I was close to a breakthrough. (I wasn't.) Papers were everywhere, intermingled with the remnants of far too many 10¢ Taco Tuesday wrappers.

A good friend stopped by to bring better food, lend a mathematical hand, and put an end to my solitary madness. She listened carefully while I jumped across the room grabbing papers and incoherently babbling about my “breakthrough.”

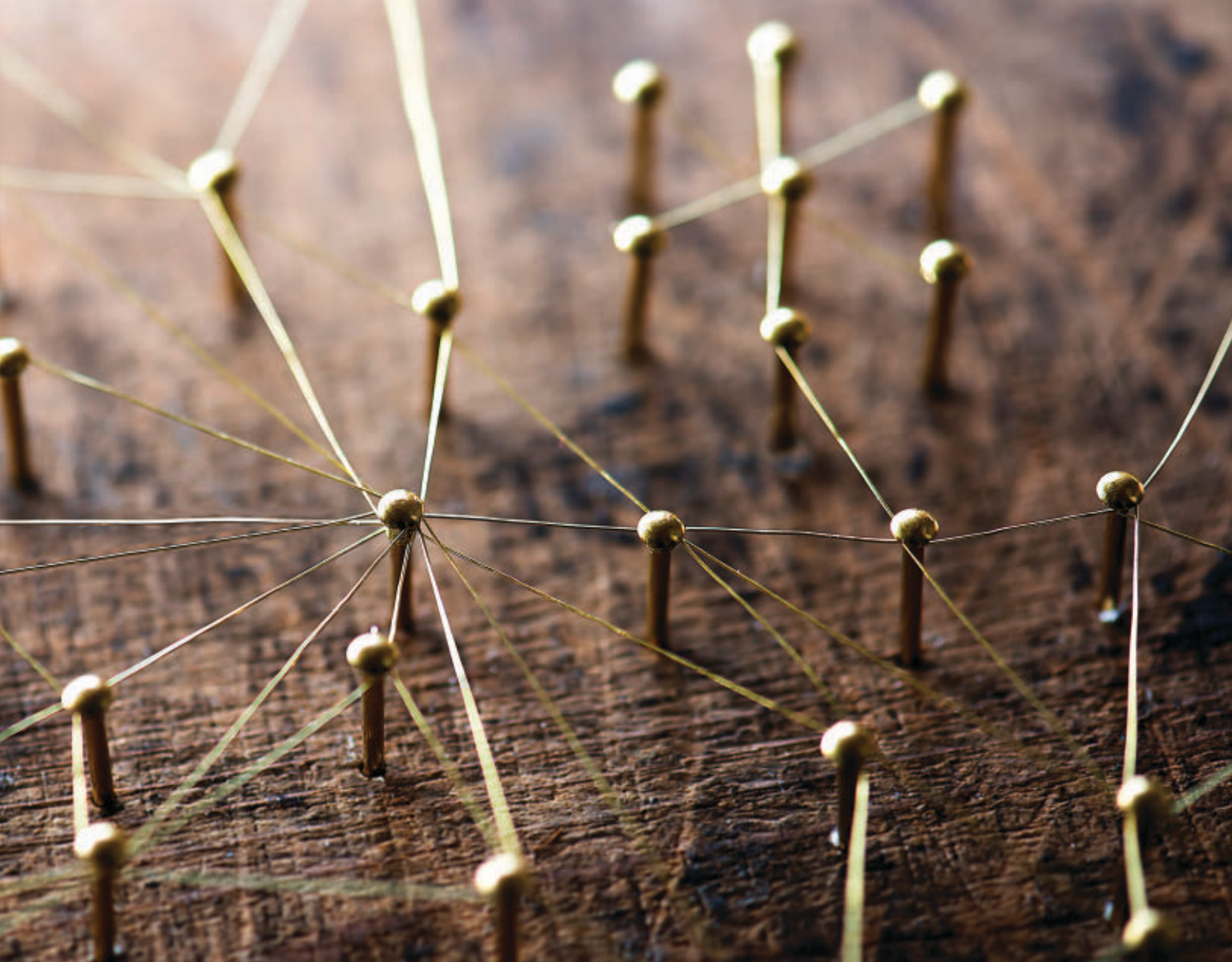
Then she somberly grabbed a pen and wrote out the flaw that I had missed, obliterating the proof.



I was stunned and heartbroken. She looked up and said, “But this is great, because what you’ve done here is define the problem more concretely.” She continued with a simple truth that I have carried with me ever since:

“Most times, defining the problem is harder and more valuable than finding the answer. The world needs hard, well-defined problems. Generally, only one or two people work together on an answer—but hundreds work on a well-defined problem.”

And so, dear reader, this is where I would like to begin. This article looks at the problems inherent in building a more decentralized Internet. Audacious? Yes, but this has become a renewed focus in recent years, even by the father of the Web himself (see Tim Berners-Lee’s Solid project⁴). Several companies and open source projects are now focusing on different aspects of the “content-delivery”



problem. Our company Edgemesh (<https://edgemesh.com>) is working on peer-enhanced client-side content acceleration, alongside other next-generation content-delivery networks such as Peer5 (<https://peer5.com>) and Streamroot (<https://streamroot.io>), both of which are focused on video delivery. Others, such as the open source InterPlanetary File System (IPFS; <https://ipfs.io>) project are looking at completely new ways of defining and distributing and defining “the Web.”

Indeed, the concept of a better Internet has crept into popular media. In season 4 of HBO’s sitcom *Silicon Valley*, the protagonist Richard Hendricks devises a new way to distribute content across the Internet in a completely distributed manner using a P2P protocol. “If we could do it, we could build a completely decentralized version of our current Internet,” Hendricks says, “with no firewalls, no tolls, no government regu-

lation, no spying. Information would be totally free in every sense of the word.” The story line revolves around the idea that thousands of users would allocate a small portion of their available storage on their mobile devices, and that the Pied Piper software would assemble the storage across these devices in a distributed storage “cloud.” Then, of course, phones explode and hilarity ensues.

The core idea of a distributed Internet does make sense, but how would it be built? As I learned in my self-imposed solitary confinement so long ago, before diving into possible solutions, you need to define the problems more clearly.

Problems of a Distributed Internet

In 2008, Tom Leighton, cofounder of Akamai, the largest content-distribution network in the world, outlined four major architectures for content distribu-

tion: centralized hosting, “big datacenter” content-delivery networks (CDNs), highly distributed CDNs, and P2P networks. Of these, Leighton noted that:

“P2P can be thought of as taking the distributed architecture to its logical extreme, theoretically providing nearly infinite scalability. Moreover, P2P offers attractive economics under current network pricing structures.”¹¹

He then noted what others have found in the past, that although the P2P design is *theoretically* the most scalable, there are several practical issues, specifically *throughput*, *availability*, and *capacity*.

Throughput. The most commonly noted issue is the limited uplink capacity of edge devices, as noted by Leighton in his 2008 article:

P2P faces some serious limitations; most notably because the total download capacity of a P2P network is throttled by its total uplink capacity.

Figure 1. Bandwidth rates (upload/download, broadband/mobile).

Region	Terrestrial		YoY%		Mobile		YoY%	
	Download (Mbps)	Upload (Mbps)	Download	Upload	Download (Mbps)	Upload (Mbps)	Download	Upload
United States	54.97	18.88	42%	51%	19.61	7.94	794%	28%
United Kingdom	40.12	40.12	18%	23%	24.19	10.03	38%	31%
Singapore	190.9	178.69	18%	20%	45.61	17.93	16%	13%
Hong Kong	153.53	146.69	13%	0%	18.9	8.78	5%	1%
Brazil	16.53	6.08	8%	11%	11.31	4.27	37%	35%

Unfortunately, for consumer broadband connections, uplink speeds tend to be much lower than downlink speeds: Comcast's standard high-speed Internet package, for example, offers 6Mbps for download but only 384Kbps for upload (1/16th of download throughput).¹¹

This limitation is not as acute today as it was nearly a decade ago when upload speeds in the U.S. hovered around .5Mbps. Figure 1 shows the current upload and download as taken from Speedtest.net (<http://www.speedtest.net/reports/>). These data points show that global “last-mile” throughput rates are nearly 30 times their 2008 counterparts. Is this enough? Would a peer with an upload rate at the lower quartile of these metrics (~4Mbps) suffice? This question has been thoroughly explored in regard to actual webpage load time.

When Mike Belshe, of Google

SPDY fame, looked at the relationship between end-client bandwidth and page-load time, he discovered that “bandwidth doesn't matter (much).”³ Once the client's available bandwidth reached 5Mbps, the impact on the end user's page load is negligible. Figure 2 shows page-load time as a function of client bandwidth, assuming a fixed RTT (round-trip time) of 60ms.

Availability. The next major hurdle for the distributed Internet is peer availability. Namely, are there enough peers, and are they online and available for enough time to provide value? In the past 10 years the edge device count has certainly increased, but has it increased enough? Looking at “Internet Trends 2017,”¹⁴ compiled by Mary Meeker of the venture capital firm KPCB, you can see how much the “available peer” count has increased over the past ten years from mobile alone (see Figure 3).

Today, approximately 49% of the

world's population is connected¹⁰—around 3.7 billion people, many with multiple devices—so that's a *big* pool of edge devices. Peter Levine of the venture capital firm Andressen Horowitz has taken us out a few years further and predicted we will shortly be going beyond billions and heading toward *trillions* of devices.¹²

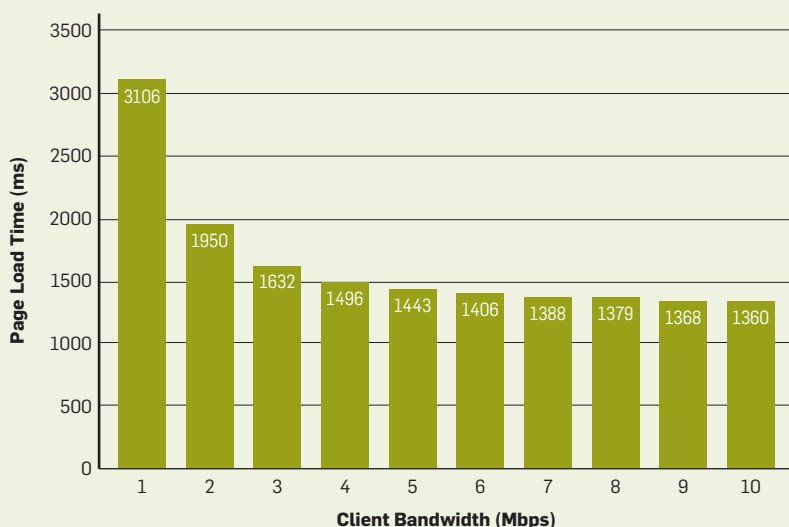
You can get a sense of scale by looking at an Edgemesh network for a single e-commerce customer's website with a global client base.

It's probably safe to say there are enough devices online, but does the average user stay online long enough to be available? What is “long enough” for a peer to be useful?

A sensible place to start might be to want peers to be online long enough for any peer to reach any other peer anywhere on the globe. Given that, we can set some bounds.

The circumference of the Earth is approximately 40,000km. The rule of thumb is that light takes 4.9 microseconds to move 1km through fiber optics. That would mean data could circumnavigate the globe in about 1/5th of a second (196 milliseconds). Oh, if wishing only made it so, but as Stanford University's Stuart Cheshire points out in “It's the Latency, Stupid,”⁶ the Internet operates at least a factor of two slower than this. This 2x slowdown would mean it would take approximately 400 milliseconds to get around the globe. Unfortunately, I have spent some time in telecom—specifically in latency-optimized businesses¹³—and I think this time needs to be doubled again to account for global transit routing; thus, the data can go around the world in some 800 milliseconds. If users are online and available for sub-800 millisecond in-

Figure 2. Page load time as a function of client bandwidth.



tervals, this may become problematic. Since most decentralized solutions would require the user to visit the content (for example, be on the website), the real question is, what is the average page-view time for users across the globe?

Turns out it is 3 minutes 36 seconds,²⁴ or 216,000 milliseconds.

To double-check this, I took all peer-session times (the amount of time Edgemesh peers were online and connected to the mesh) across the Edgemesh user base for the past six months (Figure 4). The average was right in line at 3 minutes 47 seconds.

In either case, if the node stays online just long enough to download a single webpage, that would be enough time for the data to circumnavigate the globe 270 times, certainly long enough to contact a peer anywhere on Earth.

Capacity. If enough users are online for a long enough duration, and they have an acceptable egress throughput (upload bandwidth), all that remains is the question of whether there is enough spare capacity (disk space) available to provide a functional network.

If we assume a site has 20% of its users on mobile and 80% of its users on desktops—and further expand this to 500MB of available capacity per desktop user and 50MB per mobile user (the lower end of browser-available storage pools)—we can extract an estimated required mesh size to achieve a given cache hit rate if the requested content follows a Zipf distribution.¹ Essentially, a website with 500GB of static content, that is, about 16 million average Web images, would need an online capacity of two million distinct nodes to achieve a theoretical offload of 100% of its traffic to a P2P mesh (approximately an 8:1 ratio of images to users).

Enabling a Distributed Internet

Now that we have better defined the problems and established the *theoretical* feasibility of a new solution, it's time to look at the technology available to bring to bear on the problem. To start, we can constrain our focus a bit. Implementations such as IPFS focus on distributing the entire content base, allowing you to free yourself from the restrictions of Web servers and

Figure 3. Mary Meeker's 2017 smartphone analytics.

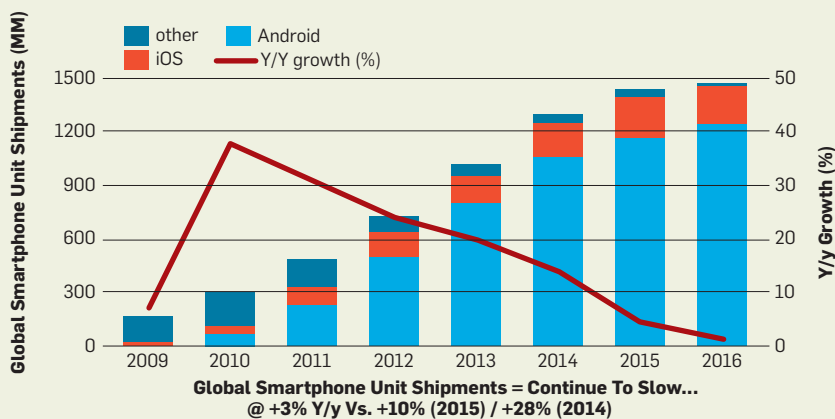
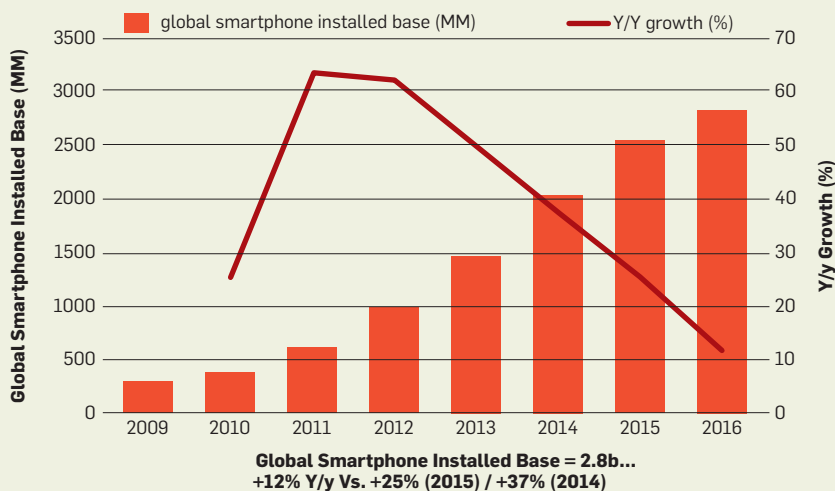
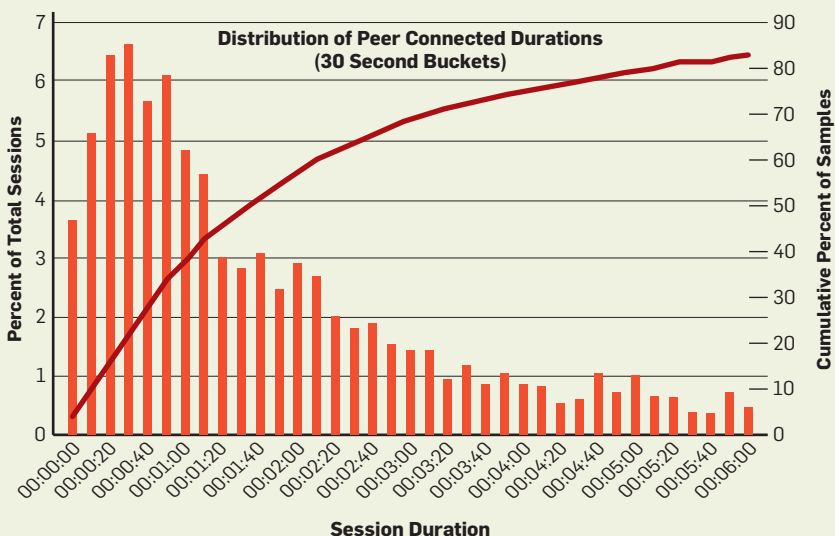


Figure 4. Histogram of peer duration.



DNS entirely. This is a fantastic wholesale change, but the tradeoff is that it will require users to dramatically modify how they access content.

Since a P2P design is dependent on the total network size, this model has difficulty growing until it reaches

critical mass. At Edgemes, we wanted to focus on enhancing existing Web-content delivery transparently (for example, in the browser) without requiring any changes to the user experience. This means ensuring that the technology abides by the following

three restrictions:

- ▶ For users, the solution should be transparent.
- ▶ For developers, the solution should require zero infrastructure changes.
- ▶ For operations, the solution should be self-managing.

The next question is where exactly to focus.

Fully enabling peer-enhanced delivery of *all* content is difficult and dangerous, especially allowing for peer-delivered JavaScript to be executed. Is there an 80% solution? Trends posted by HTTP Archive reveal that static components (images/video/fonts/CSS) make up roughly 81% of the total page weight.⁹

Given these details, let's narrow the focus to enabling/enhancing edge delivery of these more traditional CDN assets and the associated challenges of moving and storing data.

Moving data: Building a new network (overlay). To support P2P distribution, an overlay network needs to be developed to allow the P2P connections to operate within the larger existing Internet infrastructure. Luckily, such a stack is available: WebRTC (Real-Time Communications¹⁹). Started in earnest in 2011 by Google, WebRTC is an in-browser networking stack that enables peer-to-peer communication. WebRTC is primarily employed by voice and video applications (Google Hangouts/Dua/Allo, Slack, Snapchat, Amazon Chime, WhatsApp, Facebook Messenger) to facilitate peer-to-peer video- and audioconferencing.

WebRTC is a big deal; in June 2016 Google provided several key milestones⁷ from stats it collected, with some additional updates six months later:²⁵

- ▶ Two billion Chrome browsers with WebRTC.
- ▶ One billion WebRTC audio/video minutes per week on Chrome.
- ▶ One petabyte of DataChannel traffic per week on Chrome (0.1% of all Web traffic).
- ▶ 1,200 WebRTC-based companies and projects (it was 950 in June).
- ▶ Five billion mobile app downloads that include WebRTC.

WebRTC support exists in the major browsers—Chrome, Firefox, Edge, and now Safari,² Comparing WebRTC's five-year adoption against other VoIP-style protocols shows the scale.⁸

Figure 5. Flow diagram for WebRTC Enhanced asset delivery.

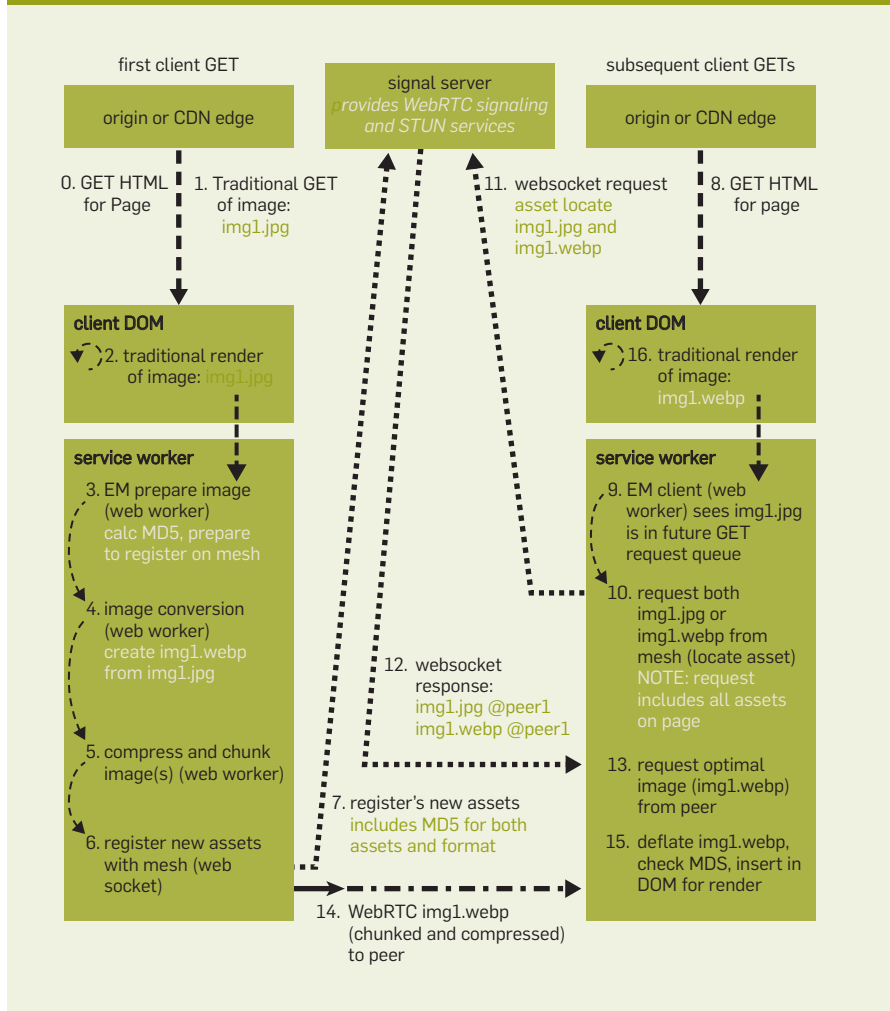
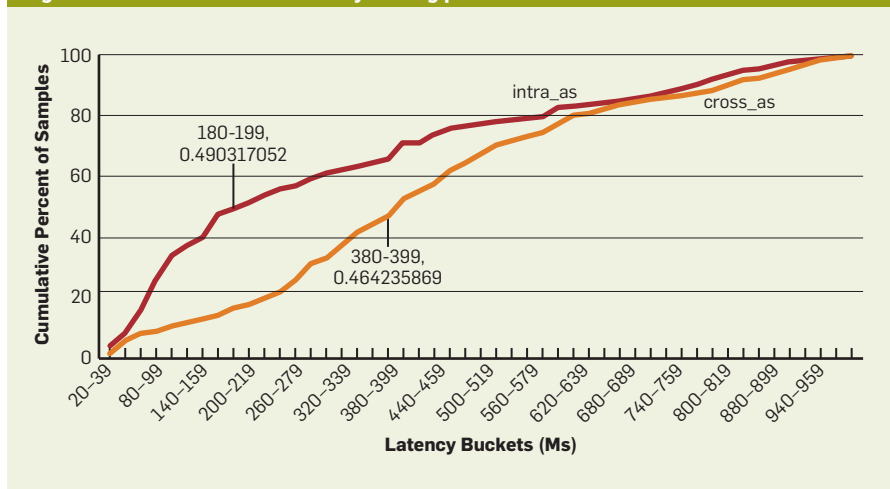


Figure 6. Performance increased by routing preference.



WebRTC is a user-space networking stack. Unlike HTTP, which is dependent on TCP for transfer, WebRTC has its roots in a much older protocol—Stream Control Transmission Protocol (SCTP)—and encapsulates this in User Datagram Protocol (UDP). This allows for much lower latency transfer, removes head-of-line blocking, and, as a separate network stack, allows WebRTC to use significantly more bandwidth than HTTP alone.

SCTP is a little like the third wheel of the transport layer of the Open Systems Interconnection (OSI) model—we often forget it’s there but it has some very powerful features. Originally introduced to support signaling in IP networks,²² SCTP quickly found adoption in next-generation networks (IMS and LTE).

WebRTC leverages SCTP to provide a reliable, message-oriented delivery transport (encapsulated in UDP or TCP, depending on the implementation⁵). Alongside SCTP, WebRTC leverages two additional major protocols: Datagram Transport Layer Security (DTLS) for security (a derivative of SSL) and Interactive Connectivity Establishment (ICE) to allow for support in network address translation (NAT) environments, for example, firewall traversal.

The details of the ICE protocol and how it works with signaling servers—for example, STUN and TURN—are beyond the scope of this article, but suffice it to say that WebRTC has all the necessary plumbing to enable real peer-to-peer networking.

A simple example is a WebRTC Golang implementation by Serene Han.²¹ Han’s chat demo allows you to pass the SDP (Session Description Protocol) details between two machines (copy paste signaling) to enable P2P chat. To run this yourself (assuming you have a Docker instance locally), simply do the following:

```
docker run -it golang bash
```

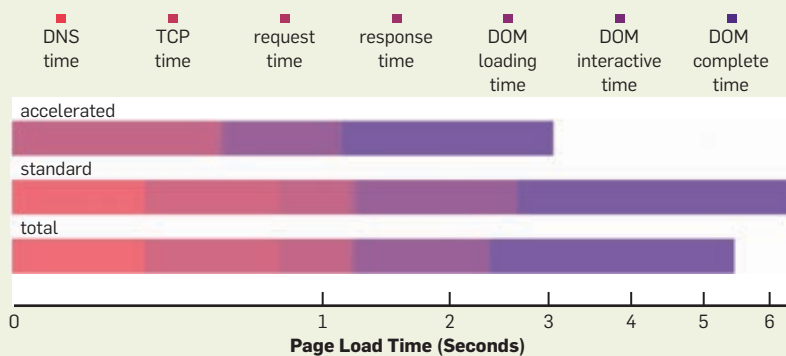
Then in the Docker instance, this one-liner will get you set up:

```
apt-get update && apt-get install libx11-dev -y && \
go get github.com/keroserene/go-webrtc && \
```

Figure 7. Effect of P2P enhanced background caching on load time.

accelerated	standard
4249 page views	7674 page views
4248 visitors	4235 visitors
2.568 average load time (seconds)	4.838 average load time (seconds)
55.658 max load time (seconds)	56.775 max load time (seconds)
0.403 average net time (seconds)	0.852 average net time (seconds)
9.361 max net time (seconds)	36.465 max net time (seconds)
803 views over 4 seconds	3524 views over 4 seconds
1779 views over 2 seconds	5912 views over 2 seconds

Figure 8. Page load time components: Accelerated vs. standard.



```
cd /go/src/github.com/keroserene/go-webrtc && \
go run demo/chat/chat.go
```

If you prefer a browser-native starting point, look at simple-peer module,²⁰ originally from Feross Aboukhadijeh’s work with WebTorrent (<https://webtorrent.io>).

Storing data: Browser storage options and asset interception. The next step is finding a method both to intercept standard HTTP requests and to develop a system for storing peer-to-peer delivered assets. For the request-intercept problem, you have to look no further than the service worker.¹⁸ The

service worker is a new feature available in most browsers that allows for a background process to run in the browser. Like a Web worker, which can be used as a proxy for threads, a service worker has restrictions on how it can interact and exchange data with the Document Object Model (DOM).

The service worker does, however, have a powerful feature that was originally developed to support offline page loads: the Fetch API.¹⁶ The Fetch API allows a service worker to intercept request and response calls, similar to an HTTP proxy.

With the service worker online, you can now intercept traditional HTTP

requests and offload these requests to the P2P network. The last remaining component will be a browser local storage model where P2P-accelerated content can be stored and distributed. Although no fewer than five different storage options exist in the browser, the IndexedDB¹⁷ implementation is the only storage API available within a service-worker context and the DOM context, where the WebRTC code can execute, which is why Edgemesh chose it as the storage base. Alternatively, the CacheStorage API may also be used within the service-worker context.¹⁵

Implementing a Distributed Internet

We have a theoretically viable model to support peer-to-peer content delivery. We have a functional network stack to enable ad-hoc efficient peer-to-peer transfer and access to an in-browser storage medium. And so, the game is afoot!

Figure 5 is a flowchart of the Edgemesh P2P-accelerated content-delivery system. The figure shows where the service-worker framework will enable asset interception, and WebRTC (aided by a signal server) will facilitate browser-to-browser asset replication.

Returning to Mike Belshe's research, we can start to dig into some of the key areas to be optimized. Unlike bandwidth, where adding incrementally more bandwidth above 5Mbps has negligible impact on page-load time, latency (RTT) dramatically increases page-load time.³

WebRTC is already an efficient protocol, but the peer-selection process presents opportunities for further latency reduction. For example, if you are located in New York, providing a peer in Tokyo is likely a nonoptimal choice.

A simple optimization might be to prefer peers that reside in the same network, perhaps identified by the autonomous system (AS)²³ number of each peer. Even this simple optimization can cut the average latency by a factor of two. Figure 6 shows performance increase by intra-AS routing preference.

Another optimization is choosing which assets to replicate into a peer. For example, if a user is currently browsing the landing page of a site, we can essentially precache all the imag-

es for the next pages, effectively eliminating the latency altogether. This is a current area of research for the team at Edgemesh, but early solutions have already shown significant promise. Figure 7 shows the effective render time for Edgemesh-enabled clients (accelerated) and non-Edgemesh-enabled clients (standard) for a single customer domain. The average page-load time has been reduced by almost a factor of two.

This is most clearly seen when most of the page content can be effectively precached, as shown in the page-load time statistics of Figure 8.

Conclusion

It had been a few days since I had been outside, and a few more since I would make myself presentable. For the previous few weeks the team and I had been locked inside the office around the clock, essentially rewriting the software from scratch. We thought it would take a week, but we were now three months in. The growing pile of empty delivery bags resting atop the ad-hoc whiteboard tables we were using was making it difficult to make out exactly what the big change was. We were convinced this was the breakthrough we had been looking for (turns out it was), and that this version would be the one that cracked the problem wide open. I was head down trying to get my parts to work, and I was lost. Then I heard the knock at the door. She came in and sat down, patiently moving aside the empty pizza boxes while I babbled on about our big breakthrough and how I was stuck.

Then, just like she had nearly two decades earlier, she grabbed the marker and said:

"Honey, I think I see the issue. You haven't properly defined the problem. Most times, defining the problem is harder and more valuable than finding the answer. So, what exactly are you trying to solve?"

The world is more connected than it ever has been before, and with our pocket supercomputers and Internet of Things' future, the next generation of the Web might just be delivered in a peer-to-peer model. It's a giant problem space, but the necessary tools and technology are here today. We just need to define the problem a little better. **C**

References

1. Adamic, L.A., Huberman, B.A. Zipf's law and the Internet. *Glottometrics* 3 (2002), 143–150; <http://www.hpl.hp.com/research/idl/papers/ranking/adamicglottometrics.pdf>.
2. Apple Inc. Safari 11.0; https://developer.apple.com/library/content/releasenotes/General/WhatsNewInSafari/Safari_11_0/Safari_11_0.html.
3. Belshe, M. More bandwidth doesn't matter (much), 2010; <https://docs.google.com/a/chromium.org/viewer?a=v&pid=sites&srcid=Y2hyb21pdW0ub3JnfGRldnxneDoxMzcyOWI1N2I4YzI3NzE2>.
4. Berners-Lee, T. Solid; <https://solid.mit.edu/>.
5. BlogGeek.Me. Why was SCTP selected for WebRTC's data channel, 2014; <https://bloggeek.me/sctp-data-channel/>.
6. Cheshire, S. It's the latency, stupid, 1996–2001; <http://www.stuartcheshire.org/rants/latency.html>.
7. Google Groups. WebRTC, 2016; <https://groups.google.com/forum/#!topic/discuss-WebRTC/IOGqzWfKJfQ>.
8. Hart, C. WebRTC: One of 2016's biggest technologies no one has heard of. WebRTC World, 2017; <http://www.webrtworld.com/topics/webrtc-world/articles/428444-webrtc-one-2016s-biggest-technologies-no-one-has.htm>.
9. HTTP Archive; <http://httparchive.org/trends.php>.
10. Internet World Stats. World Internet usage and population statistics—March 31, 2017; <http://www.internetworldstats.com/stats.htm>.
11. Leighton, T. Improving performance on the Internet. *acmqueue* 6, 6 (2008); <http://queue.acm.org/detail.cfm?id=1466449>.
12. Levine, P. The end of cloud computing. *Andriessen Horowitz*; <http://a16z.com/2016/12/16/the-end-of-cloud-computing/>.
13. Loveless, J. Barbarians at the gateways. *acmqueue* 11, 8 (2013); <http://queue.acm.org/detail.cfm?id=2536492>.
14. Meeker, M. Internet trends 2017—code conference. KPCB; <http://www.kpcb.com/internet-trends>.
15. Mozilla Developer Network. CacheStorage, 2017; <https://developer.mozilla.org/en-US/docs/Web/API/CacheStorage>.
16. Mozilla Developer Network. FetchAPI, 2017; https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API.
17. Mozilla Developer Network. IndexedDB API, 2017; https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API.
18. Mozilla Developer Network. Using service workers, 2017; https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API/Using_Service_Workers.
19. Real-time communication in Web browsers (rtcweb) Charter for Working Group; <http://datatracker.ietf.org/wg/rtcweb/charter/>.
20. Simple WebRTC video/voice and data channels. Github; <https://github.com/edgemesh/simple-peer>.
21. WebRTC for Go. Github; <https://github.com/keroserene/go-webrtc>.
22. Wikipedia. Signalling System No. 7; https://en.wikipedia.org/wiki/Signalling_System_No._7.
23. Wikipedia. Autonomous system; [https://en.wikipedia.org/wiki/Autonomous_system_\(Internet\)](https://en.wikipedia.org/wiki/Autonomous_system_(Internet)).
24. Wolfgang Digital. E-commerce KPI benchmarks, 2016; https://www.wolfgangdigital.com/uploads/general/KPI_Infographik_2016.jpg.
25. YouTube. WebRTC, 2016; <https://youtu.be/OUyfFMgtPQ0?t=16504>.

Jacob Loveless is chief executive officer at Edgemesh Corporation, the edge-network acceleration platform. Previously, he served as CEO of Lucera Financial Infrastructures and was a partner at Cantor Fitzgerald, responsible for running the firm's global low-latency trading operations.

ACM Books. In-depth. Innovative. Insightful.

ACM and Morgan & Claypool Publishers present ACM Books: an all-new series of educational, research and reference works for the computing community. Inspired by the need for high-quality computer science publishing at the graduate, faculty and professional levels, ACM Books is affordable, current, and comprehensive in scope. ACM Books collections are available under an ownership model with archival rights included. We invite you to learn more about this exciting new program.

For more info please visit
<http://books.acm.org>

or contact ACM at
ACMbooks-Info@acm.org



**Association for
Computing Machinery**
2 Penn Plaza, Suite 701
New York, NY 10121-0701, USA
Phone: +1-212-626-0658
Email: acmbooks-info@acm.org



**Morgan & Claypool
Publishers**
1210 Fifth Avenue, Suite 250
San Rafael, CA 94901, USA
Phone: +1-415-462-0004
Email: info@morganclypool.com



DOI:10.1145/3158227

Once a meme gets popular, it will have to evolve to keep being popular.

BY MICHELE COSCIA

Popularity Spikes Hurt Future Chances for Viral Propagation of Protomemes

A MEME IS a concept introduced by Dawkins¹² as an equivalent in cultural studies of a gene in biology. A meme is a cultural unit, perhaps a joke, musical tune, or behavior, that can replicate in people's minds, spreading from person to person. During the replication process, memes can mutate and compete with each other for attention, because people's consciousness has finite capacity. Meme viral spreading causes behavioral change, for the better,

as when, say, the "ALS Bucket Challenge" meme caused a cascade of humanitarian donations,^a and for the worse, as when researchers proved obesity⁷ and smoking⁸ are socially transmittable diseases. A better theory of meme spreading could help prevent an outbreak of bad behaviors and favor positive ones.

Studying memes in general is difficult because detecting and measuring them objectively is difficult. The inception of the web 30 years ago made it easier to focus on a subtype of meme—the one shared through social media. Researchers have since focused on the effect of timing, social networks, and limited user attention rather than on meme content.^{6,15,23} Being timely and shared by actors in key positions of a social network explains a large portion of a meme's virality. While these factors can help explain the meme ecosystem at large, both are exogenous to the meme itself. Even if endogenous meme characteristics have less compelling predictive power over meme popularity, it is still important to understand their relationship with virality, as such understanding can be applied case by case to specific memes, rather than produce only a description of the general mechanics of the overall system. For example, we¹¹ showed that success eschews similarity; successful memes are generally found at the periphery of meme-similarity research. The more a

a <http://www.alsa.org/fight-als/ice-bucket-challenge.html>

» key insights

- We test the hypothesis that protomemes in social media are less likely to be popular when they are too similar to other protomemes in a large, variegated dataset.
- We calculate the canonicity of each specific use of a protomeme, or how typical or common is its use.
- We show that canonicity has a non-linear relationship with protomeme popularity, increasing the similarity between protomemes and genes, in that not all mutations are beneficial for propagating protomemes.

A close-up photograph of Gene Wilder as Willy Wonka. He is wearing a brown top hat, a purple velvet suit jacket, a white shirt, and a large, light-colored bow tie. He has a slight, knowing smile and is resting his head on his right hand. The background is a soft, out-of-focus green.

**OH, YOU'RE A
NEW MEME?**

**I'M SURE YOU'LL BE
POPULAR FOREVER.**

meme is imitated, the less the original meme (and all its imitations) will be successful in going viral in the future.

Here, we focus on protomemes, or all catchphrases used frequently on social media that may or may not end up being adopted as memes. We aim to extend our earlier work¹¹ by expanding the scope of the dissimilarity-driven success theory; providing new evidence of the effects of the theory in a dynamic environment; and introducing canonicity to evaluate protomeme content.

First, we test the dissimilarity-driven success theory on a larger data source and on shifting scholarly attention from memes to protomemes. The original work¹¹ focused on a small site and very specific subtype of Internet meme. Here, we focus on Reddit and *Hacker News*, two online social media websites, and consider any kind of protomeme in the form of frequent and regularly used n-grams that can be shared on these websites, rather than limit ourselves to the image-macro meme subtype studied previously.¹¹

Second, we test the effect of popularity spikes on the future popularity of a protomeme, following Weng et al.,²¹ as we did not study spikes in the earlier work.¹¹ When a protomeme suddenly becomes very popular, or when it places highly in a ranking of user appreciation, many participants in social media use it in their own posts. The increased number of posts that include the protomeme will make it more similar to the average post of the day. As a consequence, as reported by Lakkaraju et al.,¹⁷ its expected popularity will decrease. The new posts that include the popular protomeme are then poorly ranked, as they are stealing each other's chance for success.

Lastly, we introduce a measure of “canonicity,” or capturing the amount of change introduced by a post compared to previous posts with the same protomeme, showing that the more different a post is from the canonical usage of a popular protomeme, the greater its odds of going viral. However, the effect of canonicity is not linear. High canonicity lowers the overall success of viral posts at the same time it helps non-viral posts be appreciated; that is, we correct the earlier work,¹¹ arguing here that there is a non-linear relationship between canonicity and

meme propagation. This is why other research did not find content to be a powerful predictor of success; while true that success eschews similarity, it is not true that with dissimilarity comes success. One explanation might be that protomemes follow the same dynamics as genes. Not all mutations are beneficial; some are irrelevant and some harmful. Future research will test this theory.

Related Approaches

Here, we follow the rich literature studying memes as they spread through the web. In our previous work,¹¹ we observed traditional meme dynamics (such as competition and collaboration in the context of the web¹⁰) and provided evidence for the theory that similarity with existing content penalizes a meme's odds of viral success.¹¹ The origin of this line of research is due to questions raised in the broad distribution of meme popularity observed multiple times in different contexts,^{15,23} consistent with the dynamics of fads.²

Many research projects have sought to model and predict meme success, and we are unable to mention them all here. The most successful research track focuses on the relationship between memes and the social media through which they spread. For example, social media researchers can predict how wide the cascade spread of a meme will be by observing its temporal and structural features; for example, a cascade's initial breadth, rather than its depth, is a better predictor of larger cascades.⁶ The community structure of a social network, or its tendency to form densely connected groups, influences the dynamics of news and meme spread.^{19,24}

The general conclusion of other work is that content is a secondary explanatory factor for meme propagation. However, none of them has effectively ruled out meme characteristics as partial explanation for their success. Further, it is interested in using networks to explain the shape of popularity distributions, not what makes individual memes more or less fit to go viral, which was the focus of our earlier work,¹¹ as well as of this article. Different hashtags in Twitter reflect different degrees of persistence, showing evidence that meme propagation is a

complex form of contagion, rather than a simple epidemic.²⁰ Beyond computer science, social science researchers have found links between a meme's content and its virality. Emotionally positive content is more viral than negative content, although valence alone is just one of many factors driving propagation.³ All such studies inevitably face the challenges of complex spread dynamics, as content that eventually becomes popular might be overlooked when it first appears.¹⁴

Our research perspective focuses on the role of mutation and innovation in meme usage and their relationship with success spikes;²¹ for example, Adamic et al.¹ explored an example showing the dynamics of meme mutation. Some results in the literature support the role of novelty in content diffusion,⁵ while others questioned this result,¹⁶ though in neither case was “novelty” strictly defined. Here, we build on the evidence of a negative relationship between title similarity and success on Reddit.¹⁷ The difference is that Lakkaraju et al.¹⁷ considered how similar an instance of a meme is to the past submission history of the thematic community in which it was shared. By introducing the measure of canonicity, we focus here instead on the similarity of the meme instantiation with the meme's own submission history, regardless of the context in which it was previously observed. Canonicity is closely related to the Newsjunkie framework,¹³ which is based on an information-theoretic background. However, Newsjunkie was developed with a different application—ranking novelty of full-text news articles significantly longer than Reddit and *Hacker News* post titles. For this reason, Newsjunkie is not applicable to this research scenario concerning novelty.

Protomemes

We now reflect on data collected by Wenginger et al.²⁵ from Reddit, a social-bookmarking website where users are encouraged to post interesting content. Every post can be upvoted (or downvoted) if the user likes (or dislikes) it. The upvote/downvote ratio is used to highlight high-quality content. In addition, there is a time discount; that is, no matter how many upvotes a post manages to attract, it cannot be


highly visible forever. The most popular (highest upvote/downvote ratio) posts appear on Reddit's "front page," giving them a further boost in visibility. By default, the front page hosts 25 posts. Each entry in the dataset we studied consisted of a post, its title, and its number of upvotes/downvotes that were combined in a post score by Reddit's sorting algorithm. Note our research can observe only the final score of a post, not its full upvote timeline. This might introduce bias when looking to establish whether or not the post hit the front page. We assume the final post score is highly correlated with the post score on its first day of life. We base this assumption on the fact that the vast majority of upvotes come within 24 hours following post submission.

Note the terms "score" and "popularity" are not interchangeable, as they refer to related but different concepts. Score is the one-off measurement of a single instance of a protomeme in a day, and popularity is the overall success of all instances of all memes over a longer period of time.


All 22,329,506 posts added to Reddit from April 5, 2012 to April 26, 2013 were part of the dataset. To cross-test our results, we also used a similar dataset from *Hacker News*, which uses the same dynamics as Reddit though focuses on a more specialized technical audience and has a much smaller user base. The *Hacker News* dataset included 1,194,436 posts from January 7, 2010 to May 29, 2014.

Here, we consider protomemes, or a catchphrase with the potential for going viral. Note there are more possible types of memes (such as pictures and videos), but given the nature of the data we limit ourselves here to catchphrases. Catchphrases were also used as meme proxy in Memetracker¹⁸ and Nifty.²² We extracted them using information taken exclusively from the post title.

We apply our definition by borrowing the bag-of-words methodology from the text-mining literature, meaning a protomeme is seen as a set of at least two "tokens" (also called an "n-gram" with $n \geq 2$). A token is a word that is stemmed whereby "stop words" are filtered out and are not tokens. To be classified as a protomeme, an n-gram must have been used frequently and constantly over the observation period.



The more a meme is imitated, the less the original meme (and all its imitations) will be successful in going viral in the future.



We used the frequent-itemset-mining algorithm Eclat^a to extract the frequent n-grams and discard an n-gram if it has not been used for a certain number of days. We also discarded all n-grams that are proper subsets of another n-gram.

We performed the analysis using different thresholds to ensure independence between parameter choice and results. From most- to least-restrictive threshold choices, we obtained 2,731 to 5,585 protomemes on Reddit and 817 to 2,538 protomemes on *Hacker News*; the preprocessed data we used is available for result replication.^b

Results

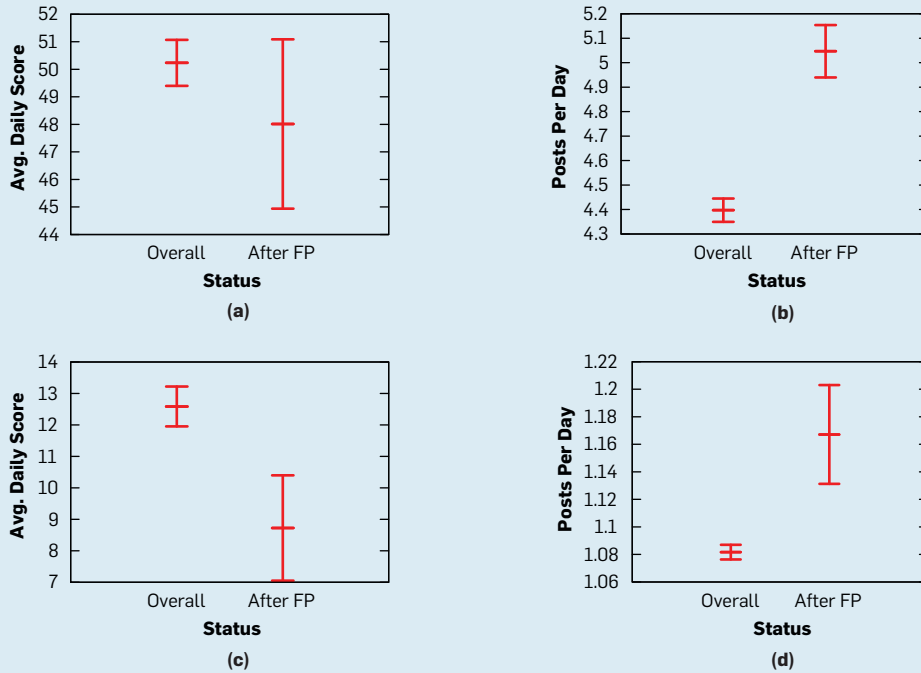
We now first show how popularity spikes result in reduced future popularity for a meme, then introduce the concept of "canonicity" and how it can shed light on this phenomenon.

Popularity curse. Common sense tells us that popular ideas are likely to be imitated; a protomeme used in a very popular post today will be used in many posts tomorrow. Such intuition about the demand-supply relation on the web is corroborated by several studies, including Ciampaglia et al.⁹ However, dissimilarity-driven success theory would predict that flooding a system with imitations of a protomeme will cause the imitations to be less popular. At first glance, such a prediction might seem to find support in two observations: the average score of the posts containing a protomeme is less than expected the day after it experiences a popularity spike (see Figure 1a and Figure 1c), and the number of posts containing that protomeme increases (see Figure 1b and Figure 1d).

These observations support our theory about viral connections but do not prove it. First, the total score awarded and the average score per post are not constant over time (see the online appendix, dl.acm.org/citation.cfm?doid=3158227&picked=for mats). The lower score might be just a relative change; if, for example, there are fewer upvotes awarded on that particular day, a lower absolute number could still represent an increase in upvote share for the day. Second, each protomeme is characterized by its own expected populari-

^b http://www.michelecoscia.com/?page_id=870

Figure 1. Distributions of score and number of posts per day of the observed memes, overall, and focusing on only those created the day after the meme was among the 25 top-scoring posts; included is the average and 95% confidence intervals.



ty; scores cluster around each protomeme’s average. Third, a protomeme’s recent history might explain this variation. If, for example, a protomeme was getting declining scores, we might expect it to get even lower scores after a random popularity spike. Finally, there is a fat tail in score distributions, so the average alone is not meaningful.

To give more solid evidence for the theory about viral connections, we tested the median popularity of a protomeme on a particular day using the following mixed model, or “MED Model”

$$E_{w_{m,i}}(p_{m,i}) = \alpha + \beta_1 F P_{m,i-1}^l + \beta_2 E(p_{m,i-1}) + u_m + \varepsilon_{m,i}.$$

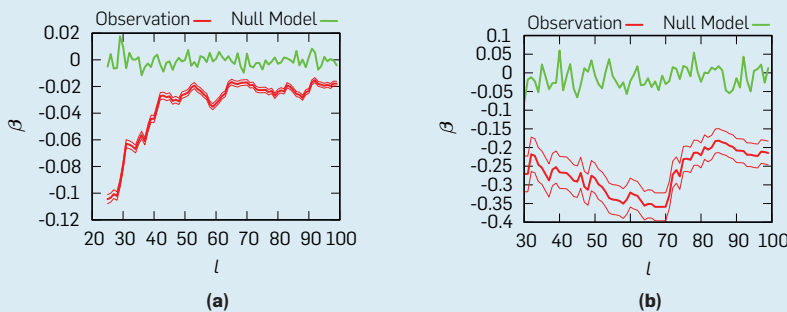
An observation ($E_{w_{m,i}}(p_{m,i})$) is the median popularity ($p_{m,i}$) of the posts containing protomeme m on day i , or the score of the post. Since post scores are count data and distributed over a skewed distribution, we used a Poisson mixed model. Each observation has weight $w_{m,i}$, or the number of posts containing m on day i , thus giving less weight to the (presumably noisy) information on protomemes m that were not used very much on day i .

$FP_{m,i-1}^l$ registers if protomeme m was on the front page on day $i - 1$. The parameter l is the number of posts hitting the front page each day. The default front page in Reddit includes 25 posts (30 for *Hacker News*). So, every day, at least 25 (30) posts hit the front page. However, users can increase the physical length of the front page. Moreover, as the day passes, front-page posts get replaced by other posts. As a result, there is no way to know how many posts hit any particular user’s front page on any given day (see the online appendix). For this reason, we ran the regression multiple times for different l values to test the robustness of our results for different front-page sizes.

$E(p_{m,i-1})$ denotes the protomeme’s popularity on the day before i , controlling for existing trends; u_m is a random effect of protomeme we used to control for the fact that different observations can refer to the same protomeme; and $\varepsilon_{m,i}$ is the error term.

Figure 2 outlines our estimates of β_1 for different values of l . The effect of being on the front page is negative; the protomeme is expected to have a lower median score than on a business-as-usual day. This expectation confirms the theory of dissimilarity-driven suc-

Figure 2. Evolution of β_1 coefficients for increasing l in the model and in its associated null model; thin lines represent 95% confidence intervals.



cess theory. Higher l values decrease the estimated effect, because we included lower-ranked posts that might not actually have hit the front page (all p -values are significant, with $p < 0.001$). Focusing on Reddit (see Figure 2a) values fall within the $]-0.1 : -0.02[$ interval. A value of -0.1 implies a score-reduction factor equal to $e^{-0.1}$, which is close to 10%. This means that ranking in the top 25 posts on a particular day reduces the next day's median score of a protomeme by almost 10%. The effect in *Hacker News* appears to be even stronger.

We obtained these results with fixed frequency thresholds, using Table 1 and Table 2 to show the robustness of the results with different threshold choices. In them, we fix $l = 25$ for Reddit and $l = 30$ for *Hacker News*. For all threshold choices in Reddit and for most threshold choices in *Hacker News*, the results were consistently negative and significant with our main result: hitting the front page results in lower expected popularity on the following day.

One could object to these results using the regression-toward-the-mean argument; that is, once the very visible front-page protomeme instance is copied many times, each copy tends to score approximately the protomeme's average, which is lower than the spike. But the regression already corrects for this using the protomeme random effects. If the argument would be true, β_1 would equal 0. In fact, the average is 0 calculated over 50 null models, where protomeme scores are generated randomly, preserving each protomeme's average score and standard deviation (see Figure 2a and Figure 2b), disproving the regression-toward-the-mean argument.

To summarize, after hitting the front page, a protomeme will likely be used more frequently—15% more in Reddit, 8% more in *Hacker News*, as reported in Figure 1b, and Figure 1d. β_1 values in the model suggest that posts including this protomeme will likely have a lower score (10% lower in Reddit, 23% lower in *Hacker News*), confirming the expectation. The effect is significant and independent of the recent overall history of the protomeme, changes in average post score, and front page size.

So, if hitting the front page is bad for

subsequent protomeme posts, why does common sense tell us the opposite? We propose that a protomeme appearing on the front page two days in a row is very noticeable, and we just do not realize that, on average, the protomeme is doing poorly. We run the same regression, changing the target variable to the maximum score (MAX Model) instead of the median. In this model, for Reddit, the sign β_1 is the opposite of the β_1 sign for the MED Model (see the online appendix). If protomeme m hits the front page on day $i - 1$, the top-scoring post containing protomeme m on day i improves. This does not happen for *Hacker News*, and our hypothesis is that *Hacker News* is more resilient to fads, as it is used mostly for professional purposes, rather than humor, as with Reddit.

Hitting the front page is thus associated with a larger number of subsequent posts including protomeme m , which by itself is associated with less expected popularity for the same posts. However, in some scenarios, the best-ranked posts containing protomeme m can still hit the front page more easily than usual. We now turn our attention to this subset of special posts, explaining why they

are able to overcome the popularity curve predicted by the theory.

The Canon Effect

Following the argument that success is associated with dissimilarity, we now hypothesize that a post including a widely used protomeme m the day after m hit the front page can still be successful if it is dissimilar from all other posts using m . In this way, the post is able to attract most of the attention network users are directing toward protomeme m .

To test this claim we first need a measure for the uniqueness of a post. In Coscia,¹¹ we proposed a meme similarity measure that cannot be used here because it calculates meme-meme similarity, while here we consider only post-post similarity within the same protomeme. Moreover, our earlier¹¹ measure applies only to a subtype of the memes shared on Reddit. We cannot use the measure developed by Lakkaraju et al.¹⁷ because it measures the similarity of a post to the subcommunity it is shared in, ignoring the meme it implements. Here, we focus on Reddit and expect a null result for *Hacker News*, as we showed it to be less prone to the fad effect.

Table 1. Effect on β_1 of different threshold options for the Reddit dataset. Each row is a different frequency threshold, or minimum share of posts that must include the protomeme, or 0.004 = 0.4% of posts. Each column is the minimum share of days in which at least one post including the protomeme appeared, or 0.91 = 91% of days. Significant values with $p < 0.01$ are in bold. The threshold values included in Figure 2a are in italic.

sup-end	0.91	0.92	0.93	0.94	0.95
0.0045	-0.095	-0.087	-0.101	-0.075	-0.039
0.0045	-0.095	-0.087	-0.100	-0.075	-0.039
0.005	-0.109	-0.098	-0.104	-0.080	-0.046
0.0055	-0.093	-0.087	-0.079	-0.071	-0.038
0.006	-0.064	-0.056	-0.055	-0.061	-0.043

Table 2. Effect on β_1 of different threshold choices for the *Hacker News* dataset. Rows and columns are interpreted, as in Table 1, and significant values with $p < 0.01$ are highlighted in bold. The threshold values in Figure 2b are in italic. We chose the different threshold values to accommodate the significant difference in size between the two datasets.

sup-end	0.02	0.03	0.04	0.0433	0.0466
0.0055	-0.125	-0.099	-0.271	-0.138	-0.294
0.006	-0.173	-0.152	-0.284	-0.138	-0.294
0.0065	-0.214	-0.199	-0.284	-0.135	-0.294
0.007	-0.129	-0.109	-0.129	-0.101	-0.239
0.0075	0.007	0.019	0.007	0.009	-0.132

We introduce the concept of canonicity of a post, measuring how much a post containing a protomeme m differs from the usual usage of m . A post is said to be canonic if it uses m as expected, without introducing elements not strongly associated with m itself. Consider, for example, a post M as a bag of words. Each word μ in the bag of words co-occurs with protomeme m with a given probability $\pi_{m,\mu}$. If m appears in 100 posts, and in 30 of them the post title also includes μ , then $\pi_{m,\mu} = 0.3$. The canonicity of M is calculated like this

$$\Gamma(M,m) = \sum_{\nu \in M} \pi_{m,\nu} / |M|.$$

The formula means the canonicity of a post M is the average probability of its words to appear with the meme it contains. Note some posts contain no other word than the words of the protomeme m itself. For this reason, the formula includes the m words in M . Otherwise, such posts will have $\Gamma(M,m) = 0/0$, which is unacceptable. Moreover, posts including only a protomeme's words must have $\Gamma(M,m) = 1$ because they use m in its purest form. Since the protomeme's μ s

always appear in posts containing the protomeme itself, their $\pi_{m,\mu}$ always equals 1. Finally, a low canonicity score is obtained when there are many words in the post *and* they have low $\pi_{m,\mu}$. If a post includes only one unusual word, its canonicity score is still high, because it is still composed mostly of the protomeme itself.

How canonicity is distributed in Reddit is reported in the online appendix. To test the connection between canonicity and popularity for posts using protomemes appearing on the front page the previous day, we create a rank binary variable ϕM that records whether or not the post was among the 5% best-scoring posts of the day. This is the target variable of the following logistic regression

$$\phi M_i = \alpha + \beta \Gamma(M_i,m) + u_m + \varepsilon_m.$$

This is the ϕ Model. Given a post M containing protomeme m , the ϕ Model estimates its probability of experiencing a popularity spike on day i , after m hit the front page on day $i - 1$. Note the set of posts we include in the model is still dependent on the l parameter. For

different l values the set of posts included is different, because, for increasing front page size, more protomemes hit the front page, and more posts on the day after will thus be considered in the model.

Figure 3 reports ϕ Model's β s for increasing l . For Reddit (see Figure 3a), β never takes values greater than -0.7 , suggesting a noticeable and notable effect: high canonicity halves the odds of being a high-scoring post; for a deeper discussion see the online appendix. As we increase l , the canonicity effect gets weaker and weaker. This is expected, as we are considering the regression posts that might have not hit the front page. All β values in Figure 3a are significant ($p < 0.0001$). We thus expect a null result in *Hacker News*, given the result of the MAX Model covered earlier. Indeed, Figure 3b reports the effect of canonicity in *Hacker News* is zero, as no p-value reported for any l is less than 0.01.

We also run two Poisson mixed models with the same form of the ϕ Model, with the only difference being the dependent variable (in this case the post score) and the data included in them. In the Zero Model, we consider only the posts for which $\phi M_i = 0$, while in the One Model we focus on the posts for which $\phi M_i = 1$. In practice, the ϕ Model tells us the effect of canonicity on the odds of experiencing two popularity spikes in a row, while the One and Zero models reveal the score effect of canonicity on the posts that did and did not experience two popularity spikes in a row.

In the One Model, β has a negative sign (see Figure 4a); all β s are significant with $p < 0.0001$. If the ϕ Model told us that canonicity lowers the odds of experiencing two popularity spikes in a row, the One Model would tell us that if a post can nevertheless overcome those odds, it is additionally penalized with a worse score. In the Zero Model (see Figure 4b), β is positive and significant. For the unsuccessful posts in the Zero Model, canonicity has a positive effect. For robustness, we also ran a negative binomial model, resulting in similar estimates as the Poisson model (see the online appendix).

The discordance of β signs in the Zero and One models can be interpreted as a similarity between protomeme

Figure 3. Distribution of the ϕ Model's β for varying l ; thin lines represent the 95% confidence intervals.

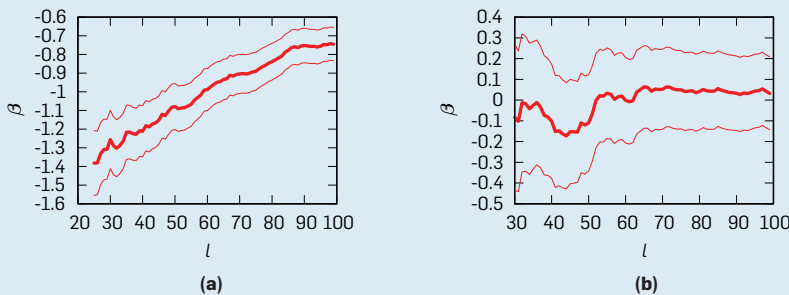
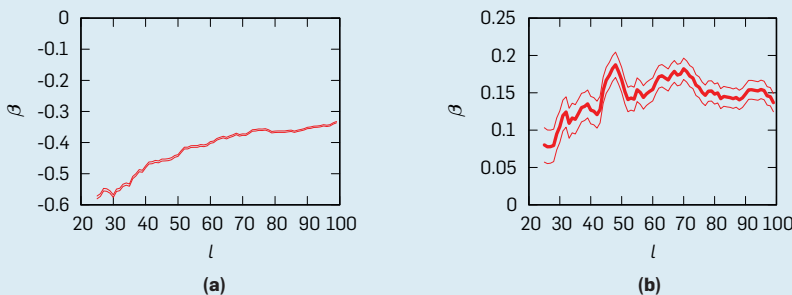


Figure 4. Distribution of the One Model's and Zero Model's β for varying l ; thin lines represent the 95% confidence intervals.



and gene dynamics. Most mutations are harmful or irrelevant, while also lowering an organism’s fitness. In protomemes, if the change is not judged “suitable” for the protomeme by the user community, it will be selected against and gradually lose relevance. This is one of many possibilities and must be properly tested before it can be considered suitable. We leave such a test for future work. However, this result could explain why meme content is not a promising predictor of meme popularity.⁶ Since changing meme content can go both ways, increasing or decreasing meme fitness, the effects might cancel out.

Conclusion

We have tested some of the predictions of a theory that claims that meme success eschews similarity, because similar memes interfere with one another and get less attention.¹¹ We tested the theory on Reddit and *Hacker News*, two popular social-bookmarking websites. Successful posts can hit each site’s highly visible front page and then be copied many times over by people who want to use them to be able to get their own posts to appear on the front page. The expected popularity of these posts should thus decrease. We showed that this is the case, though on Reddit some posts might still experience subsequent popularity spikes; *Hacker News* appears to be resilient to this phenomenon. We explain this apparent contradiction by showing these posts (with persistent popularity spikes on Reddit) have low canonicity; that is, they are usually dissimilar from the average post containing their protomeme. We showed that canonicity has a non-linear effect.

These results open the way to future work. First, computational social scientists can now move the theory closer to practice, performing, say, a controlled experiment where they select front-page memes from Reddit and semiautomatically generate imitating posts with varying degrees of canonicity. By releasing the posts on Reddit, they should observe in which cases low-canonicity posts tend to garner more upvotes and in which cases high canonicity is helpful. And second, the theory makes claims that are not in line with another

theory of meme popularity—the one giving factors other than meme content greater weight in predicting its success. In Cheng et al.,⁶ Gleeson et al.,¹⁵ and Weng et al.,²³ meme content and structure were found to be a weaker explanatory factor for meme popularity. Better predictors are meme timing and the social network position of the meme creators. We thus recommend reuniting the two theories in a unified meme-analysis framework.

Finally, computational social scientists could extend the investigation of memes by studying the effect of negative votes: we expect it will show non-trivial dynamics; a vote, even if negative, still comes from a person paying attention to the concept, though its effect is to prevent other people from seeing it. This information was not available at the time of our study, but Reddit started to provide it in 2017. □

References

1. Adamic, L., Lento, T., Adar, E., and Ng, P. Information Evolution in Social Networks. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining* (San Francisco, CA, Feb. 22–25). ACM Press, New York, 2016.
2. Bauckhage, C., Kersting, K., and Hadji, F. Mathematical models of fads explain the temporal dynamics of Internet memes. In *Proceedings of the Seventh AAAI International Conference on Weblogs and Social Media* (Cambridge, MA, July 8–11). Association for the Advancement of Artificial Intelligence, Palo Alto, CA, 2013.
3. Berger, J. and Milkman, K.L. What makes online content viral? *Journal of Marketing Research* 49, 2 (Apr. 2012), 192–205.
4. Borgelt, C. Efficient implementations of Apriori and Eclat. In *Proceedings of the International Conference on Data Mining* (Melbourne, FL, Nov. 19–22). IEEE Computer Society, New York, 2003.
5. Cha, M., Haddadi, H., Benevenuto, F., and Gummadi, P.K. Measuring user influence in Twitter: The million-follower fallacy. In *Proceedings of the AAAI International Conference on Weblogs and Social Media* (Washington, D.C., May 23–26). Association for the Advancement of Artificial Intelligence, Palo Alto, CA, 2010, 30–38.
6. Cheng, J., Adamic, L.A., Dow, P.A., Kleinberg, J.M., and Leskovec, J. Can cascades be predicted? In *Proceedings of the World Wide Web Conference* (Seoul, Korea, Apr. 7–11). ACM Press, New York, 2014, 925–936.
7. Christakis, N.A. and Fowler, J.H. The spread of obesity in a large social network over 32 years. *The New England Journal of Medicine* 357, 4 (July 2007), 370–379.
8. Christakis, N.A. and Fowler, J.H. The collective dynamics of smoking in a large social network. *The New England Journal of Medicine* 358, 21 (May 2008), 2249–2258.
9. Ciampaglia, G.L., Flammini, A., and Menczer, F. The production of information in the attention economy. *Scientific Reports* 5, 2015; <https://www.nature.com/articles/srep09452>
10. Coscia, M. Competition and success in the meme pool: A case study on quickmeme.com. In *Proceedings of the Seventh AAAI International Conference on Weblogs and Social Media* (Boston, MA, July 8–10). Association for the Advancement of Artificial Intelligence, Palo Alto, CA, 2013.
11. Coscia, M. Average is boring: How similarity kills a meme’s success. In *Scientific Reports* 4, 2014; <https://www.nature.com/articles/srep06477>
12. Dawkins, R. *The Selfish Gene*. Oxford University Press, Oxford, U.K., 1976.
13. Gabrilovich, E., Dumais, S., and Horvitz, E. Newsjunkie:

- Providing personalized newsfeeds via analysis of information novelty. In *Proceedings of the 13th International World Wide Web Conference* (New York, May 17–22). ACM Press, New York, 2004, 482–490.
14. Gilbert, E. Widespread underprovision on Reddit. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work* (San Antonio, TX, Feb. 23–27). ACM Press, New York, 2013, 803–808.
15. Gleeson, J.P., O’Sullivan, K.P., Baños, R.A., and Moreno, Y. Determinants of meme popularity. arXiv, 2015; <http://cosnet.bifi.es/wp-content/uploads/2015/02/1501.05956v1.pdf>
16. Harrigan, N., Achananuparp, P., and Lim, E.-P. Influentials, novelty, and social contagion: The viral power of average friends, close communities, and old news. *Social Networks* 34, 4 (Oct. 2012), 470–480.
17. Lakkaraju, H., McAuley, J.J., and Leskovec, J. What’s in a name? Understanding the interplay between titles, content, and communities in social media. In *Proceedings of the Seventh International Conference on Weblogs and Social Media* (Boston, MA, July 8–11). Association for the Advancement of Artificial Intelligence, Palo Alto, CA, 2013.
18. Leskovec, J., Backstrom, L., and Kleinberg, J. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Paris, France, June 28–July 1). ACM Press, New York, 2009, 497–506.
19. Nematzadeh, A., Ferrara, E., Flammini, A., and Ahn, Y.-Y. Optimal network modularity for information diffusion. *Physical Review Letters* 113, 8 (Aug. 2014).
20. Romero, D.M., Meeder, B., and Kleinberg, J. Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on Twitter. In *Proceedings of the 20th International Conference on World Wide Web* (Hyderabad, India, Mar. 28–Apr. 1). ACM Press, New York, 2011, 695–704.
21. Santì, C. and Lambiotte, R. Local variation of hashtag spike trains and popularity in Twitter. arXiv, 2015; <https://arxiv.org/abs/1503.03349>
22. Suen, C., Huang, S., Eksombatchai, C., Sosic, R., and Leskovec, J. Nifty: A system for large-scale information flow tracking and clustering. In *Proceedings of the 22nd International Conference on World Wide Web* (Rio de Janeiro, Brazil, May 13–17). ACM Press, New York, 2013, 1237–1248.
23. Weng, L., Flammini, A., Vespignani, A., and Menczer, F. Competition among memes in a world with limited attention. *Scientific Reports* 2, 2012; <https://www.nature.com/articles/srep00335>
24. Weng, L., Menczer, F., and Ahn, Y.-Y. Predicting successful memes using network and community structure. In *Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media* (Ann Arbor, MI, June 2–4). AAAI Press, Palo Alto, CA, 2014.
25. Weninger, T., Zhu, X.A., and Han, J. An exploration of discussion threads in social news sites: A case study of the Reddit community. In *Proceedings of the International Conference on Advances in Social Network Analysis and Mining* (Niagara Falls, ON, Canada, Aug. 25–28). IEEE Computer Society, New York, 2013, 579–583.

Michele Coscia (michele_coscia@hks.harvard.edu) is a research fellow in the Center for International Development at Harvard University, Cambridge, MA.

The author thanks Clara Vandeweerd, Frank Neffke, and Andres Gomez for useful discussions on the statistical methodology. This research is partly supported by the Walloon National Fund for Scientific Research (FNRS), grant #24927961.

DOI:10.1145/3158333

In a decentralized marketplace, buyers and sellers transact directly, without manipulation by intermediary platforms.

BY HEMANG SUBRAMANIAN

Decentralized Blockchain-Based Electronic Marketplaces

E-COMMERCE MARKETPLACES—WHETHER business-to-consumer (B2C) or business-to-business (B2B)—are examples of a two-sided market.^{6,19} Each side involves networks of participants. Network effects—the incremental value added by each new participant—play the dominating technological role in such markets. Over time, network effects inevitably yield a monopoly in which a single e-commerce firm manages the entire marketplace.²⁴

In line with the “theory of the firm”—or a set of economic theories that explains and predicts the nature of the firm, company, or corporation, including its existence, behavior, and structure—most e-commerce firms today aim to maximize profit for shareholders



» key insights

- Decentralization alters the paradigms of firm-controlled marketplaces by providing security, trust, privacy, lower transaction costs, and transaction integrity.
- Blockchain-based marketplaces improve matching, transaction facilitation, and support for institutional infrastructure.
- Blockchain-based e-marketplaces open new frontiers in prediction markets, financial engineering, automated reasoning systems, and, smart-contracts-based systems.

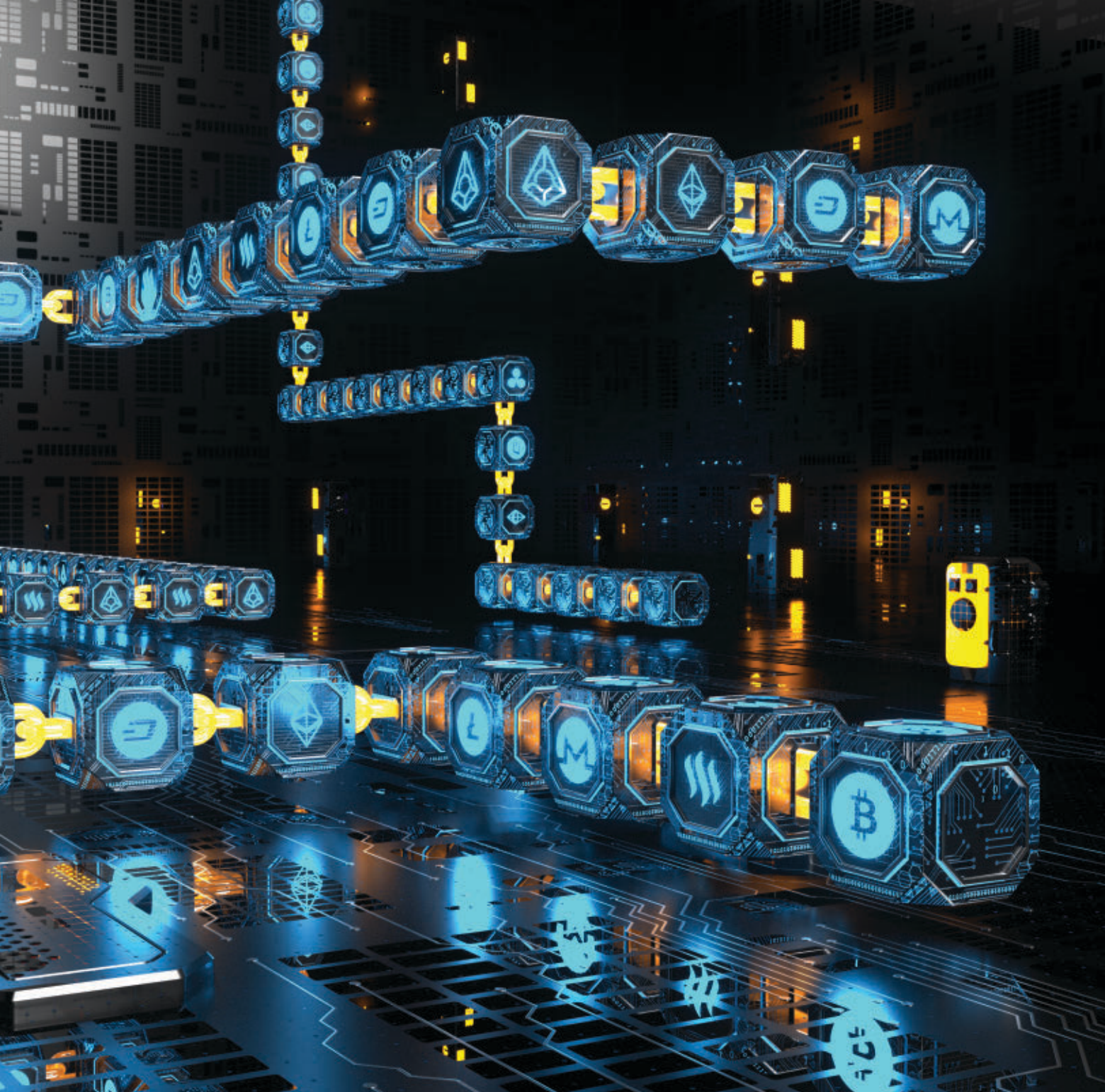


ILLUSTRATION BY SPOOKY POOKA AT DEBUT ART

by growing the corresponding network of participants using the platform, including sellers, buyers, developers, resellers, intermediary service providers (such as logistics providers), payment-gateway services, and institutional intermediaries, including legal advisors.

Strategies deployed by electronic marketplaces (e-marketplaces) to increase network effects include personalization of service offerings on the platform; recommendation systems for goods and services, trust mecha-

nisms, and simplification of transactions. Here, I describe an alternative to the firm-controlled marketplace—a decentralized marketplace based on the blockchain.

The blockchain is a shared, distributed transaction ledger that records all transactions and operates through the Bitcoin protocol.^{3,14} Each transaction is validated by a network of nodes, each hosting the blockchain and corresponding validation software. The validation algorithms confirm the trans-

action by preventing problems (such as “double spend,” whereby a given amount of coins is spent more than once in the same transaction). Once the transaction is confirmed, its details are stored on a public ledger generated through an algorithmic process called “mining.” A network of nodes, each with a local copy of the blockchain, provides an alternative to a centralized platform in which each node can maintain functions of the larger platform, partly or fully.

The blockchain supports several key functions:

Distributed storage and listings. A network of nodes lists items offered through the marketplace by individual businesses, eliminating single-point-of-failure scenarios and preventing a single controller firm from manipulating the shared central database;

Transactional validity. Fraudulent and duplicate transactions are prevented through timestamp-based validation;

Transactional persistence. All transactions concerning an asset or service traded in the marketplace are stored on a publicly accessible and verifiable ledger;

Transactional anonymity. The true identity of marketplace participants is hidden from other participants and the network by allowing users to create multiple wallets to be used on the network for transactions;¹⁸

Transactional privacy. Transaction details are hidden from the network (though the ledger is public) by the blockchain automatically encrypting transactions;

Transactional traceability. Each transaction can be traced back to the sender's and the receiver's true identities through a combination of techniques (such as IP tracing and blockchain graph analysis).¹⁸ Traceability is used by government regulators (or analysts) to detect theft, money laundering, and other illegal activity on the network, and can be both computationally and economically expensive, depending on blockchain implementation; and

Transactional immediacy. A mechanism built into the network consummates each transaction in the shortest possible time, with many implementations of the blockchain achieving instant validation through "proof of service,"¹⁴ "consensus,"²¹ and "proof of stake."²²

Most important, the blockchain eliminates the central authority needed to validate transactions, thus realizing many computational efficiencies; for example, transaction costs due to contract enforcement (such as following a sale) can be eliminated when the network validates the transaction. Likewise, the payment (good) transfer between buyer (seller) and seller (buyer) is recorded in a common, secure ledger.⁷ To illustrate such efficiency, consider a conventional e-commerce store. When a buyer

clicks the checkout button, multiple systems (such as payment and credit-card networks) each charge a fee. Despite such payment, fraudulent transactions are not necessarily always eliminated. In a decentralized e-marketplace, traders transact with each other securely and directly, and the network of nodes validates and records each transaction.³

Decentralized marketplaces using the blockchain as a foundational block are a viable alternative to firm-controlled marketplaces, yielding advantages from how decentralization supports marketplace functions, matching transaction support and institutional infrastructure.¹

Limitations of Firm-Controlled E-Marketplaces

Firms controlling e-commerce platforms support multiple networks, including customers, resellers, application developers, advertising partners, and financial intermediaries. Here, I address limitations with respect to firm-controlled marketplaces and the main marketplace functions.¹

Matching sellers and buyers. E-marketplaces thrive due to the network effects they facilitate when buyers, sellers, and third parties (such as resellers) trade with one another. Either a seller-driven (or marketplace-driven) promotion or customer-initiated search will facilitate matching. Buyers benefit because the marketplace reduces search costs. Sellers benefit because the marketplace offers product listings at no marginal cost and because of shared inventory and logistics costs. E-marketplaces facilitate reach (buyers can be located anywhere) and transaction immediacy (buyers/sellers can trade at any time).²²

E-marketplaces reduce search costs for buyers by efficiently listing and retrieving goods and services in databases. In addition to reducing such costs, marketplaces increase revenue by encouraging consumption by altering users' purchase preferences through product recommendations, bundling, and other product (or service) machinations. Likewise, B2B/B2C marketplaces facilitate customer credit to increase consumption. By providing APIs, marketplace platforms facilitate ease of integration with third-party

resellers or franchisees to further increase a platform's reach.

Matching characteristics of markets is not performed efficiently in today's B2B/B2C marketplaces; for example, price changes facilitated by algorithms can make certain goods pricier online compared to similar offerings sold through conventional brick-and-mortar stores. Likewise, price matching offered by brick-and-mortar retailers negate the price-based advantages of e-marketplaces. The behaviors of sellers and buyers affect an individual firm's decisions with respect to the marketplace and vice versa; for example, if firms controlling a marketplace decide to stop accepting certain payment methods, market participants would have to either alter their payment behaviors policies or stop transacting on the platform. If firms controlling the platform decide to provide differential pricing for similar products across customer segments based on profit-maximization algorithms, traders would gain (or lose) the ability to buy (sell) from (in) markets where price is low (high). Consider how the taxi-ride-hailing service Uber uses proprietary algorithms to determine ridesharing prices that may not account for an individual driver's actual profit margin. This has led to protests in cities with traditional taxi services drivers have found to not be profitable.⁵ When network effects are disruptive to the market, a monopolistic firm that aims to maximize its own profit can try to alter participant behavior, without necessarily improving the efficiencies that might otherwise be realized in an e-marketplace. With Uber, though the rider (customer) benefits due to lower prices compared to a conventional taxi service, the drivers (service providers) would be worse off due to discounted pricing as determined by the platform.

Facilitating transactions. Facilitating transactions is what a marketplace does to enable the exchange of value between buyers and sellers. The buyer pays the seller and the seller transfers the physical good (or service) to the buyer on the platform. A variable transaction cost is associated with each transaction due to banks, credit institutions, logistics providers, and other intermediaries. In most transactions, a legal entity ensures transaction valida-

tion by enforcing a legal contract.⁷

Trust plays a major role in any kind of transaction between a buyer and a seller on any marketplace platform.⁹ Seller reputation displayed on the platform influences the buyer and vice versa. Either third-party services (such as Yelp.com and the Better Business Bureau) or marketplace-controlled “ratings and reviews” systems inform users about the reputation(s) of sellers. However, no reputation system is foolproof, possibly being influenced by spam, tampered ratings and reviews, and paid reviews. Reputation systems themselves can thus be a major source of concern undermining trust.

Modern e-marketplaces are serious about privacy. Enforcing “privacy with security,” a feature all users want, is difficult on a centrally controlled platform. The seller’s and the buyer’s full identities, in addition to the transaction details, are disclosed to each other and possibly to others on the platform. Such disclosure has multiple uses, enabling, say, personalization algorithms to infer a user’s purchase behavior or a seller’s online behavior and target future promotions. Because such information can be misused, use of transaction information and personal information has been subject to much debate in Internet policy and law. Moreover, other marketplace participants (such as credit-validation services) use it to validate payment mechanisms or credit limits accessed by customers.

In a conventional marketplace, transactions are validated and confirmed through third-party agencies like credit-card companies using open-loop or closed-loop networks.¹² Personal information, including addresses, Social Security information, and credit-card details, has proven to be the most vulnerable source for security attacks. A single cyberattack on the database hosting personal information leads to disproportionate losses, including of trust by customers in the platform.¹⁷ Every marketplace (such as Amazon, eBay, Sony, and Target) in recent years has been targeted by at least one attack involving loss of information.

Other disadvantages can include scenarios in which transaction costs are greater than the actual sale price, limitations to payment modes, and network infrastructure challenges



In a decentralized e-marketplace, traders transact with each other securely and directly, and the network of nodes validates and records each transaction.



(such as sites unable to handle large volumes of transactions). Although e-marketplaces are geared for trade across geographies or national borders, B2B trade is limited due to friction related to currency-transfer laws and complex logistics. For example, if a buyer in the Middle East wants to buy music from the iTunes store and the payment gateway does not support international credit cards, the sale would be suspended, even though technology willingly supports the trade.

Facilitating institutional infrastructure. E-marketplaces traditionally enable contracts that are honored by participants through such mechanisms as click-wrap, shrink-wrap, and web-wrap enforcement. E-marketplaces validate transactions and enable automatic enforcement of contracts through exchange of payment and goods (or services). Likewise, e-marketplaces protect participants’ intellectual property by ensuring copyright laws are followed and counterfeit goods are prohibited, thus protecting brand value of the goods being traded.

Contracts in e-marketplaces for outsourced labor often involve significant transaction costs because contracts between vendors and customers are based on time and money or labor requirements. Renegotiating contracts increases transaction costs, precluding contractual support for agile project management (such as software development and architectural design) needed in the marketplace.

Blockchain-Based Decentralized Marketplaces

I now explore examples of such marketplaces, along with their architectures and potential business advantages. The Bitcoin cryptocurrency is the most widely used application of decentralization and can be viewed as a bearer bond in which each transaction accounts for value transfer between the two parties. Network participants, or “miners,” validate transactions through a process known as “mining.” Likewise, other participants host nodes that run the blockchain and “validate” transactions.

Ethereum is another blockchain-based protocol, enabling programmable contracts through distributed validation on the blockchain.²⁵ Once

transaction rules are agreed upon by a participating buyer and seller, those rules can be programmed into a contract to then reside on the blockchain. Nodes on the network are incentivized by “ether” rewards for validating and securing transactions on the blockchain. Such incentives have spawned e-marketplaces for contract-specific transactions (such as prediction markets and initial coin offers like the “decentralized autonomous organization”).

The Lazooz distributed ride-sharing network is another example of a decentralized marketplace in which customers sharing rides use a mobile application to order the ride. On the Lazooz network, individual participants produce a “Zooz” token used to compensate drivers. Each transaction, or ride, is recorded on the blockchain’s network of nodes.¹⁶

OpenBazaar is a decentralized marketplace in which software is installed on each seller’s node where listings are created. The marketplace accepts Bitcoin as its mode of payment and helps users trade with one another by reducing transaction costs compared to a conventional marketplace. The main advantage is that it offers participants

pseudo-anonymity of transactions, direct payment for goods and services via Bitcoin, and search that is unaltered by the marketplace. The marketplace’s quality of service is assured by independent third-party brokerage services. Likewise, reputation brokers maintain user reputations throughout the network.¹³

Multilayered platform. Figure 1 outlines the architecture of a decentralized e-marketplace architecture in which Layer 1 is the network infrastructure consisting of hardware nodes and client software. The client software provides listings of goods in which each node runs a local copy of the network’s blockchain that also includes its own product listings. Layer 2 is the mining software used to create new blocks of data consisting of network transactions; newer tokens of value are issued into the network based on mining algorithms. Layer 3 is the software responsible for validating transactions on the network and for storing validated transaction records. In layer 4, distributed applications might include a peer-to-peer marketplace or a seller-logistics marketplace (such as the electronic data interchange interface and reputation models). And Layer 5, or the quality-of-services layer, is where a marketplace’s customer relationship management functionality is implemented. Reputation models, designed to increase trust between buyers and sellers, are implemented through user feedback, ratings, and reviews. Likewise, dispute resolution is facilitated through third-party brokers. Search is also facilitated in this layer by third

parties or directly by the platform. Figure 2 outlines transactions in a decentralized marketplace.

Advantages of a Decentralized Marketplace

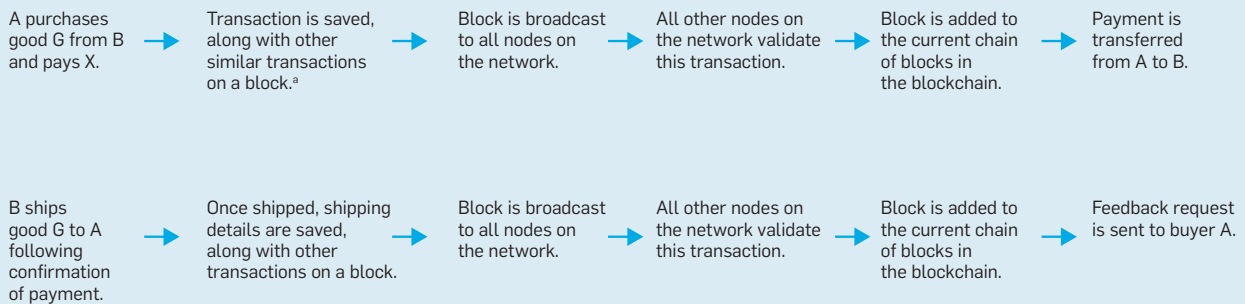
In a decentralized marketplace, the firm responsible for ensuring the marketplace functions properly by matching buyers and sellers, facilitating transactions, and/or enabling institutional infrastructure is replaced by a network of nodes, each independently and concurrently accomplishing the same functionality as that of a centralized marketplace. As in Figure 1, blockchain-based programs validate each transaction transparently and securely.^{14,25} Decentralized platforms ensure privacy and security for transactions while facilitating trust among platform participants.

Matching buyers and sellers. Decentralized marketplaces can provide unmodified “access” to information (such as listings) as desired by the seller, since each node is able to list prices, goods, and reviews pertaining to goods. The individual sellers are themselves responsible for creating the product listings that are then redundantly distributed throughout the network. Information transfers are completed much more reliably when search results pertaining to goods listed by the seller are unchanged. Likewise, listing errors are minimized by design, since price-altering and preference-altering algorithms can be disabled or managed by individual sellers. And matching functionality through search engines can exist independent of the marketplace; for

Figure 1. Multilayered architecture of a decentralized e-marketplace.

Layer 5	Quality of Service
Layer 4	Decentralized Applications
Layer 3	Blockchain
Layer 2	Mining Software
Layer 1	Network Infrastructure

Figure 2. Transactions flows in a decentralized marketplace.



^a The transaction here could piggyback on the Bitcoin network¹⁴ or follow a different blockchain implementation (such as Ethereum).

example, market platforms Duo Search and Bazaar Bay provide search listings for OpenBazaar.

Transactions. Transaction costs are minimized because intermediaries (such as payment gateways) are excluded. Buyers pay sellers directly, and the current network or other cryptocurrency networks validate payment transactions. Transactions are more secure, owing to the fact they cannot be manipulated by anyone in the marketplace. Transactional anonymity and transactional privacy reduce incentives for cyberattacks on both individual accounts and transactions. Micro-transactions (such as tipping and micropayments) are also facilitated. Disputes involving transactions are handled by third-party conflict brokers. The reputations of both sellers and buyers are thus managed independent of the marketplace. And returning goods and payments is handled in accordance with the mediation facilitated by brokers through such mechanisms as multi-signature contracts, notarization, and arbitration.

Institutional infrastructure. Programmable contracts facilitated through the network of nodes ensure buyers and sellers adhere to the rules and norms of a contract. Fully automated enforcement, partial automation, and manual enforcement modes for contracts thus ensure all participants in a transaction adhere to the negotiated, agreed-upon terms. Table 1 compares various features of decentralized e-marketplaces with those of their counterparts in traditional e-marketplaces.

Conclusion

Decentralized marketplaces provide many advantages to all market participants, including security, trust, privacy, lower transaction costs, and transaction integrity. Decentralization alters the paradigms of today’s conventional marketplaces in which a large intermediary firm that controls the platform is able to control every aspect of a trade, from product listings to price discovery, product search, logistics, and the customer experience.

While I have proposed an alternative to the existing firm-controlled marketplace, it is likely only certain functions within existing centralized

marketplaces would be better off decentralized. For example, transaction facilitation on many centralized platforms (such as Amazon and Overstock) accept cryptocurrencies as a payment option. However, in certain marketplaces (such as crowdfunding and venture-finance IPOs), initial coin offers can replace existing mechanisms to

provide a viable alternative with more functionality.²⁰ Smart contracts could be used to set up complex financial instruments with preset rules for paying dividends to investors. Table 2 outlines caveats concerning decentralization for a variety of e-marketplaces.

Decentralization is used to create a social network (such as Steem)

Table 1. Decentralized e-marketplaces vs. traditional marketplaces.

Marketplace Feature	Blockchain-Based Decentralized Marketplace	Traditional E-Marketplace
Trust through contract enforcement	Distributed validation, including proof-of-work mechanisms or proof-of-stake mechanisms. The network enforces the contract between seller and buyer. The network validates reputation ratings, including reviews and feedback mechanisms.	Third parties (such as a bank, certifying authority, promissory note, transfer systems, or other forms of contractual mechanisms). Usually controlled by the firm. Potential for significant alteration.
Transaction time	Can be instantaneous due to fast network validation. Delays can be mitigated using proof-of-stake/proof-by-consensus algorithms. ^{3,19}	Promissory note, letter of credit, or acceptance of credits that can take a long time.
Value	The network can reward participants with tokens or by accepting third-party tokens.	Banking systems (such as national exchanges, currency, and underwriters).
Privacy and security	Identity is not disclosed on the network. Tracking transactions can be facilitated, though with difficulty. Transaction details can be hidden behind layers of encryption. Cost of tampering with the network’s validation mechanism is high. ^a	Identity fully disclosed in the marketplace. As secure as the network’s components.


^a To break the network’s validation, an attacker would have to be able to control >50% of the network’s hash power, involving a huge economic cost in case of proof-of-work validation mechanisms. For proof-of-stake or proof-by-consensus mechanisms, tampering with the network’s validation represents an economic disincentive.

Table 2. Decentralization in different e-marketplaces.

E-Marketplace	Decentralization Possibility	Reasons	Cryptocurrency Support
Physical products	Partly decentralized	Many components to decentralize, including B2B support, accounting, payment gateway, and reputation	Bitcoin, Dash, Ethereum, Monero
Digital products (such as e-books, music, video, and domains)	Very likely	Fully online payment and delivery of goods	Bitcoin, Dash, Ethereum
User-generated content marketplace	Very likely	Online content, including blogs, reviews, and online reputation	Bitcoin, Ethereum, Steem
Prediction markets	Very likely	Blockchain-based validation to enforce contracts	Augur, Bitcoin, Ethereum, Truthcoin
Crowdfunding, sharing marketplaces	Very likely	Simpler validation; functionality supported by blockchain	Bitcoin, Dash, Ethereum, Zoon
Currency exchanges, remittances, complex financial contracts	Very likely	Easy-to-create complex contracts and low transaction costs	Bitcoin, Dash, Ethereum, Ripple


in which content creators are identified, recognized, and rewarded.¹¹ Blockchain-based prediction markets include applications in basic sciences, where, say, drug discovery or patient outcomes might be predicted accurately through the wisdom of the crowd.¹⁵ Likewise, blockchain-based software is being deployed in the financial industry for implementing automated reasoning for executing the complex rules in financial contracts.¹⁰ Given the blockchain's special characteristics, successful execution and validation of rules by a network of nodes extends its blockchain-based applications to artificial intelligence applications in complex rule-based systems.²³ In financial markets, for example, the blockchain enables international remittance opportunities with instantaneous currency transfers and “de-risks” international currency exchange; for example, BITT is a Bitcoin-based platform for inter-bank money transfer among 16 Caribbean nations, some running the risk of currency de-recognition. Similarly, the blockchain plays an important role mitigating problems in business environments where market friction due to weak legal institutions is a challenge, as in the real estate market where property records on the blockchain are being sought as a solution.⁸

Conclusion


By facilitating key marketplace functions, decentralization will, if successful, complement and rival traditional conventional e-marketplaces. 

References

1. Bakos, Y. The emerging role of electronic marketplaces on the Internet. *Commun. ACM* 41, 8 (Aug. 1998), 35–42.
2. Buterin, V. What proof of stake is and why it matters. *Bitcoin Magazine* (Aug. 26, 2013); <https://bitcoinmagazine.com/articles/what-proof-of-stake-is-and-why-it-matters-1377531463/>
3. Cusumano, M.A. The Bitcoin ecosystem. *Commun. ACM* 57, 10 (Oct. 2014), 22–24.
4. Duffield, E. and Diaz, D. Dash: A privacy-centric cryptocurrency. White Paper. Dash, Sept. 2014; <https://www.dash.org/wp-content/uploads/2015/04/Dash-WhitepaperV1.pdf>
5. The Economist. Workers on tap: The on-demand economy. *The Economist* (Dec. 2014); <https://www.economist.com/news/leaders/21637393-rise-demand-economy-poses-difficult-questions-workers-companies-and>
6. Eisenmann, T., Parker, G., and Van Alstyne, M.W. Strategies for two-sided markets. *Harvard Business Review* 84, 10 (Oct. 2006), 92.
7. Hart, O. and Holmström, B. *The Theory of Contracts*. Department of Economics, Massachusetts Institute of Technology, Cambridge, MA, 1986; <https://dspace.mit.edu/bitstream/handle/1721.1/64265/theoryofcontract00hart.pdf?sequence=1>



Decentralization is used to create a social network (such as Steem) in which content creators are identified, recognized, and rewarded.



8. Hodson, H. Bitcoin moves beyond money. *New Scientist* 220, 2945 (Nov. 20, 2013), 24.
9. Hoffman, D.L., Novak, T.P., and Peralta, M. Building consumer trust online. *Commun. ACM* 42, 4 (Apr. 1999), 80–85.
10. Hull, R., Batra, V.S., Chen, Y.-M., Deutsch, A., Heath III, F.F.T., and Vianu, V. Towards a shared ledger business collaboration language based on data-aware processes. In *Proceedings of the 14th International Conference on Service-Oriented Computing* (Banff, Alberta, Canada, Oct. 10–13), Springer International Publishing, 2016, 18–36.
11. Larimer, D., Scott, N., Zavgorodnev, V., Johnson, B., Calfee, J., and Vandenberg, M. *Steem: An Incentivized, Blockchain-Based Social Media Platform*. White Paper. Steem, New York, Mar. 2016; <https://steem.io/SteemWhitePaper.pdf>
12. ter Maat, M. The economics of e-cash. *IEEE Spectrum* 34, 2 (Feb. 1997), 68–73.
13. Migliardi, M., Merlo, A., and Passaglia, A. On the feasibility of moderating a peer-to-peer CDN system: A proof-of-concept implementation. In *Proceedings of the 10th International Conference on P2P, Parallel, Grid, Cloud, and Internet Computing* (Krakow, Poland, Nov 4–6). IEEE Press, 2015, 689–694.
14. Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008; <https://bitcoin.org/bitcoin.pdf>
15. Peterson, J. and Krug, J. Augur: A decentralized, open-source platform for prediction markets. *arXiv* (Jan. 5, 2015); <https://arxiv.org/abs/1501.01042>
16. Pick, F. and Dreher, J. Sustaining hierarchy—Uber isn't sharing. *Kings Review* (May 12, 2015); <http://magazine.ouishare.net/2015/05/sustaining-hierarchy-uber-isnt-sharing/>
17. Ratnasingham, P. Trust in web-based electronic commerce security. *Information Management & Computer Security* 6, 4 (1998), 162–166.
18. Reid, F. and Harrigan, M. An analysis of anonymity in the bitcoin system. In *Proceedings of the Third IEEE International Conference on Social Computing* (Boston, MA, Oct. 9–11). IEEE Press, 2011, 1318–1326.
19. Rochet, J.C. and Tirole, J. Two-sided markets: A progress report. *The RAND Journal of Economics* 37, 3 (2006), 645–667.
20. Rosov, S. Beyond Bitcoin. *CFA Institute Magazine* 26, 1 (Jan./Feb. 2015), 37–37.
21. Schwartz, D., Youngs, N., and Britto, A. *The Ripple Protocol Consensus Algorithm*. White Paper. Ripple Labs Inc., San Francisco, CA, 2014; <http://www.the-blockchain.com/docs/Ripple%20Consensus%20Whitepaper.pdf>
22. Subramanian, H. and Overby, E. Electronic commerce, spatial arbitrage, and market efficiency. *Information Systems Research* 28, 1 (Mar. 2017), 97–116.
23. Swan, M. Blockchain thinking: The brain as a decentralized autonomous corporation [commentary]. *IEEE Technology and Society Magazine* 34, 4 (Dec. 2015), 41–52.
24. Weyl, E.G. A price theory of multi-sided platforms. *The American Economic Review* 100, 4 (Jan. 2009), 1642–1672.
25. Wood, G. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Yellow Paper. Ethereum Project; <http://gavwood.com/paper.pdf>

Hemang Subramanian (hsubrama@fiu.edu) is an assistant professor in the Department of Information Systems and Business Analytics in the College of Business at Florida International University, Miami, FL.

© 2018 ACM 0001-0782/18/1 \$15.00



Watch the author discuss his work in this exclusive *Communications* video. <https://cacm.acm.org/videos/decentralized-blockchain-based-e-marketplaces>



DEBS2018

25–29 June 2018, Hamilton, New Zealand

Bay of Islands, NZ ©Alistair Guthrie

12th ACM International Conference on Distributed and Event-Based Systems

DEBS2018 will be held at the University of Waikato in Hamilton, New Zealand.

The ACM International Conference on Distributed and Event-based Systems (DEBS) is a premier venue for contributions in the fields of distributed and event-based systems. The objectives of the DEBS conference are to provide a forum dedicated to the dissemination of original research, the discussion of practical insights, and the reporting of experiences relevant to distributed systems and event-based computing. The DEBS conference aims at providing a forum for academia and industry to exchange ideas through industry papers and demo papers. The conference will also host a doctoral symposium, a workshop, tutorials and a grand challenge competition.

Grand Challenge

This year's Grand Challenge will use machine learning to make the naval transportation business more reliable. Explore gigabytes of real maritime spatio-temporal streaming data and compete with peers from academia and industry for the Grand Challenge prize. **Challenge start:** 15th of January 2018 **Submission deadline:** 15th of April 2018. The winner of the grand challenge will be awarded a \$1000 cash prize also!

For more information about the Grand Challenge visit:

<http://debs.org/2018/calls/gc.html>

Important Dates

Abstract submission for research track	Feb 21st, 2018
Research and industry paper submission	Feb 26th, 2018
Tutorial proposal submission	Mar 5th, 2018
Grand challenge solution submission	Apr 15th, 2018
Research and industry paper notification	Apr 17th, 2018
Poster, demo, doctoral workshop submission	Apr 29th, 2018
Conference	Jun 25th–29th, 2018

www.debs.org/2018

www.facebook.com/debs2018

We look forward to seeing you in New Zealand in 2018.

General Co-Chairs:

Annika Hinze, University of Waikato

David Eysers, University of Otago

Program Committee Co-Chairs:

Martin Hirzel, IBM T.J. Watson

Matthias Weidlich, Humboldt-Universität

Grand Challenge Co-Chairs:

Holger Ziekow, Furtwangen University

Zbigniew Jerzak, SAP

Martin Strohbach, AGT International

Pavel Smirnov, AGT International

Dimitris Zissis, MarineTraffic

Vincenzo Gulisano, Chalmers Uni of Technology

Workshops Chair:

Jat Singh, University of Cambridge

Tutorials Chair:

Andy Gokhale, Vanderbilt University

Publicity Chair:

Ruben Mayer, Universität Stuttgart

Design Chair:

Nicholas Vanderschantz, University of Waikato

The practice of hiding ill-gotten data in digital objects is rising among cyber thieves. New initiatives serve to educate, train, and thwart these activities.

BY WOJCIECH MAZURCZYK AND STEFFEN WENDZEL

Information Hiding: Challenges for Forensic Experts

INFORMATION HIDING IS a research domain that covers a wide spectrum of methods that are used to make (secret) data difficult to notice. Due to improvements in network defenses such techniques are recently gaining an increasing attention from actors like cybercriminals, terrorist and state-sponsored groups as they allow to store data or to cloak communication in a way that is not easily discoverable.²² There are several real-world cases that reached the attention of the public media, including the following:^{23,38}

▶ the arrest of one of al Qaeda's members in Berlin with video files containing hidden information on ongoing and future terrorists' operations (2012),^a

▶ the exfiltration of confidential data from the U.S. to Moscow by Russian spies (2010),^b

▶ the transfer of child pornographic material by a group of pedophiles called "Shadowz Brotherhood" (2002),^c and

▶ the planning of a terrorist attack after the September 11, 2001 attacks. A number of articles suggested that al Qaeda members used steganography to coordinate their actions (2001).^d

In these cases, information-hiding techniques were used to hide the confidential or illegal data into innocent-looking material, for example, digital pictures.

Steganography is a well-known subfield of information hiding that aims to cloak secret data in a suitable carrier. Since the time of Ancient Greece, over the Medieval Ages, to today's world, information hiding has been often used to conceal messages on their way to a desired recipient.³⁸ For instance, music notes were utilized to embed secret information that was only recognizable by the person that knows where to look for it. Another example for steganography is the writing with invisible ink.²⁶ The use of covert techniques grew significantly during the two World Wars, in which the mili-

a <http://edition.cnn.com/2012/04/30/world/al-qaeda-documents-future/>

b <http://www.bbc.com/news/10442869>

c <http://news.bbc.co.uk/2/hi/science/nature/2082657.stm>

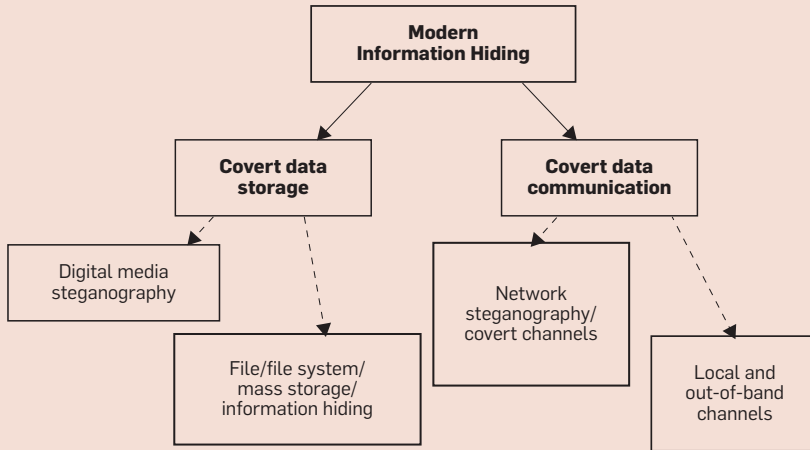
d <http://usatoday30.usatoday.com/tech/news/2001-02-05-binladen-side.htm>

» key insights

- This article provides an introduction to the current state of information hiding.
- Steganography-utilization in cybercrime is explored.
- The objectives of the Criminal Use of Information Hiding (CUIing) initiative are examined.



Figure 1. Modern information hiding classification based on the techniques' application and exemplary groups of methods.



Comparison of two types of carriers for data hiding.

Feature/Type of the carrier	Digital media	Network traffic
Method's capacity/bandwidth	Limited by the type of the digital media and the size of a file	Limited by the type of the traffic and the length of a transmission
Hidden data embedding	Cannot exceed file capacity	Can be slow but continuous over longer period of time
Data hiding application	Covert storage	Covert communication
Nature	Permanent	Ephemeral
Clues for forensic analysis	Can be available for forensic experts after transmission	Often not available when transmission ends
Method's detectability	Easy only if an original file is available	Hard due to different forms of acceptable traffic and varying network conditions
Cost of applying data hiding	Decrease in digital media quality	Increased delays, raised packet loss level, reduced feature set of protocols and/or affected user transmission quality
Robustness (secret data resistance to modifications)	Typically cannot survive conversion to another format	Typically vulnerable to dynamically changing network conditions

tary developed several methods to hide information in innocent-looking objects. So-called microdots, for example, hide text by shrinking it to the size of a punctuation mark that can be hidden on a sheet of paper.

Today's form of information hiding follows the same origins as the digital era. Modern information-hiding techniques can be divided based on their application into two broad groups: covert data storage and covert data communication (Figure 1). Covert data storage allows the application of data-hiding techniques to conceal secret in-

formation in such a way that no one besides the involved persons will know where the information is stored or how to extract it. Digital media steganography and file/file system/mass storage steganography are the most prominent classes belonging to this group. On the other hand, covert data communication methods focus on hiding the fact that any communication process took place and were initially described as channels that were not foreseen for communication.¹⁸ This means that involved parties can participate in a covert communication

and, in principle, a third-party observer would be unaware of it. The most important classes belonging to this group include out-of-band covert channels, network steganography (also known as network covert channels), as well as local covert channels (that are limited in communication range to the single device).

Here, we will briefly describe an evolution of the classes of techniques mentioned.

Modern Information Hiding: An Evolution of Techniques

It must be also noted that in the years between the early 1990s and about 2001, mostly academics considered information hiding as a relevant research discipline. This research domain was, however, shifted back into the focus of applied researchers, security professionals and law enforcement agencies (LEAs) after new cases became known in which information hiding was successfully applied for malicious purposes. Obviously, the biggest concern in LEAs is that covert techniques are being used to ensure stealth communication among terrorist/criminals and cybercriminals.

Below we briefly summarize the evolution of the data-hiding groups presented in Figure 1.


Digital media steganography incorporates techniques to hide information within digital images, audio files, and digital videos.⁸ Johnson and Katzenbeisser group such steganographic methods into six categories:¹⁵ steganographic methods can either substitute redundant data in cover objects, embed data in a signal's transform space, utilize spread spectrum techniques, change statistical properties of a cover object, or represent secret information by introducing a distortion into a signal. A sixth category differs from the first five and contains steganographic methods that create cover objects only for carrying secret information (instead of modifying existing cover objects). For a survey of this type of methods please refer to Cheddad.⁶

Another branch of modern information hiding is related to steganographic file systems. The first such approach was proposed by Anderson et al.¹ The main idea relied on the fact that encrypted data resembles random


bits naturally present on the disk. Therefore, only the ability to extract the vectors marking the file boundaries permitted the location process. Later, another approach for a hidden file system was proposed in Pang.²⁵ Authors implemented a Linux-based steganographic file system that could preserve the integrity of the stored files and employ a hiding scheme in the disk space by camouflaging with the aid of dummy hidden files and abandoned blocks. Recently, the concept of a steganographic system has been proposed in Neuner,²⁴ which utilizes system timestamps as a covert data storage. An alternative approach of secret data storage is to utilize chosen locations (such as Master File Table entries) or unorganized storage space on mass storage media.³⁰

Single-host (local) covert channels have been studied for several decades.¹⁸ These covert channels allow the leakage of information between isolated processes by utilizing shared resources, for example, local files or disk arms.¹⁰ Their major goal is to break a mandatory security policy. Also, the applications and operating systems were analyzed to detect and analyze such possible covert channels and to prevent and limit them, for example, within the VAX security kernel.¹⁴ Recently, the most common case in which local covert channels were investigated involves the colluding applications scenario²⁹ (proposed first in 2011). This scheme assumes the device is infected with a malware composed of two processes running in the separate sandboxes so they are unable to directly communicate. They use data-hiding techniques to establish a local covert channel to bypass the security framework of the infected device. Typically malicious processes are modulating common resources like shared notifications and file-system locks, or they alter the idle state of the CPU or the system load (for more information see methods surveyed in Mazurczyk²¹).

During last few years there is also an increased interest in so-called out-of-band covert channels quite similar to single-host covert channels but utilize a shared physical medium that can be accessed by both the sending and receiving process.⁴ Out-of-band channels do



The biggest concern in law enforcement agencies is that covert techniques are being used to ensure stealth communication among terrorist/ criminals and cybercriminals.



not necessarily break a mandatory security policy but rather allow the stealthy transfer of secret information. By using physical mediums, out-of-band covert channels overcome air-gaps between systems, for example, using non-audible acoustic channels,¹¹ light, or vibration.¹² Several other out-of-band covert channels are described in Carrara.⁴

It must be noted that recently secret-concealing techniques have spanned considerably to enable data hiding in many other areas, especially in network traffic: the so-called *network information hiding* (and in particular its sub-field called *network steganography*) constitutes a growing branch of modern information hiding.

Network steganography deals with the concealment of information within network transmissions.²³ This means that network data that appears to be innocent is actually carrying hidden data. Network information hiding can be used, for example, by malware to conceal its command and control communication (instead of only encrypting it) while it is also suitable for a long-term stealthy data leakage, for example, after an organization was attacked using an advanced persistent threat.²² Indeed, since the Linux Fokirtor malware^e (2013), which hides data in SSH traffic, and Regin^f (2014), which utilizes several network protocols to signal hidden information through these, more malware with more sophisticated features arose. Malware also increasingly exploits functions of online social networks to transfer hidden information, for example, embedded into exchanged messages or posted images. Fisk et al. reported in 2002 that a typical Web server could leak up to several gigabytes of data per year via network information-hiding methods.⁷ Now, 15 years later, Internet-based information leakage is considered to be much more powerful, resulting in exponentially increasing amounts of leaked data per time.

In comparison to digital media steganography, network steganography can be used for a constant data leakage.²³ Therefore, network infor-

e http://www.theregister.co.uk/2013/11/15/stealthy_linux_backdoor/

f <http://www.symantec.com/connect/blogs/regin-top-tier-espionage-tool-enables-stealthy-surveillance>

mation hiding methods modify either the timing or the content of network traffic, for example, by modifying unused bits, the structure of protocol headers, the rate in which traffic is sent or the order of packets.³⁶ First methods were introduced already in the 1980s and 1990s,^{9,28} but most hiding methods for networks were published after 2000 and were focused on newer protocols such as IPv6²⁰ and LTE advanced,²⁷ cyber-physical systems (such as smart homes/buildings),³² industrial communication protocols such as Modbus,¹⁹ and cloud computing.⁵ Extensive surveys discussing more than hundred methods can be found in Zander,³⁶ Wendzel,³⁵ and Mazurczyk.²³ While a digital image allows to carry a rather limited number of bytes per single file, net-

work traffic can permanently carry little amounts of data, at day and at night. A comparison of the two digital carrier types, such as digital media (the most well-researched and most popular cover for information hiding methods) and network traffic is shown in the accompanying table.

From this perspective, it must be emphasized that information-hiding technique utilization on a suspect's computer will not be discovered by a forensic analysis if it is not being directly sought. In general, all that criminals or terrorists need to enable a covert communication is to agree upon the carrier in which the secret data will be embedded. The carrier can be a digital image, audio, video, text file, network traffic, or any other digital medium. Obviously, this also

involves specialized software exploiting this carrier.

Moreover, a covert sender and receiver will often utilize an encryption scheme (and a password/key) that will allow for securing the content even if the hidden communication is discovered. It must be also noted that in practice there is a plethora of various types of information-hiding techniques in which carriers can be modified and a great number of carriers that can be used for this purpose, which adds another dimension to the challenges for the forensic experts.

Current State of Information Hiding in Cybercrime and Forensics Challenges

In general, it is impossible to precisely evaluate how widespread the use of information-hiding techniques is among criminals/terrorists, cybercriminals, or state-sponsored groups. However, there are signs that information-hiding utilization can be heavily underestimated as security experts do not always correctly recognize and classify techniques used. For instance, by observing how malware developers increasingly apply information-hiding techniques we can be certain this trend is most likely going to increase. Figure 2 presents data that can support this claim. It illustrates the percentage of information hiding-capable malware identified (with respect to the total number of discovered malware) between 2011 and 2016 (historical data collected by members of the Criminal Use of Information Hiding—CUIng—initiative, to be discussed later). We treated malicious software as an information hiding-capable malware when it has been used at least once for data-hiding techniques defined in Figure 1. To collect this information, we relied on reports from security companies, our own continuous malware landscape analysis and data from LEAs.

Nevertheless, the trend observed in Figure 1 may still be only a “tip of an iceberg.” As a result, discovery of data-hiding tools will become a great challenge for LEAs, counterterrorism organizations, and forensic experts.

It is also worth noting that information-hiding techniques were initially only found in highly sophisticated mal-

Figure 2. Percentage of the identified information hiding-capable malware between 2011 and 2016 (historical data collected by members of the CUIng initiative).

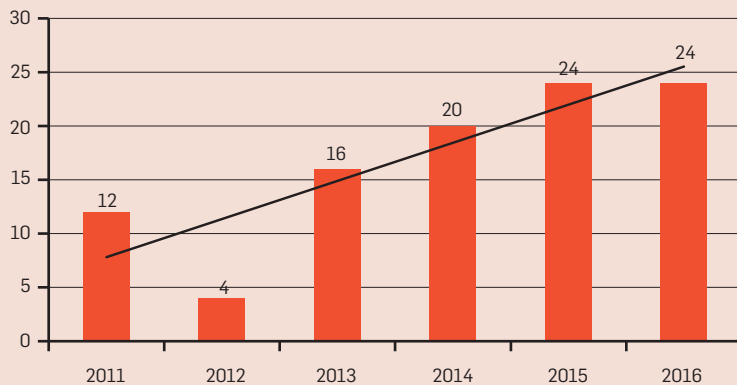
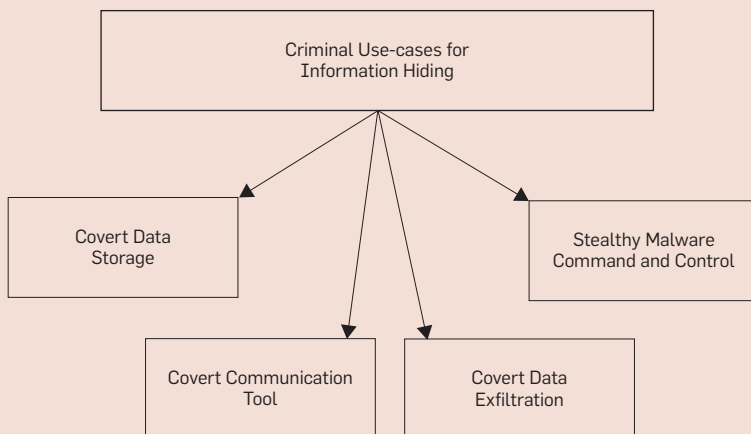


Figure 3. Use-cases for the criminal application of information hiding methods.



ware like Regin, Duqu,^g or Hammertoss,^h which by security experts are thought to be created by nation-states to infiltrate a wide range of international targets and eventually launch attacks if necessary.

However, currently it can be observed that not only APTs, but even the typical malware is turning toward increased utilization of data hiding. This is somehow expected as typically sophisticated malicious software is supported by an actor that is not strongly resource-constrained (in money, human resources or in time). Therefore, takeover of advanced information hiding techniques by cybercriminals can be the result of a “trickle down” effect from “milware” (that is, state-sponsored malicious software) to “malware” (created by non-state groups) as described in Zielińska.³⁸ It is also worth noting that, typically, cybercriminals will mostly focus on hiding as much information as necessary, whereas nation-state actors will try to conceal as much data as possible.

Based on the aim to be achieved, information-hiding techniques can be used by criminals/terrorists and other malicious actors for the following purposes (Figure 3):

- ▶ As a mean for covert storage: To hide secret data in such a way that no one besides the owner is authorized to discover its location and retrieve it. In other words, the aim is to not reveal the stored secret to any unauthorized party. This way criminals/terrorists can store their secret data in a hidden manner (such in the case of pedophiles group “Shadowz Brotherhood” mentioned earlier).

- ▶ As a covert communication tool: To communicate messages with the aim of keeping some aspect of their exchange secret. Criminals/terrorists can use information hiding to covertly exchange their confidential data (for example, as in case of the Russian spy ring discovered in U.S.).

- ▶ As a data exfiltration technique: Cybercriminals/insiders can use it to steal/exfiltrate confidential data (this is the case for a Zeus/Zbot trojanⁱ).

g <http://resources.infosecinstitute.com/duqu-2-0-the-most-sophisticated-malware-ever-seen/#gref>

h <https://www2.fireeye.com/rs/848-DID-242/images/rpt-apt29-hammertoss.pdf>

i <https://blog.malwarebytes.com/threat-analysis/2014/02/hiding-in-plain-sight-a-story-about-a-sneaky-banking-trojan/>

Challenges for Forensic Examiners

- ▶ Due to the large number of information hiding techniques and a diversity of potential carriers, *no general and effective off-the-shelf detection solutions exist*. As covert data exchange is typically tightly coupled with the adopted carrier it makes detection poorly generalizable and a challenging task. Some tools exist to help forensic experts (the majority is for digital media steganography) but they are typically crafted for a particular technique or a small group of them (for example, one type of carrier).
- ▶ Typically, investigators do not focus on recovering hidden data but on *discovering whether any known information hiding program has been installed and used* which does not lead toward revealing what message has been hidden especially if a proprietary tool is utilized.
- ▶ There are *no guidelines provided for investigators that will allow for the systematic search for a hidden content*.
- ▶ Some carriers are of a more ephemeral nature, for example, network traffic. *If the traffic it is not captured while being sent, then it is very unlikely that it can be recovered later*. This makes network information hiding and forms of out-of-band covert channels even a greater challenge for forensics examiners than typical “classical” data hiding methods. Moreover, there are *practically no tools for detection of information hiding in especially network traffic and in the physical medium*.
- ▶ If the inspected computer is infected with information hiding-capable malware, then for an investigator it could be difficult to discover such a fact.
- ▶ Forensics examiners often rely solely on hash sets of the known data-hiding tools or the tools they know, thus they *may not recognize a steganographic tool even if they found it, for example, proprietary*.
- ▶ *The existing software for information hiding is becoming increasingly difficult to discover*; it can be placed on a removable media and executed directly, without additional installation. In this case, no remnants of the program would be found on the suspect’s hard drive. However, it turns out not always to be true as recently a tool called RSAS (Removed Steganography Application Scanner) has been introduced that allows to discover artifacts of the known steganographic programs even if they were previously uninstalled or run from a portable memory storage.
- ▶ With the ever-increasing amount of network traffic, hard drive storage capacity and the number of diverse files (images, videos, audio, text files) that a typical user stores, *a complete search for carries with embedded secret data becomes a tremendously time-consuming task*.
- ▶ Given the fact that also passive signaling of hidden information can be applied, *forensics cannot solely focus on active signaling methods during their analysis*. For instance, research demonstrates that the timing behavior of server-side applications can be enough to deduce information about the existence of users on a system. This can be done by measuring and comparing the response time for requests with different user-names of a web application.^{3,29} A recent vulnerability in the popular OpenSSH daemon falls into this category.^a

a <http://seclists.org/fulldisclosure/2016/Jul/51>

- ▶ As a mean for covert malware communication: Finally, malware can be equipped with information hiding techniques to become stealthier while residing on the infected host and/or while communicating with Command & Control (C&C) servers (for example, a Hammertoss APT).

From the forensic challenges perspective first it must be noted that there is a huge asymmetry when it comes to devising new information hiding techniques and its detection/elimination. Although research on countermeasures started early (Zander³⁷), their application in practice can be challeng-

ing or impossible (for example, because they were designed for the design phase of a system, not for a forensic analysis). Developing new data-hiding methods is usually much easier than the effort needed to detect them.

Additionally, if the carrier is selected properly (that is, if the carrier is popular enough so it is not an anomaly itself) even trivial techniques can remain hidden for long periods of time. What is worse from a cybercrime perspective: there are many information-hiding tools that are easy to access and use, even for an unexperienced user. In April 2014, the

Steganography Analysis and Research Center (SARC) claimed their latest version of the Steganography Application Fingerprint Database contained over 1,250 steganography applications.

It must be also noted that many of the commercial tools for information-hiding detection do not exactly focus on revealing the embedded secret data but rather try to find artifacts left behind by the hiding tools. This appears to be a good approach; however, it is only successful for the list of well-known data hiding tools or under assumption that it was the legitimate user of the device who installed this type of software. In practice, if a proprietary data-hiding tool is utilized or the device is infected with information hiding-capable malware, revealing its artifacts will be not possible or the true intention of the attacker will be still difficult to establish.

In contrast, the detection of hidden data, which is typically done by the forensics examiner for LEAs or anti-terrorism units, is far more challenging and the extraction/recovery of the secrets is even harder (for example, due to utilized encryption of covert data).


Another point is that still many forensics examiners do not routinely check a suspect's computer for information-hiding software and, even if they do, several issues arise (see the accompanying sidebar).

When discussing forensic challenges for information hiding, the two most important aspects that should be considered are *technical capabilities of the suspect and type of the crime*.¹⁷


The technical capabilities of the suspect may map to the resources that he has on his computer (installed software, hardware) or which he accessed (for example, visited webpages or downloaded e-books).

The type of a suspected crime can also point to a utilization of data hiding. For example, terrorists or child pornographers tend to hide their secrets in images and then send it through email or by posting it on a website. A similar case is with crimes that involve the transfer of business-type records. Obviously, if a cybercrime is investigated then information hiding usage is always a viable option.

As mentioned, methods as well as applications of information hiding



Many of the commercial tools for information-hiding detection do not exactly focus on revealing the embedded secret data but rather try to find the artifacts left behind by the hiding tools.



have become significantly more sophisticated in recent years. For example, an arising challenge for forensic experts are solutions like SonicVortex Transactions (<http://www.sonic-coin.com/>) that can be treated as a next-generation crypto currency platform. It enables hiding encrypted bitcoin transactions in innocent pictures and offers a stealthy address and built-in TOR support. This can potentially provide tremendous difficulties for investigating financial crimes.

In addition, there have been press releases stating that criminals/terrorists are exploiting different aspects of online games/gaming consoles to enable covert communication¹ as they offer many aspects including digital images, video, network traffic, and even elements within the virtual world that can be modified in order to conceal messages. It must be noted this option was recognized in the academia community almost a decade ago.³⁰

A few academic publications already deal with the transfer and storage of hidden data in the Internet of Things (IoT).^{23,32} It is likely it will only be a matter of time before IoT services, such as smart homes and wearables, will be subjected to information hiding-based cybercrime. IoT devices provide entirely new ways to store hidden data in their actuators and in the memory of embedded components—places where no current methods will search for hidden data and for which no tailored tools are available.

Also, other popular and innocent-looking online services like Skype, IP telephony, BitTorrent, or cloud storage systems can be exploited to enable covert communication.²³ Therefore, network traffic and data exchanged during such transmissions can be utilized for information hiding purposes often without overt sender and overt receiver knowledge or consent.

Moreover, hiding tools became increasingly adaptive. In this context, adaptiveness refers to a malware function that can automatically adjust to a changing environment. For instance, imagine an administrator

j <http://cjel.law.columbia.edu/preliminary-reference/2016/communicating-terror-the-role-of-gaming-consoles-and-backdoors/>

discovered a malware using network information hiding and he decides to block its communication channel. In such a case, the administrator could introduce new filter rules for firewalls or traffic normalizers. However, adaptive malware would detect the blocked channel and would most likely find a way to route around this barrier. Therefore, it uses one of the several different hiding methods available, eventually building a covert overlay network with dynamic routing capabilities. These techniques were already discovered several years ago in academia.^{2,33}

A New Initiative to Fight Information Hiding-Based Cybercrime

In the context of the forensic challenges mentioned previously, policymakers, governmental organizations, and law-enforcement, security industry and academia should work jointly to build novel products and methods to protect companies and citizens.

Criminal Use of Information Hiding (CUIng) is an initiative recently launched in cooperation with Europol's European Cybercrime Centre (EC3). The initiative is open for all interested members from different backgrounds to participate in it. The current structure of the initiative consists of the Steering Committee and regular members. The Steering Committee is responsible for setting the strategic direction of the initiative and proposing, approving and coordinating all its activities. The Steering Committee is a mix of members from academia, industry, LEAs, and institutions.

Its main objectives, which are summarized in Figure 4, are the following:

- ▶ to raise awareness for the criminal use of information hiding on all relevant levels (from IT administration to governments),
- ▶ to track progress of academic research in the domain,
- ▶ to monitor the technology's utilization by criminals,
- ▶ to share information about incidents between the relevant players,
- ▶ to provide practical advice to these players,
- ▶ to work jointly with researchers around the world, and
- ▶ to foster the education and training

on the professional and academic level.

Although the CUIng initiative started only recently, its circle of involved organizations and individuals has managed to collect and categorize a vast amount of relevant information, be it about discovered malware pieces, information about incidents or research output. Currently it consists of more than 100 experts from over 30 countries worldwide who are presenting different backgrounds. The initiative gathers and shares the following information:

- ▶ General background on information hiding: it provides a general overview of recent trends and techniques.
- ▶ Scientific publications: relevant papers, which present the state-of-the-art in academic research in this area.
- ▶ Information hiding-capable malware reports: analyses of real-life malicious software that uses data hiding techniques.
- ▶ Relevant tools: applications which allow to conceal data as well as different approaches for countermeasure/detection.
- ▶ New categorization and didactic concepts and training course materials: new concepts on how to teach/train information hiding to make the topic more accessible and materials from the previous trainings.

First surveyed results and trends were presented at relevant events

(both academic and industry conferences). For instance, CUIng was a program partner for several industry eCRIME conferences and it will organize a dedicated event CUING 2017 workshop (with ARES 2017) and a special session (with IWDW 2017) on these topics.

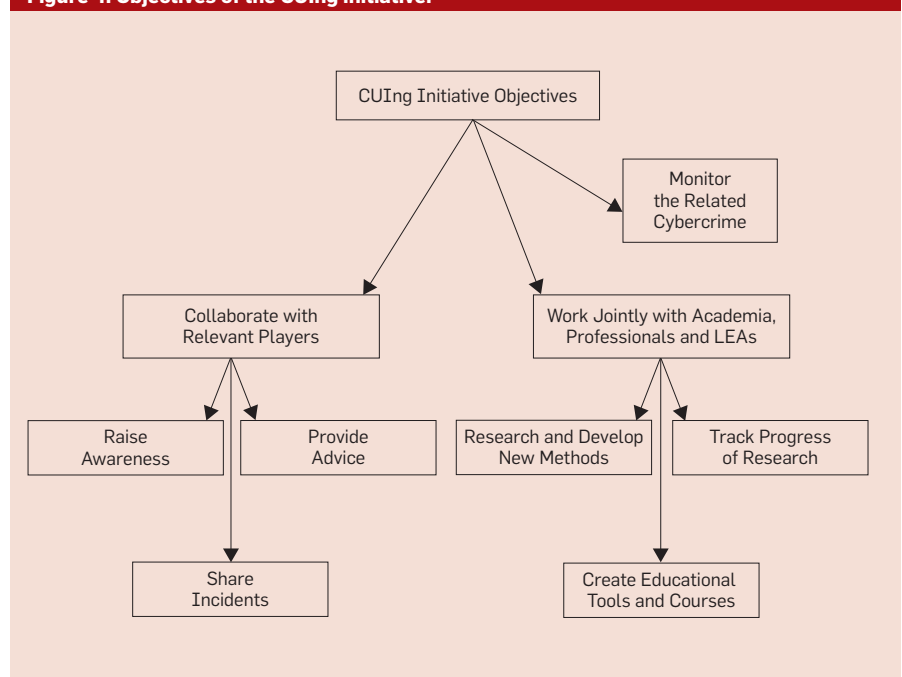
CUIng helped Europol's EC3 to create a CyberBit (intelligence notification on cyber-related topics that aim to raise awareness and to trigger discussions or further actions), that is, a brief backgrounder for the Trends Series entitled "Steganography for Increased Malware Stealth." In January 2016, a dedicated training course for EC3 entitled "Training on Information Hiding Techniques and its Utilization in Modern Malware" was organized.

CUIng members are also involved in creating new tools, projects, and concepts for digital forensic purposes. The most notable examples include:

- ▶ Network Information Hiding Patterns project^k that allows the reduction of a large number of available hiding techniques to only several patterns—this can aid the community to remain focused on core developments and to understand better the network hiding concepts.
- ▶ Covert Channel Educational Analysis Protocol (CCEAP) tool,³⁴

k <http://ih-patterns.blogspot.de/p/introduction.html>

Figure 4. Objectives of the CUIng initiative.



which defines a sample protocol to teach various network hiding patterns and can be used in didactic environments. The tool is unique as it lowers the barrier for understanding network covert channels by eliminating the requirement of understanding several network protocols in advance. However, it must be noted, that testbeds not based on hiding patterns³³ exist.


► Removed Steganography Application Scanner (RSAS) tool¹ that enables to discover artifacts of the known steganographic applications even if they were previously uninstalled or run from a portable memory storage.

Current CUIng members' experiences related to the initiative show that cooperating jointly and building a robust community will take advantage of the expert knowledge and expertise from academia, industry, law enforcement, and institutions. This networking approach does not eliminate but limits the problem of the criminal use of information hiding before it becomes a much more widespread phenomenon. It must be also noted that CUIng is about to release a first set of guidelines for the protection of organizations and the forensic analysis in the coming year.

Outlook

The increasing number of known cases in which modern information hiding is applied in cybercrime as well as the constantly rising number of academic publications in the field underpin the importance of the topic and the broad interest in it. It is important to foster professional and academic training on information hiding, a better understanding and the improvement of the methodology in the field, especially for forensics. Another need is to enable a better sharing of incidents and trends. The CUIng initiative presented here is a vehicle to push these processes.

More information about CUIng can be found at <http://www.cuing.org>.

Acknowledgments. The authors thank the anonymous reviewers for helpful and constructive comments that greatly contributed to improving this article. 

¹ <https://nicolatalin.github.io/rsas/>

References

- Anderson, R., Needham, R. and Shamir, A. The steganographic file system. *Information Hiding*. Springer, 1998, 73–82.
- Backs, P., Wendzel, S. and Keller, J. Dynamic routing in covert channel overlays based on control protocols. In *Proceedings of the ISTEP'12 Workshop* (2012). IEEE, 32–39.
- Bortz, A. and Boneh, D. Exposing private information by timing web applications. In *Proceedings of the WWW* (2007). ACM, 621–628.
- Carrara, B. and Adams, C. Out-of-band covert channels—A survey. *Computing Surveys* 49, 2 (2016). ACM, 23.
- Caviglione, L., Podolski, M., Mazurczyk, W. and Tanigro, M. Covert channels in personal cloud storage services: The case of Dropbox. *IEEE Trans. Industrial Informatics*, 2016.
- Cheddad, A., Condell, J., Curran, K. and Mc Keivitt, P. Digital image steganography: Survey and analysis of current methods. *Signal Processing* 90, 3 (Mar. 2010), 727–752.
- Fisk, G., Fisk, M., Papadopoulos, C. and Neil, J. Eliminating steganography in Internet traffic with active wardens. *Information Hiding*, LNCS 2578 (2002). Springer, 18–35.
- Fridrich, J. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2009.
- Girling, C.G. Covert Channels in LANs. *IEEE Trans. Softw. Engin.* 13, 2 (1987). IEEE, 292–296.
- Gold, B.D., Linde, R.R., Peeler, R.J., Schaefer, M., Scheid, J.F. and Ward, P.D. A security retrofit of VM/370. In *Proceedings of the AFIPS Conference* (1979). AFIPS Press, 335–344.
- Hanspach, M. and Goetz, M. On covert acoustical mesh networks in air. *J. Communications* 8, 11 (2013).
- Hasan, R., Saxena, N., Halevitz, T., Zawoad, S. and Rinehart, D. Sensing-enabled channels for hard-to-detect command and control of mobile devices. In *Proceedings of the Symp. Information, Computer and Communications Security*. ACM, New York, NY, 2013, 469–480.
- Herr, T. and Armbrust, E. Milware: Identification and implications of state authored malicious software. In *Proceedings of the 2015 New Security Paradigms Workshop*. ACM, New York, NY, 29–43.
- Hu, W.M. Reducing timing channels with fuzzy time. *J. Computer Security* 1, 3/4 (1992), 233–254.
- Johnson, N.F. and Katzenbeisser, S.C. A survey of steganographic techniques. *Information Hiding*. Artech House, 2000.
- Kemmerer, R.A. Shared resource matrix methodology: An approach to identifying storage and timing channels. *ACM Trans. Computer Systems* 1, 3 (Aug. 1983), 256–277.
- Kessler, G.C. An overview of steganography for the computer forensics examiner. *Forensic Science Communications* 6, 3 (Jan. 2004), 1–2.
- Lampson, B.W. A note on the confinement problem. *Commun. ACM* 16, 10 (Oct. 1973), 613–615.
- Lemay, A., Fernandez, J.M. and Knight, S. A Modbus command and control channel. In *Proceedings of the Annual IEEE Systems Conference*. IEEE, 2016.
- Lucena, N.B., Lewandowski, G. and Chapin, S.J. Covert channels in IPv6. In *Proceedings of Privacy Enhancing Technologies*, LNCS 3856 (2005). Springer, 147–166.
- Mazurczyk, W. and Caviglione, L. Steganography in modern smartphones and mitigation techniques. *IEEE Commun. Surveys & Tutorials* 17, 1 (2014), 334–357; DOI: 10.1109/COMST.2014.2350994
- Mazurczyk, W. and Caviglione, L. Information hiding as a challenge for malware detection. *IEEE Security and Privacy*, 2 (2015).
- Mazurczyk, W., Wendzel, S., Zander, S., Houmansadr, A. and Szczypiorski, K. *Information Hiding in Communication Networks*. Wiley-IEEE Press, 2016.
- Neuner, S., Voyiatzis, A.G., Schmiedecker, M., Brunthaler, S., Katzenbeisser, S. and Weippl, E.R. Time is on my side: Steganography in file system metadata. *Digital Investigation* 18 (2016), S76–S86.
- Pang, H., Tan, K. and Zhou X. Stegfs: A steganographic file system. In *Proceedings of the International Conf. on Data Engineering*, 2003, 657–667.
- Petitcolas, F., Anderson, R. and Kuhn, M. Information hiding—A survey. *IEEE* 87, 7 (1999), 1062–1078.
- Rezaei, F., Hempel, M., Peng, D., Qian, Y. and Sharif, H. Analysis and evaluation of covert channels over LTE advanced. In *Proceedings of the Wireless Communications and Networking Conference*. IEEE, 2013, 1903–1908.
- Rowland, C.H. Covert channels in the TCP/IP protocol suite. *First Monday* 2, 5 (1997).
- Schlegel, R., Zhang, K., Zhou, X., Intwala, X., Kapadia, A., Wang, X.: Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones, in: *Network and Distributed System Security Symposium*, 2011.
- Thompson, I. and Monroe, M. FragFS: An advanced data hiding technique. BlackHat Federal, 2006; <http://www.blackhat.com/presentations/bh-federal-06/BH-Fed-06-Thompson/BH-Fed-06-Thompson-up.pdf>
- Tseby, T., Iglesias Vázquez, F., Bernhardt, V., Frkat, D. and Annessi, R. A network steganography lab on detecting TCP/IP covert channels. *IEEE Trans. Education* 59, 3 (2016), 224–232.
- Wendzel, S., Kahler, B. and Rist, T. Covert channels and their prevention in building automation protocols—A prototype exemplified using BACnet. *GreenCom/ CPSCom*. IEEE, 2012, 731–736.
- Wendzel, S. and Keller, J. Hidden and Under Control—A survey and outlook on covert channel-internal control protocols. *Annals of Telecommunications* 69, 7 (2014). Springer, 417–430.
- Wendzel, S. and Mazurczyk, W., Poster: An educational network protocol for covert channel analysis using patterns. In *Proceedings of the ACM Conference on Computer and Communications Security* (Vienna, Austria, Oct. 24–28, 2016), 1739–1741.
- Wendzel, S., Zander, S., Fechner, B. and Herdin, C. Pattern-based survey and categorization of network covert channel techniques. *Computing Surveys* 47, 3 (2015). ACM, 50.
- Zander, S., Armitage, G. and Branch, P. A survey of covert channels and countermeasures on computer network protocols. *IEEE Communications Surveys & Tutorials* 9, 3 (2007). IEEE, 44–57.
- Zander, S., Armitage, G. and Branch, P. Covert channels in multiplayer first person shooter online games. In *Proceedings of the 33rd IEEE Conference on Local Computer Networks* (2008), 215–222.
- Zielińska, E., Mazurczyk, W. and Szczypiorski, K. Trends in steganography. *Comm. ACM* 57, 3 (Mar. 2014). ACM, 86–95.

Wojciech Mazurczyk (wmazurczyk@tele.pw.edu.pl) is an associate professor at Warsaw University of Technology, Warsaw, Poland.

Steffen Wendzel (wendzel@hs-worms.de) is an associate professor at Worms University of Applied Sciences, Worms, and a researcher at Fraunhofer FKIE, Bonn, Germany.

© 2018 ACM 0001-0782/18/1 \$15.00



Watch the authors discuss their work in this exclusive *Communications* video. <https://caacm.acm.org/videos/information-hiding>

research highlights

P. 96

Technical Perspective Moving Spectrum

By David C. Parkes

P. 97

Deep Optimization for Spectrum Repacking

By Neil Newman, Alexandre Fréchet, and Kevin Leyton-Brown

P. 105

Technical Perspective Can High Performance Be Portable?

By Manuel Chakravarty

P. 106

Halide: Decoupling Algorithms from Schedules for High-Performance Image Processing

By Jonathan Ragan-Kelley, Andrew Adams,
Dillon Sharlet, Connelly Barnes, Sylvain Paris, Marc Levoy,
Saman Amarasinghe, and Frédo Durand

Technical Perspective

Moving Spectrum

By David C. Parkes

HOW CAN YOU transform the use of radio spectrum from low-value use such as over-the-air television to high-value use such as wireless broadband? A “big bang” response was proposed in the early 2000s. The idea was to use a market in which owners of spectrum could sell their spectrum for new uses by others. But this was plagued with difficulty.

Imagine TV stations as owners of cars in a parking lot. Each wireless company wants to buy a large, contiguous space in the lot. Doing so will create great value. But to succeed, it must buy space from each of a number of stations. And each station will try to seek a huge amount of money for this space! The solution to this “holdout problem” came in a new regulation that gave the Federal Communications Commission (FCC) the right to move TV stations to new frequencies. This creates competition. As long as enough stations sell spaces *anywhere* in the lot then remaining cars can be moved around to create space.

The FCC’s “incentive auction” proceeds in stages, each stage associated with a *clearing target*. To illustrate, suppose the target is to sell half of the parking lot. Each stage proceeds in two interrelated parts. First, buyers compete to set prices for the large, contiguous spaces that would be created if half the lot were sold. Buyers drop out over time and the auction continues until every buyer who remains can be allocated space within the clearing target. Second, sellers place lower and lower bids in a “reverse auction” to sell their individual spaces. They drop out over time (refusing to sell their space) and the auction continues until every seller who remains is needed to be able to move around the sellers who refuse to sell in order to meet the clearing target. This repeats, each stage with a progressively smaller target, until the revenue generated is at least that of the cost of the reverse auction.


The incentive auction presents a critical algorithmic challenge—that of solving a *repacking problem* when operating the reverse auction, and doing so quickly and at scale. The real problem is more complicated than repacking a parking lot. It is as if all the parking slots were shaped like Tetris pieces, and furthermore my psychedelic-colored car cannot go right next to your white car (in reality, there are complex, physics-based interference constraints that dictate what is feasible). Consider a seller who is willing to sell at the current price. The repacking problem asks: *Is there a feasible assignment of the available spectrum (the spectrum not targeted to buyers) to this seller and all the other stations who have already chosen not to sell?* If the answer is NO (or not provably YES) then we must finalize the price to this seller. If the answer is YES then we can lower the price and give the station the choice of taking the next price or walking away.

This scenario provides the backdrop for the breathtaking research contribution presented in Newman et al. For the auction to run in a reasonable amount of time, the FCC wanted to run two rounds of price updates per day, giving a few hours to process a round. Bids had to be processed sequentially. Taking into account the fraction of instances that finish very quickly, this required solving each repacking instance within approximately one minute. State-of-the-art algorithms were able to solve only 80% of hard instances within this time cut-off. The solver in Newman et al. can solve 88% of hard instances in *under a second* and 96% within the cutoff. The authors estimate the impact of these improvements could represent a \$1 billion gain to the U.S. government in lower prices and thus reduced cost in the reverse auction.

How did they do this? The approach is that of *data-driven algorithm design*. This allows decisions about the precise design of an algo-

rithm, and in this case the design of a portfolio of different, complementary algorithms that run in parallel, to be made in a principled, data-driven way. The role of an algorithm designer becomes that of designing configurable algorithms as well as using creativity to come up with domain-specific heuristics that could prove useful. In addition, the designer provides a realistic distribution of inputs on which to optimize performance. Having exposed 191 parameters in the design space, each nested as many as four-levels deep, the authors call their approach “deep optimization.”

There are larger lessons here. First, through outstanding computer science in developing a customized solution to this difficult and important problem, the researchers provided confidence that these large-scale, NP-hard problems could be solved at scale. Second, this success speaks to the importance of sustained research—this is the culmination of a concerted research effort over more than a decade into data-driven algorithm design. This project itself is a tour de force in “big bang” computer science, with an open source simulator, the inclusion of 20 state-of-the-art SAT solvers, and days of compute time to generate their results.

Eventually, the FCC’s Incentive auction generated \$19.8 billion of revenue, \$10.1 billion of which flowed to broadcasters. It moved 84MHz of spectrum to highest and best use with all trades voluntary. This is one of the biggest successes to date for the Economics and Computer Science research agenda. 

David C. Parkes is the George F. Colony Professor of Computer Science, and Co-Director, Harvard Data Science Initiative in the Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA.

Deep Optimization for Spectrum Repacking

By Neil Newman, Alexandre Fréchet, and Kevin Leyton-Brown

Abstract

Over 13 months in 2016–17 the U.S. Federal Communications Commission conducted an “incentive auction” to repurpose radio spectrum from broadcast television to wireless internet. In the end, the auction yielded \$19.8 bn, \$10.05 bn of which was paid to 175 broadcasters for voluntarily relinquishing their licenses across 14 Ultra High Frequency (UHF) channels. Stations that continued broadcasting were assigned potentially new channels to fit as densely as possible into the channels that remained. The government netted more than \$7 bn (used to pay down the national debt) after covering costs (including retuning). A crucial element of the auction design was the construction of a solver, dubbed SAT-based Feasibility Checker (SATFC), that determined whether sets of stations could be “repacked” in this way; it needed to run every time a station was given a price quote. This paper describes the process by which we built SATFC. We adopted an approach we dub “deep optimization,” taking a data-driven, highly parametric, and computationally intensive approach to solver design. More specifically, to build SATFC we designed software that could pair both complete and local-search SAT-encoded feasibility checking with a wide range of domain-specific techniques, such as constraint graph decomposition and novel caching mechanisms that allow for reuse of partial solutions from related, solved problems. We then used automatic algorithm configuration techniques to construct a portfolio of 8 complementary algorithms to be run in parallel, aiming to achieve good performance on instances that arose in proprietary auction simulations. To evaluate the impact of our solver in this paper, we built an open-source reverse auction simulator. We found that within the short time budget required in practice, SATFC solved more than 95% of the problems it encountered. Furthermore, the incentive auction paired with SATFC produced nearly optimal allocations in a restricted setting and substantially outperformed other alternatives at national scale.

1. INTRODUCTION

Many devices, including broadcast television receivers and cellphones, rely on the transmission of electromagnetic signals. These signals can interfere with each other, so transmission is regulated: For example, in the U.S., by the Federal Communications Commission (FCC). Since electromagnetic spectrum suitable for wireless transmission is a scarce resource and since it is difficult for a central authority to assess the relative merits of competing claims on it, since 1994 the FCC has used *spectrum auctions* to allocate broadcast rights (see, e.g., Milgrom²⁷). Many regulators around the world have followed suit. At this point, in the US (as in many other countries), most useful radio spectrum has been

allocated. Interest has thus grown in the *reallocation* of radio spectrum from less to more valuable uses. Spectrum currently allocated to broadcast television has received particular attention, for two reasons. First, over-the-air television has been losing popularity with the rise of cable, satellite, and streaming services. Second, the upper UHF frequencies used by TV broadcasters are particularly well suited to wireless data transmission on mobile devices—for which demand is growing rapidly—as they can penetrate walls and travel long distances.²³

It thus made sense for at least some broadcasters to sell their licenses to wireless internet providers willing to pay for them. Ideally, these trades would have occurred bilaterally and without government involvement, as occurs in many other markets. However, two key obstacles made such trade unlikely to produce useful, large-scale spectrum reallocation, both stemming from the fact that wireless internet services require large, contiguous blocks of spectrum to work efficiently. First, a buyer’s decision about which block of spectrum to buy would limit the buyer to trading only with broadcasters holding licenses to parts of that block; it could be hard or impossible to find such a block in which all broadcasters were willing to trade. Second, each of these broadcasters would have “holdout power,” meaning the broadcaster could demand an exorbitant payment in exchange for allowing the deal to proceed. The likely result would have been very little trade, even if broadcasters valued the spectrum much less than potential buyers.

A 2012 Act of Congress implemented a clever solution to this problem. It guaranteed each broadcaster interference-free coverage in its broadcast area on *some* channel, but not necessarily on its currently used channel. This meant that if a broadcaster was unwilling to sell its license, it could instead be moved to another channel, solving the holdout problem. To free up the channel that would permit this move to take place, broadcast rights could be bought from another station in the appropriate geographical area, even if this second station did not hold a license for spectrum due for reallocation. In what follows, we call such an interference-free reassignment of channels to stations a *feasible repacking*.

These trades and channel reassignments were coordinated via a novel spectrum auction run by the FCC between March 2016 and April 2017, dubbed the *Incentive Auction*. It consisted of two interrelated parts. The first was a *forward auction* that sold large blocks of upper UHF spectrum to interested buyers in a manner similar to previous auctions of unallocated spectrum. The key innovation was the second

This paper builds in part on an AAAI 2016 conference publication by the same authors (see Fréchet et al.¹¹).

part: a *reverse auction* that was specially designed to perform well in the Incentive Auction.^{25, 28} It identified both a set of broadcasters who would voluntarily give up their broadcast rights and prices at which they would be compensated, simultaneously ensuring that all remaining broadcasters could feasibly be repacked in the unsold spectrum. The choice of how much spectrum to reallocate, called the *clearing target*, linked these two parts: the incentive auction alternated between reverse and forward auction stages with progressively shrinking clearing targets until revenue generated by the forward auction covered the cost of purchasing and reassigning stations in the reverse auction.

We now describe the reverse auction's rules in more detail. First, all participating stations are given initial price quotes and respond either that they agree to sell their broadcast rights at the quoted price or that they "exit the auction" (decline to participate), meaning that they will be guaranteed some interference-free channel. The auction then repeatedly iterates over the active bidders. Every time a bidder i is considered, the software first checks whether i can be feasibly repacked along with all exited stations. If such a feasible repacking exists, i is given a (geometrically) lower price quote and again has the options of accepting or exiting. Otherwise, i is *frozen*: its price stops descending and it is no longer active. The auction ends when all bidders are either frozen, exited, or receive price quotes of zero.

The problem of checking the feasibility of repacking is central to the reverse auction, likely to arise tens of thousands of times in a single auction. Unfortunately, this problem is NP-complete, generalizing graph coloring. The silver lining is that interference constraints were known in advance—they were derived based on the locations and broadcast powers of existing television antennas—and so it was reasonable to hope for a heuristic algorithm that achieved good performance on the sorts of problems that would arise in a real auction. However, identifying an algorithm that would be fast and reliable enough to use in practice remained challenging. Since each feasibility check depends on the results of those that came before—if a station is found to be frozen, it cannot exit—these problems must be solved sequentially. Time constraints for the auction as a whole required that the auction iterate through the stations at least twice a day, which worked out to a time cutoff on the order of minutes. It was thus inevitable that some problems would remain unsolved. Luckily, the auction design is robust to such failures, treating them as proofs of infeasibility at the expense of raising the cost required to clear spectrum.

This paper describes our experience building SATFC 2.3.1, the feasibility checker used in the reverse auction. We leveraged automatic algorithm configuration approaches to derive a portfolio of complementary algorithms that differ in their underlying (local and complete) search strategies, Satisfiability (SAT) encodings, constraint graph decompositions, domain-specific heuristics, and use of a novel caching scheme. We use the term "deep optimization" to refer to this approach,^a with the goal of emphasizing its conceptual similarity to deep learning. Classical machine learning relied on features crafted based on expert insight, model families selected manually, and model hyperparameters tuned

essentially by hand. Deep learning has shown that it is often possible to achieve substantially better performance by relying less on expert knowledge and more on enormous amounts of computation and huge training sets. Specifically, deep learning considers parametric models of very high dimension, using expert knowledge only to identify appropriate invariances and model biases, such as convolutional structure. (In some cases it is critical that these models be "deep" in the sense of having long chains of dependencies between parameters, but in other cases great flexibility can be achieved even with models only a couple of levels deep; e.g., Zhang et al.³⁵) We argue that a similar dynamic applies in the case of heuristic algorithms for discrete optimization, which aim to achieve good performance on some given dataset rather than in the worst case. Traditionally, experts have designed such heuristic algorithms by hand, iteratively conducting small experiments to refine their designs. We advocate an approach in which a computationally intensive procedure is used to search a high-dimensional space of parameterized algorithm designs to optimize performance over a large set of training data. We aim to minimize the role played by expert knowledge, restricting it to the identification of parameters that could potentially lead to fruitful algorithm designs. We also encourage deep dependencies via chains of parameters each of whose meaning depends on the value taken by one or more parents.

Overall, this paper demonstrates the value of the deep optimization approach via the enormous performance gains it yielded on the challenging and socially important problem of spectrum repacking. After formally stating the station repacking problem, we define our large algorithm design space and the search techniques we used. We assess the results on problems that arose in runs of our new open-source reverse auction simulator, investigating both our solver's runtime and its impact on economic outcomes.

2. THE STATION REPACKING PROBLEM

We now describe the station repacking problem in more detail.^b Each television station in the US and Canada $s \in \mathcal{S}$ is currently assigned to a channel $c_s \in \mathcal{C} \subseteq \mathbb{N}$ that ensures that it will not excessively interfere with other, nearby stations. (Although Canadian stations did not participate in the auction, they were eligible to be reassigned new channels.) The FCC determined pairs of channel assignments that would

^a There exists a large body of prior work that investigates the use of algorithm configuration to design novel algorithms from large, parameterized spaces (some of which, indeed, we have coauthored); we believe, however, that the work described in this paper is the most consequential application of such techniques to date. Much of the literature just mentioned focuses on algorithm configuration tools^{2, 15, 16, 20, 26, 33} (which we take as given in this paper) rather than algorithm design methodology. Most work in the latter vein either addresses the much broader problem of algorithm synthesis (e.g., Di Gaspero and Schaerf; Monette et al.²⁹; Westfold and Smith³²) or defines the overall approach only implicitly (e.g., KhudaBukhsh et al.²²). The most prominent exception is "programming by optimization,"¹⁴ however, it emphasizes connections to software engineering and does not limit itself to parametric design spaces.

^b Similar problems have been studied in other contexts, falling under the umbrella of *frequency assignment problems*. See for example, Ref. Aardal et al.¹ for a survey and a discussion of applications to mobile telephony, radio and TV broadcasting, satellite communication, wireless Local Area Networks (LAN), and military operations. We are not aware of other published work that aims to optimize feasibility checking in the Incentive Auction setting.

cause harmful interference based on a complex, grid-based physical simulation (“OET-69” Ref. FCC7); this pairwise constraint data is publicly available.⁹ Let $\mathcal{I} \subseteq (S \times C)^2$ denote a set of *forbidden station–channel pairs* $\{(s, c), (s', c')\}$, each representing the proposition that stations s and s' may not concurrently be assigned to channels c and c' , respectively. The effect of the auction was to remove some broadcasters from the airwaves and to reassign channels to the remaining stations from a reduced set. This reduced set was defined by a *clearing target*, fixed for each stage of the reverse auction, corresponding to some channel $\bar{c} \in C$ such that all stations are only eligible to be assigned channels from $\bar{C} = \{c \in C | c < \bar{c}\}$. Each station can only be assigned a channel on a subset of \bar{C} , given by a *domain* function $\mathcal{D} : S \rightarrow 2^{\bar{C}}$ that maps from stations to these reduced sets. The *station repacking problem* is then the task of finding a repacking $\gamma : S \rightarrow \bar{C}$ that assigns each station a channel from its domain that satisfies the interference constraints: That is, for which $\gamma(s) \in \mathcal{D}(s)$ for all $s \in S$, and $\gamma(s) = c \Rightarrow \gamma(s') \neq c'$ for all $\{(s, c), (s', c')\} \in \mathcal{I}$. A problem instance thus corresponds to a set of stations $S \subseteq S$ and channels $C \subseteq \bar{C}$ into which they must be packed, with domains \mathcal{D} and constraints \mathcal{I} implicitly being restricted to S and C ; we call the resulting restrictions D and I .

Why should we hope that this (NP-complete) problem can be solved effectively in practice? First, we only need to be concerned with problems involving subsets of a fixed set of stations and a fixed set of interference constraints: those describing the television stations currently broadcasting in the United States and Canada. Let us define the *interference graph* as an undirected graph in which there is 1 vertex per station and an edge exists between two vertices s and s' if the corresponding stations participate together in any interference constraint: That is, if there exist $c, c' \in C$ such that $\{(s, c), (s', c')\} \in \mathcal{I}$. Figure 1 shows the Incentive Auction interference graph. As it turns out, interference constraints come in 2 kinds. *Co-channel constraints* specify that 2 stations may not be assigned to the same channel; *adjacent-channel constraints* specify that two stations may not be assigned to two nearby channels. Hence, any forbidden station–channel pairs are of the form $\{(s, c), (s', c + i)\}$ for some stations $s, s' \in S$, channel $c \in C$, and $i \in \{0, 1, 2\}$. Furthermore, channels can be partitioned into 3 equivalence classes: Low

Figure 1. Interference graph visualizing the FCC’s constraint data⁹ (2 990 stations; 2 575 466 channel-specific interference constraints).



Very High Frequency (LVHF) (channels 1–6), High Very High Frequency (HVHF) (channels 7–13), and UHF (channels 14–51) with the property that no interference constraint involves channels in more than one band.

Second, note that we are not interested in optimizing worst-case performance even given our fixed interference graph, but rather in achieving good performance on the sort of instances generated by actual reverse auctions. These instances depend on the order in which stations exit the auction, which depends on stations’ valuations, which depend in turn (among many other factors) on the size and character of the population reached by their broadcasts. The distribution over repacking problems is hence far from uniform.

Third, descending clock auctions repeatedly generate station repacking problems by adding a single station s^+ to a set S^- of provably repackable stations. This means that every station repacking problem $(S^- \cup \{s^+\}, C)$ comes with a partial assignment $\gamma^- : S^- \rightarrow C$ that we know is feasible on restricted station set S^- ; we will see in what follows that this fact is extremely useful.

Finally, many repacking problems are trivial: in our experience, problems involving only Very High Frequency (VHF) channels can all be solved quickly; furthermore, the vast majority of UHF problems can be solved greedily simply by checking whether s^+ can be augmented directly with γ^- . However, solving the remaining problems is crucial to the economic outcomes achieved by the auction (as we shown in Section 6). In what follows, we restrict ourselves to “non-trivial” UHF problems that cannot be solved by greedy feasibility checking.

3. A DEEP OPTIMIZATION APPROACH

As we shown in our experiments (see Section 5), off-the-shelf solvers could not solve a large enough fraction of station repacking problems to be effective in practice. To do better, we needed a customized algorithm optimized to perform well on our particular distribution of station repacking problems. We built our algorithm via the deep optimization approach, meaning that we aimed to use our own insight only to identify design ideas that showed promise, relegating the work of combining these ideas and evaluating the performance of the resulting algorithm on realistic data (see Section 4) to an automatic search procedure.

3.1. The design space

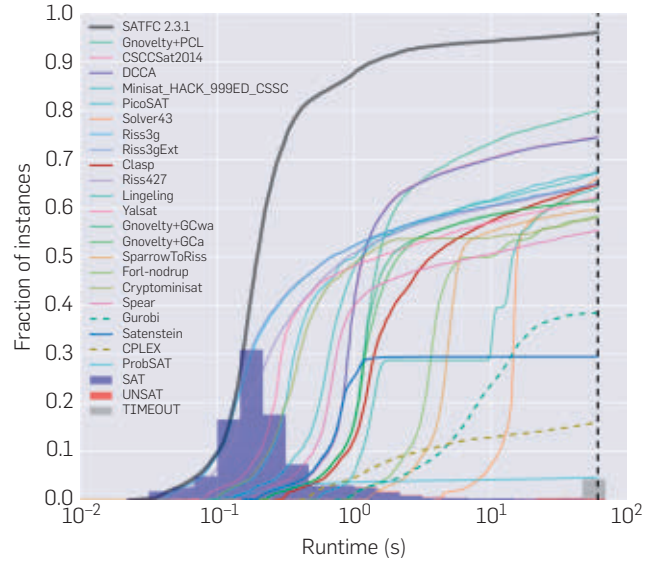
Our first task was thus to identify a space of algorithm designs to consider. This was not just a pen- and-paper exercise, since each point in the space needed to correspond to runnable code. We focused on encoding station repacking as a propositional SAT problem. The SAT formalism is well suited to station repacking, which is a pure feasibility problem with only combinatorial constraints. (It may also be possible to achieve good performance with Mixed-Integer Program (MIP) or other encodings; we did not investigate such alternatives in depth.) The SAT reduction is straightforward: given a station repacking problem (S, C) with domains D and interference constraints I , we create a Boolean variable $x_{s,c} \in \{\top, \perp\}$ for every station–channel pair $(s, c) \in S \times C$, representing the proposition that station s is assigned to channel c . We

then create three kinds of clauses: (1) $\bigvee_{d \in D(s)} x_{s,d} \forall s \in S$ (each station is assigned at least one channel); (2) $\neg x_{s,c} \vee \neg x_{s,c'} \forall s \in S, \forall c, c' \neq c \in D(s)$ (each station is assigned at most one channel); (3) $\neg x_{s,c} \vee \neg x_{s',c'} \forall \{(s, c), (s', c')\} \in I$ (interference constraints are respected). Note that (2) is optional: if a station is assigned more than one channel, we can simply pick one channel to assign it from among these channels arbitrarily. We thus created a parameter indicating whether to include these constraints. In the end, a SAT encoding of a problem involving all stations at a clearing target of 36 involved 73 187 variables and 2 917 866 clauses.

Selecting solvers. Perhaps the most important top-level parameter determines which SAT solver to run. (Of course, each such solver will have its own (deep) parameter space; other parameters will describe design dimensions orthogonal to the choice of solver, as we will discuss in what follows.) The SAT community has developed a very wide variety of solvers and made them publicly available (see e.g., Jarvisalo et al.¹⁹). In principle, we would have made it possible to choose every solver that offered even reasonable performance. However, doing so would have been too costly from the perspective of software integration and (especially) reliability testing. We thus conducted initial algorithm configuration experiments (see Section 3.2) on 20 state-of-the-art SAT solvers, drawn mainly from SAT solver competition entries collected in ACLib.¹⁸ We illustrate the performance of their default configurations as shown in Figure 2; most improved at least somewhat from their default configurations as a result of algorithm configuration. We identified two solvers that ended up with the strongest post-configuration performance—one complete and one based on local search—both of which have been shown in the literature to adapt well to a wide range of SAT domains via large and flexible parameter spaces. Our first solver was `clasp`,¹² an open-source solver based on conflict-driven nogod learning (98 parameters). Our second was the open-source SATenstein framework,²² which allows arbitrary composition of design elements taken from a wide range of high-performance stochastic local search solvers (90 parameters), including DCCA and `gNovelty+` (of which three solvers in our set are variants).

Using the previous solution. While adapting `clasp` and SATenstein to station repacking data yielded substantial performance improvements, neither reached a point sufficient for deployment in the real auction. To do better, it was necessary to leverage specific properties of the incentive auction problem. Rather than committing to specific speedups, we exposed a wide variety of possibilities via further parameters. We began by considering two methods for taking advantage of the existence of a partial assignment γ^- . The first method checks whether a simple transformation of γ^- is enough to yield a satisfiable repacking. Specifically, we construct a small SAT problem in which the stations to be repacked are s^+ and all stations $\Gamma(s^+) \subseteq S$ neighboring s^+ in the interference graph, fixing all other stations $S \setminus \Gamma(s^+)$ to their assignments in γ^- . Any solution to this reduced problem must be a feasible repacking; however, if the reduced problem is infeasible we cannot conclude anything. However (depending on the value of a parameter), we can keep searching: unfixing all stations that neighbor a station in $\Gamma(s^+)$, etc.

Figure 2. Empirical Cumulative Density Function (ECDF) of runtimes for default configurations of MIP and SAT solvers and for SATFC 2.3.1. The curves show fraction of instances solved (y axis) within different amounts of time (x axis; note the log scale). The legend is ordered by percentage of problems solved before the cutoff. The histogram indicates density of SAT and UNSAT instances binned by their (fastest) runtimes; unsatisfiable instances constituted fewer than 1% of solved instances.



Our second method uses γ^- to initialize local search solvers. Such solvers search a space of complete variable assignments, typically following gradients to minimize an objective function such as the number of violated constraints, with occasional random steps. They are thus sensitive to their starting points. Optionally, we can start at the assignment given by γ^- (randomly initializing variables pertaining to s^+). We can also optionally redo this initialization on some fraction of random restarts.

Problem simplification. Next, we considered three preprocessing techniques that can simplify station repacking problems. First, we added the option to run the arc consistency algorithm, repeatedly pruning values from each station’s domain that are incompatible with every channel on a neighboring station’s domain.

Second, we enabled elimination of unconstrained stations. A station s is unconstrained if, given any feasible assignment of all of the other stations in $S \setminus s$, there always exists some way of feasibly repacking s . Unconstrained stations can be removed without changing a problem’s satisfiability status. Various algorithms exist for identifying unconstrained stations; we determine this choice via a parameter. (All such stations can be found via a reduction to the polytime problem of eliminating variables in a binary Constraint Satisfaction Problems (CSP);³ various sound but incomplete heuristics run more quickly but identify progressively fewer unconstrained stations.)

Third, the interference graph induced by a problem may consist of multiple connected components; we can optionally run a linear-time procedure to separate them into distinct SAT problems. We only need to solve the component to which s^+ belongs: γ^- supplies feasible assignments for all others. Arc consistency and unconstrained station removal can

simplify the interference graph by removing edges and nodes respectively. This can shrink the size of the component containing s^+ and make this technique even more effective.

Containment caching. Finally, we know that every repacking problem will be derived from a restriction of the interference graph to some subset of S . We know this graph in advance of the auction; this suggests the possibility of doing offline work to precompute solutions. However, our graph has 2 990 nodes, and the number of restricted graphs is thus $2^{2990} \approx 10^{900}$. Thus, it is not possible to consider all of them offline.

Not every restricted problem is equally likely to arise in practice. To target likely problems, we could simply run a large number of simulations and cache the solution to every repacking problem encountered. Unfortunately, we found that it was extremely rare for problems to repeat across sufficiently different simulator inputs, even after running hundreds of simulations (generating millions of instances and costing years of CPU time). However, we can do better than simply looking for previous solutions to a given repacking problem. If we know that S is repackable then we know the same is true for every $S' \subseteq S$ (and indeed, we know the packing itself—the packing for S restricted to the stations in S'). Similarly, if we know that S was not packable then we know the same for every $S' \supseteq S$. This observation dramatically magnifies the usefulness of each cached entry S , because each S can be used to answer queries about an exponential number of subsets or supersets. This is especially useful because sometimes it can be harder to find a repacking for subsets of S than it can be to find a repacking for S .

We call a cache meant to be used in this way a *containment cache*, because it is queried to determine whether one set contains another (i.e., whether the query contains the cache item or vice versa). To the best of our knowledge, containment caching is a novel idea. A likely reason why this scheme is not already common is that querying a containment cache is nontrivial: one cannot simply index entries with a hash function; instead, an exponential number of keys can match a given query. We were nevertheless able to construct an algorithm that solved this problem quickly in our setting. We observe that containment caching is applicable to any family of feasibility testing problems generated as subsets of a master set of constraints, not just to spectrum repacking.

In more detail, we maintain two caches, a *feasible cache* and an *infeasible cache*, and store each problem we solve in the appropriate cache. We leverage the methods from Section 3.1 to enhance the efficiency of our cache, storing full instances for SAT problems and the smallest simplified component for UNSAT problems. When asked whether it is possible to repack station set S , we first check whether a subset of S belongs to the infeasible cache (in which case the original problem is infeasible); if we find no matches, we decompose the problem into its smallest simplified component and check if the feasible cache contains a superset of those stations, in which case the original problem is feasible.

3.2. Searching the design space

Overall, our design space had 191 parameters, nested as much as four levels deep. We now describe how we searched

this space to building a customized solver. Identifying a set of parameters that optimize a given algorithm's performance on a given dataset is called *algorithm configuration*. There exist a wide variety of algorithm configuration tools.^{2,15,16,26} We used Sequential Model-based Algorithm Configuration (SMAC),¹⁵ the publicly available method that arguably achieves the best performance (see e.g., Hutter et al.¹⁷). SMAC uses the “Bayesian optimization” approach of interleaving random sampling and the exploration of algorithm designs that appear promising based on a learned model.

Unfortunately, even after performing algorithm configuration, it is rare to find a single algorithm that outperforms all others on instances of an NP-complete problem such as SAT. This inherent variability across solvers can be exploited by *algorithm portfolios*.^{13,31,34} Most straightforwardly, one selects a small set of algorithms with complementary performance on problems of interest and, when asked to solve a new instance, executes them in parallel. Of course, we wanted to construct such algorithm portfolios automatically as part of our deep optimization approach. We did this by using a method called Hydra³³ which runs iteratively, at each step directing the algorithm configurator to optimize marginal gains over the given portfolio. This allows Hydra to find algorithms that may perform poorly overall but that complement the existing portfolio. Overall, we ran Hydra for eight steps, thereby producing a portfolio of novel solvers (dubbed SATFC) that could run on a standard 8-core workstation. The Incentive Auction used SATFC 2.3.1, which is available online at <https://github.com/FCC/SATFC>.

4. DATA FROM AUCTION SIMULATIONS

During the development of SATFC the FCC shared with us a wide range of anonymized problem instances that arose in auction simulations they performed in order to validate the auction design. These formed the “training set” we used in the deep optimization process when constructing SATFC 2.3.1. These simulations explored a very narrow set of answers to the questions of which stations would participate and how bidders would interact with the auction mechanism; they do not represent a statement either by us or by the FCC about how these questions were resolved in the real auction (indeed, by law the answers will not be revealed to us or to the public for two years). It is of course impossible to guarantee that variations in the assumptions would not have yielded computationally different problems.

While SATFC 2.3.1 is itself open-source software, it is unfortunately impossible for us to share the data that was used to build it. In this paper, we have opted for what we hope is the next best thing: *evaluating* SATFC 2.3.1 and various alternatives using a publicly available test set. We thus wrote our own reverse auction simulator and released it as open source software (see <http://cs.ubc.ca/labs/beta/Projects/SATFC>). We used this simulator to simulate 20 auctions, in each case randomly sampling bidder valuations from a publicly available model⁶ using parameters obtained directly from its authors. This model specifies stations' values for broadcasting in UHF, $v_{s,UHF}$. Of course, a station has no value for going off air: $v_{s,OFF} = 0$. In some cases the reverse auction can reassign a UHF station to a channel in 1 of 2 less valuable VHF

bands (LVHF, HVHF) in exchange for lesser compensation. We assume that $v_{s,HVHF} = \frac{2}{3} v_{s,UHF}$ and $v_{s,LVHF} = \frac{1}{3} v_{s,UHF}$. We excluded from our simulator all stations for which the authors of the model were unable to supply us with parameters: stations outside the mainland U.S. and Hawaii, all U.S. VHF stations, and an additional 25 U.S. UHF stations. This left us with 1 638 eligible U.S. stations. We further included all Canadian stations in our simulations: because the auction rules forbade them from being paid to leave their home bands, we did not need to model their valuations. Specifically, from Canada we included 113 LVHF stations, 332 HVHF stations, and 348 UHF stations.

We set the auction's opening prices to the values announced by the FCC in November 2015.¹⁰ We assumed that stations chose to participate in the reverse auction if their opening price offer for going off air was greater than their valuation for remaining on air in their current band. We assumed that stations always selected the option that myopically maximized their utility. We used the interference constraints and station domains announced by the FCC in November 2015.⁹ For each simulation, we used the largest clearing target for which we could find a feasible assignment for the non-participating stations; in all cases this led to a clearing target of 84 Mhz, corresponding to a maximum allowable channel of 36. We note that this is the amount of spectrum actually cleared by the Incentive Auction. Just like the real auction, an auction simulator needs a feasibility checker to determine which price movements are possible. We used SATFC 2.3.1 with a cutoff of 60 sec. We sampled 10 000 "nontrivial" UHF problems uniformly at random from all of the problems across all simulations to use as our dataset, where we defined nontrivial problems as those that could not be solved by greedily augmenting the previous solution. Fewer than 3% of UHF problems in our simulations were nontrivial. This test set consisted of 9 482 feasible problems, 121 infeasible problems, and 397 problems that timed out at our one min cutoff and therefore have unknown feasibility.

5. RUNTIME PERFORMANCE

We now evaluate SATFC's performance by contrasting it with various off-the-shelf alternatives. The FCC's initial investigations included modeling the station repacking problem as a MIP and using off-the-shelf solvers paired with problem-specific speedups.⁸ Unfortunately, the problem-specific elements of this solution were not publicly released, so we cannot evaluate them in this article. Instead, to assess the feasibility of a MIP approach, we ran what are arguably the two best-performing MIP solvers—CPLEX and Gurobi—on our test set of 10 000 non-trivial instances. To encode the station repacking problem as a MIP, we created a variable $x_{s,c} \in \{0, 1\}$ for every station–channel pair, representing the proposition that station s is assigned to channel c . We imposed the constraints $\sum_{c \in D(s)} x_{s,c} = 1 \forall s \in S$ and $x_{s,c} + x_{s',c'} \leq 1 \forall \{(s, c), (s', c')\} \in I$, ensuring that each station is assigned to exactly 1 channel and that interference constraints are not violated. Both MIP solvers solved under half of the instances within our cutoff time of one min; the results are shown in Figure 2. Such performance would likely have been insufficient for deployment in practice, since it implies unnecessarily high payments to many stations.

As already discussed, there exist a wide variety of SAT solvers that are available for use off the shelf. Figure 2 illustrates the performance of the 20 state-of-the-art solvers we considered in our initial configuration experiments in their default configurations. With few exceptions, the SAT solvers outperformed the MIP solvers, as can be seen by comparing the solid and dashed lines as shown in Figure 2. However, run-times and percentages of instances solved by the cutoff time were still not good enough for us to recommend deployment of any of these solvers in the actual auction. The best solver in its default configuration, Gnovelty+PCL, was able to solve the largest number of problems—79.96%—within the cutoff. (As mentioned earlier, the SATenstein design space includes Gnovelty+PCL alongside many other solvers.) The parallel portfolio of all 20 solvers from Figure 2 was little better, being able to solve only 81.58% of problems.

We now turn to SATFC 2.3.1. This 8-solver parallel portfolio stochastically dominated every individual solver that we considered and achieved very substantial gains after a few tenths of a second. It solved 87.73% of the problems in under a second and 96.03% within the 1 min cutoff time. The histogram at the bottom of the figure indicates satisfiability status of instances solved by SATFC grouped by runtime; our instances were overwhelmingly satisfiable.

6. IMPACT ON ECONOMIC OUTCOMES

We now ask whether SATFC's improved performance is likely to have translated into a better economic outcome in the Incentive Auction, assessing both cost and efficiency. Preliminary work³⁰ examined this question using a simplified simulator that restricted bidding to the UHF band and considered a small set of stations in the vicinity of New York City (described in detail below). Here we perform larger scale versions of these initial experiments and then use our enhanced reverse auction simulator to run national-scale simulations. The cost of an auction is the sum of payments to $\sum_{s \in S} v_{s, \text{pre}(s)} - v_{s, \text{post}(\gamma, s)}$, the winning stations. To assess efficiency we measured the total value lost $\sum_{s \in S} v_{s, \text{pre}(s)} - v_{s, \text{post}(\gamma^*, s)}$ by the auction, comparing the sums of values of stations for their allocated bands both before and after the auction.^c If we can find an efficient repacking γ^* , then we can compute the additional fraction of value lost by some other repacking γ . We call this the *value loss ratio*: where $\text{pre}(s)$ returns the band to which s was assigned before the auction and $\text{post}(\gamma, s)$ returns either the band to which s is assigned under channel assignment γ or OFF if s is not assigned to a band under γ . When it is intractable to compute γ^* , we resort to comparing the absolute value loss between different assignments.

Given our interest in the efficiency of the reverse auction, it is natural to compare it to the Vickrey–Clarke–Groves (VCG) mechanism, which always chooses the optimal packing γ^* . VCG pays losing stations nothing and pays each winning station s the difference between the sum of values of

^c The more standard measure of efficiency—the sum of stations' values for their allocated bands—has the same optimum. Value loss has the advantage that it is influenced only by stations that a feasibility checker is unable to repack in their home bands; it is thus more appropriate for comparing feasibility checkers. The more standard measure is sensitive to changes in the values of easy-to-repack stations, even those that do not participate in any interference constraints.

stations other than s for γ^* and the sum of the same stations' values for a packing that is optimal subject to the constraint that s does not win. We identified these optimal packings using the MIP encoding from Section 5 with 2 changes. First, we added the objective of maximizing the aggregate values of the participating stations: maximize $\sum_{s \in S_{\text{bidding}}} \sum_{c \in D(s)} x_{s,c} \cdot v_{s,\text{band}(c)}$, where $\text{band}(c)$ is a function that returns the corresponding band for a given channel. Second, we allowed the option of not assigning a channel to a bidding station.

6.1. Greater New York city simulations

Unfortunately, it was impossible to solve these optimization problems at a national scale, even given several days of computing time. We therefore constructed tractable problems by restricting ourselves to stations in the vicinity of New York City, which we chose because it corresponds to one of the most densely connected regions in the interference graph. More specifically, we dropped all Canadian stations and restricted ourselves to the UHF band using the smallest possible clearing target (maximum allowable channel of 29). Using the interference graph induced by these restrictions, we then dropped every station whose shortest path length to a station in New York City exceeded 2. The result was a setting with 218 stations including those in Boston, Philadelphia and Washington DC. The setting contained 78 499 channel-specific interference constraints, yielding a MIP encoding with 2 465 variables and 78 717 constraints.

We randomly generated 20 different valuation profiles, using the methodology described in Section 4 but restricting ourselves to the stations in the restricted interference graph. For each valuation profile, we conducted 5 simulations. The first was of a VCG auction; we computed allocations and payments using CPLEX, solving all MIPs optimally to within 10^{-6} absolute MIP gap tolerance. We also ran 4 reverse auction simulations for each valuation profile, varying the feasibility checker to consider 3 alternatives to SATFC. The first is the greedy feasibility checker, which represents the simplest reasonable feasibility checker and thus serves as a baseline. The

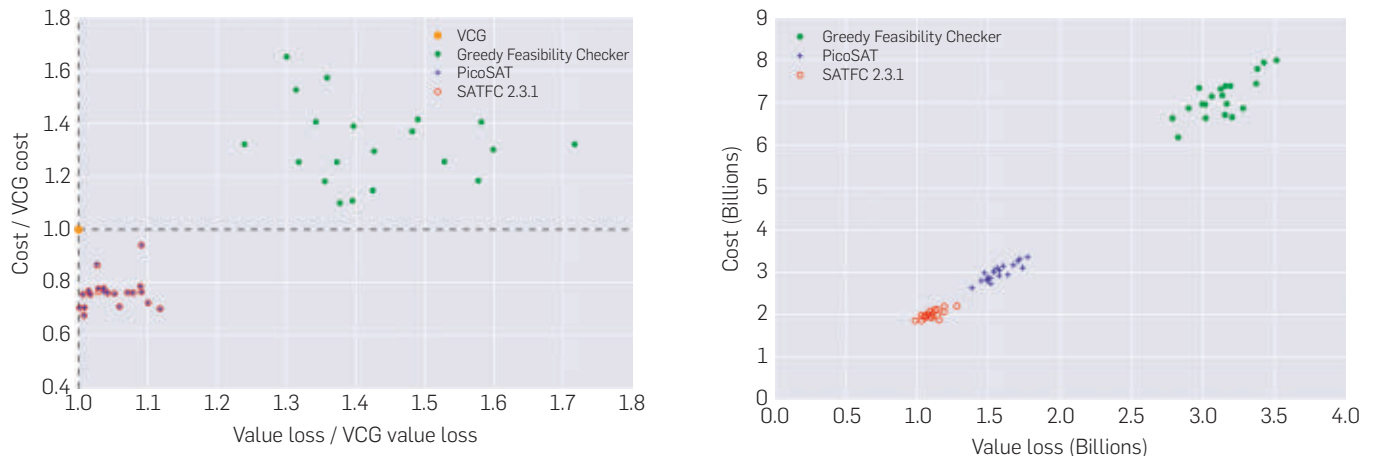
second is the default configuration of PicoSAT. To our knowledge, alongside MIP approaches this is the only other solver that has been used in publications on the Incentive Auction,^{4, 21} probably because we showed it to be the best among a set of alternatives in an early talk on the subject.²⁴ The third is the default configuration of Gurobi, the best performing MIP solver on our test set, which we include as a proxy for the MIP approach initially investigated by the FCC.

In total, our 100 simulations consumed over five years of CPU time (dominated by the VCG simulations). Figure 3 (left) illustrates the results. Each point shows the value loss (x axis) and cost (y axis) of a single simulation; in both cases, these quantities are normalized by the corresponding quantity achieved by VCG for the same valuation profile. The SATFC simulations had a mean value loss ratio of 1.048 and a mean cost ratio of 0.760, indicating that the reverse auction achieved nearly optimal efficiency at much lower cost than VCG. The PicoSAT results were nearly identical, differing in only 2 value profiles, and only slightly. The Gurobi results were again extremely similar but marginally worse, with a mean value loss ratio of 1.053 and a mean cost ratio of 0.762. All of the SATFC, PicoSAT, and Gurobi runs dominated the greedy runs according to both metrics; on average, reverse auctions based on greedy cost 1.742 times more and lost 1.366 times as much broadcaster value than those based on SATFC. Despite these differences, all of the solvers were able to solve a very large fraction of the feasibility checking problems encountered in their respective simulations (which took different trajectories once 2 solvers differed in their ability to solve a given problem): 99.978%, 99.945%, 99.948%, and 99.118% for SATFC, PicoSAT, Gurobi, and greedy respectively (including trivial problems).

6.2. National simulations

We were more interested in economic outcomes at the national scale, even though we could not simulate VCG in such a large setting. We generated 20 valuation profiles for our full set of stations and ran reverse auction simulations using our 4 feasibility checkers. In total, these experiments

Figure 3. Comparing value loss and cost of the greedy feasibility checker, Gurobi, PicoSAT, and SATFC 2.3.1 for 20 different value profiles. (Left) Fraction of VCG cost versus fraction of VCG value loss for Greater New York City simulations. All VCG points lie at (1,1). (Right) Value loss and cost for national simulations.



consumed over 13 days of CPU time.

All solvers were again able to solve a large fraction of the problems they encountered: 99.902%, 99.765%, 99.7433%, and 98.031% for SATFC, PicoSAT, Gurobi, and greedy respectively (including trivial problems). The economic impact of these differences is illustrated as shown in Figure 3 (right). In this graph, x - and y -axis values correspond to unnormalized value loss and cost respectively. Each SATFC, Gurobi, and PicoSAT simulation again dominated its greedy counterpart in both efficiency and cost. Averaging over all of our observations, reverse auctions based on greedy feasibility checking cost 3.550 times (\$5.114 bn) more and lost 2.850 times as much (\$2.030 bn) broadcaster value than those based on SATFC. At the larger scale we also found that SATFC dominated PicoSAT and Gurobi in every simulation: on average, PicoSAT auctions cost 1.495 times (\$987 million) more and lost 1.427 times as much (\$469 million) value than SATFC auctions and Gurobi auctions cost 2.195 times (\$2.390 bn) more and lost 2.017 times as much (\$1.117 bn) value than SATFC auctions.

7. CONCLUSION

Station repacking in the Incentive Auction is a difficult but important problem, with progress translating into significant gains in both government expenditures and social welfare. We designed a customized solution to this problem using an approach we dub deep optimization. Specifically, we drew on a large parameterized design space to construct a strong portfolio of heuristic algorithms: SATFC 2.3.1, an open-source solver that was used in the real auction. To evaluate it for this paper, we conducted experiments with a new reverse auction simulator. We found that replacing SATFC with an off-the-shelf feasibility checker resulted in both efficiency losses and increased costs. It thus appears likely that our efforts led to significant economic benefits to broadcasters, the US government, and the American public.

Acknowledgments

We gratefully acknowledge support from Auctionomics and the FCC; valuable conversations with Paul Milgrom, Ilya Segal, and James Wright; help from Peter West in conducting some of the experiments reported in Section 6; contributions (mostly in the form of code) from past research assistants Nick Arnosti, Guillaume Saulnier-Comte, Ricky Chen, Alim Virani, Chris Cameron, Emily Chen, Paul Cernek; experimental infrastructure assistance from Steve Ramage; and help gathering data from Ulrich Gall, Rory Molinari, Karla Hoffman, Brett Tarnutzer, Sasha Javid, and others at the FCC. This work was funded by Auctionomics and by Natural Sciences and Engineering Research Council (NSERC) via a Discovery Grant and an E.W.R. Steacie Fellowship; it was conducted in part at the Simons Institute for Theoretical Computer Science at UC Berkeley.

References

- Aardal, K.I., Van Hoesel, S.P., Koster, A.M., Mannino, C., Sassano, A. Models and solution techniques for frequency assignment problems. *Annals of Operations Research* 153, 1 (2007), 79–129.
- Ansótegui, C., Sellmann, M., Tierney, K. A gender-based genetic algorithm for the automatic configuration of algorithms. In *Conference on Principles and Practice of Constraint Programming (CP)* (2009), 142–157.
- Cohen, D.A., Cooper, M.C., Escamocher, G., Živny, S. Variable and value elimination in binary constraint satisfaction via forbidden patterns. *Journal of Computer and System Sciences* 81, 7 (2015), 1127–1143.
- Cramton, P., Lopez, H., Matec, D.,

- Sujarittanonta, P. Design of the reverse auction in the broadcast incentive auction. Working Paper (2015), <http://www.cramton.umd.edu/papers/2015-2019/cramton-reverse-auction-design-fcc-comment-pn.pdf>, 2015.
- Di Gaspero, L., Schaerf, A. Easysyn++: A tool for automatic synthesis of stochastic local search algorithms. In *Conference on Engineering Stochastic Local Search Algorithms (SLS)* (2007), 177–181.
- Doraszelski, U., Seim, K., Sinkinson, M., Wang, P. Ownership concentration and strategic supply reduction. Working Paper, <http://www.nber.org/papers/w23034>, National Bureau of Economic Research, January 2017.
- FCC. Office of engineering and technology releases and seeks comment on updated OET-69 software. *FCC Public Notice*, DA 13-138, February 2013.
- FCC. Information related to incentive auction repacking feasibility checker. *FCC Public Notice*, DA 14-3, 1 2014.
- FCC. Repacking constraint files (2015). http://data.fcc.gov/download/incentive-auctions/Constraint_Files/, 2015. Accessed 2015-11-20.
- FCC. Reverse auction opening prices (2015). http://wireless.fcc.gov/auctions/incentive-auctions/Reverse_Auction_Opening_Prices_111215.xlsx, 2015. Accessed 2015-11-15.
- Fréchet, A., Newman, N., Leyton-Brown, K. Solving the station repacking problem. In: *AAAI Conference on Artificial Intelligence* 8 (2016), 702–709. <http://dl.acm.org/citation.cfm?id=3015812.3015917>, 3015917. Frechette:2016:SR:3015812.3015917.
- Gebser, M., Kaufmann, B., Neumann, A., Schaub, T. clasp: A conflict-driven answer set solver. In *Logic Programming and Nonmonotonic Reasoning* (2007), 260–265.
- Gomes, C.P., Selman, B. Algorithm portfolios. *Artificial Intelligence* 126, 1 (2001), 43–62.
- Hoos, H.H. Programming by optimization. *Communications of the ACM* 55, 2 (Feb. 2012), 70–80.
- Hutter, F., Hoos, H.H., Leyton-Brown, K. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization Conference (LION)* (2011), 507–523.
- Hutter, F., Hoos, H.H., Leyton-Brown, K., Stützle, T. ParamILS: an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research* 36, 1 (2009), 267–306.
- Hutter, F., Lindauer, M., Bayless, S., Hoos, H., Leyton-Brown, K. Results of the configurable SAT solver challenge 2014. Technical report, University of Freiburg, Department of Computer Science, 2014.
- Hutter, F., López-Ibáñez, M., Fawcett, C., Lindauer, M., Hoos, H.H., Leyton-Brown, K., Stützle, T. A Clib: A benchmark library for algorithm configuration. In *Conference on Learning and Intelligent Optimization (LION)* (Springer, 2014), 36–40.
- Järvisalo, M., Le Berre, D., Roussel, O., Simon, L. The international SAT solver competitions. *Artificial Intelligence Magazine* 33, 1 (2012), 89–92.
- Kadioglu, S., Malitsky, Y., Sellmann, M., Tierney, K. ISAC—instance-specific algorithm configuration. In *European Conference on Artificial Intelligence (ECAI)* (2010), 751–756.
- Kearns, M., Dworkin, L. A computational study of feasible repackings in the FCC incentive auctions. *CoRR*, abs/1406.4837, 2014.
- KhudaBukhsh, A.R., Xu, L., Hoos, H.H., Leyton-Brown, K. SATenstein: Automatically building local search SAT solvers from components. *Artificial Intelligence* 232 (2016), 20–42.
- Knutson, R., Gryta, T. Verizon, AT&T may face bidding limits in spectrum auction. *Wall Street Journal* (Apr. 2014). <http://www.wsj.com/articles/SB10001424052702304626304579510154106120342>. Accessed 2016-12-12.
- Leyton-Brown, K. The viability of exact feasibility checking. Talk at the Stanford Institute for Economic Policy Research Conference on the design of the U.S. Incentive Auction for reallocating spectrum between wireless telecommunications and television broadcasting, February 2013.
- Li, S. Obviously strategy-proof mechanisms. Available at SSRN 2560028, 2015.
- López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Stützle, T., Birattari, M. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives* 3 (2016), 43–58.
- Milgrom, P. *Putting Auction Theory to Work*. Churchill Lectures in Economics. Cambridge University Press, 2004.
- Milgrom, P., Segal, I. Deferred-acceptance auctions and radio spectrum reallocation. In *ACM Conference on Economics and Computation (EC)* (2014), 185–186.
- Monette, J.-N., Deville, Y., Van Hentenryck, P. *Aeon: Synthesizing Scheduling Algorithms from High-Level Models*, 47 (Springer, Boston, 2009), 43–59.
- Newman, N., Leyton-Brown, K., Milgrom, P., Segal, I. Assessing economic outcomes in simulated reverse clock auctions for radio spectrum. Technical report, abs/1706.04324 (2017). <http://arxiv.org/abs/1706.04324>.
- Nudelman, E., Leyton-Brown, K., Andrew, G., Gomes, C., McFadden, J., Selman, B., Shoham, Y. Satzilla 0.9. Solver description, International SAT Competition, 2003.
- Westfold, S.J., Smith, D.R. Synthesis of efficient constraint-satisfaction programs. *The Knowledge Engineering Review* 16, 1 (March 2001), 69–84.
- Xu, L., Hoos, H.H., Leyton-Brown, K. Hydra: Automatically configuring algorithms for portfolio-based selection. In *AAAI Conference on Artificial Intelligence* (2010), 210–216.
- Xu, L., Hutter, F., Hoos, H.H., Leyton-Brown, K. SATzilla: portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research* 32 (June 2008), 565–606.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530, 2016.

Neil Newman, Alexandre Fréchet, and Kevin Leyton-Brown (newmanne, afrech, kevinlb@cs.ubc.ca), Department of Computer Science, University of British Columbia, Canada.

Technical Perspective

Can High Performance Be Portable?

By Manuel Chakravarty

THE DEVELOPMENT OF high-performance software has always suffered from a tension between achieving high performance on the one hand and portability and simplicity on the other hand. By specializing an algorithm for optimal performance, considering the memory hierarchy and other architectural particulars, we introduce architecture-specific detail. This obscures algorithmic structure and conflates the general with the specific, compromising simplicity and clarity. It also hurts portability to all but very similar architectures—simple changes, such as different cache sizes, can have substantial performance implications. Moreover, distinctly different architectures, such as CPUs versus GPUs versus DSPs, often require fundamentally different optimization strategies. As a result, high-performance code is difficult to write, debug, maintain, and port.

Numerous research efforts were aimed at addressing this issue by applying automatic code transformations and other forms of compiler optimizations. Ultimately, we would prefer the software developer simply code the algorithm and leave it to the machine to specialize that algorithm to any particular architecture for efficient execution. In this ideal world, portability is a matter of retargeting a compiler's optimization engine. Unfortunately, architectural complexity and the lack of architectural models that are simultaneously sufficiently detailed and tractable have prevented us from realizing this vision.

The following work by Ragan-Kelley et al. on the image processing language Halide explores a substantially different approach to architecture-specific code optimization. By shifting our perspective on how to express architectural constraints and how to generate high-performance code, it achieves the impressive feat of simplifying high-performance code, while at the same

The following work on the image processing language Halide explores a substantially different approach to architecture-specific code optimization.


time improving both portability and performance beyond that of traditional complex and non-portable approaches. This threefold success is indicative of a qualitative breakthrough, a definitive step forward in the state of the art.

Key to the authors' approach is the strict separation of the algorithmic code from an explicit specification of how to optimize that code for a given architecture. This specification, which they call the execution schedule, determines evaluation order, the amount of inlining, storage of intermediate data structures, and the choice between caching versus recomputation. With all the details of execution separated out, the remaining algorithmic code is purely functional.

This idea of separating the algorithmic code from the details of how to specialize that code for a specific architecture has been put forward before—for example, in the work on algorithmic skeletons. However, previous work lacks the clarity and simplicity of Halide and has failed to provide practical benefits at the scale of Halide. Its extraordinary success is due to the choice of the architectural specifics included in the schedule together with the specific optimization and code-generation technology informed by the schedule.

Ultimately, the work on Halide combines conceptual insight with the engineering prowess required to turn this insight into a distinct improvement for realistic applications. In this context, it is important to recognize that Halide started with a tight focus on a specific application area, namely image processing. While the concepts underlying Halide are more general, the tight domain focus has led to convincing applications—for example, Halide is used in Google's Pixel phone, Google Photos, and YouTube.

Looking ahead, the core question is to what extent Halide's approach can be generalized to applications outside of image processing and, more broadly, how Halide's programming model can be generalized. At its core, image processing is a subdomain of array programming. This provides a natural progression for Halide's approach to grow into other domains. First steps in this direction have been undertaken by successfully applying Halide, as is, to algorithms from linear algebra and machine learning. More challenging will be to extend the expressiveness of Halide to cover a broader range of computational forms than currently supported by its algorithmic language, while retaining the clear separation of algorithmic code from the execution schedule.

In addition to generalizing the application domain, a second question is the complexity of developing execution schedules. The authors note that, even in the current context, complex schedules require expert knowledge. While this is hardly surprising, as conventional high-performance optimization requires experts as well, machine support is a tantalizing option. The authors have begun to study this, but many questions remain. 

Manuel Chakravarty is a functional programming evangelist at Tweag I/O, Paris, France.

Copyright held by author.

Halide: Decoupling Algorithms from Schedules for High-Performance Image Processing

By Jonathan Ragan-Kelley, Andrew Adams, Dillon Sharlet, Connelly Barnes, Sylvain Paris, Marc Levoy, Saman Amarasinghe, and Frédo Durand

Abstract

Writing high-performance code on modern machines requires not just locally optimizing inner loops, but globally reorganizing computations to exploit parallelism and locality—doing things such as tiling and blocking whole pipelines to fit in cache. This is especially true for image processing pipelines, where individual stages do much too little work to amortize the cost of loading and storing results to and from off-chip memory. As a result, the performance difference between a naïve implementation of a pipeline and one globally optimized for parallelism and locality is often an order of magnitude. However, using existing programming tools, writing high-performance image processing code requires sacrificing simplicity, portability, and modularity. We argue that this is because traditional programming models conflate the computations defining the algorithm with decisions about intermediate storage and the order of computation, which we call the *schedule*.

We propose a new programming language for image processing pipelines, called Halide, that separates the algorithm from its schedule. Programmers can change the schedule to express many possible organizations of a single algorithm. The Halide compiler then synthesizes a globally combined loop nest for an entire algorithm, given a schedule. Halide models a space of schedules which is expressive enough to describe organizations that match or outperform state-of-the-art hand-written implementations of many computational photography and computer vision algorithms. Its model is simple enough to do so often in only a few lines of code, and small changes generate efficient implementations for x86, ARM, Graphics Processors (GPUs), and specialized image processors, all from a single algorithm.

Halide has been public and open source for over four years, during which it has been used by hundreds of programmers to deploy code to tens of thousands of servers and hundreds of millions of phones, processing billions of images every day.

1. INTRODUCTION

Computational photography and computer vision algorithms require highly efficient implementations to be used in practice, from power-constrained mobile devices to data centers processing billions of images. This is not a simple matter of programming in a low-level language such as C: even in C, the performance difference between naïve

and highly optimized image processing code for the same algorithm is often an order of magnitude. Unfortunately, optimization usually comes at a large cost in programmer time and code complexity, as computation must be globally reorganized to efficiently exploit the memory system (locality, e.g., in caches) and many execution units (parallelism, e.g., across threads and vector lanes).

Image processing pipelines are both wide and deep: they consist of many data-parallel stages that benefit hugely from parallel execution across pixels, but stages are often memory bandwidth limited—they do little work per load and store. Gains in performance and efficiency therefore come not just from optimizing the inner loops, but also from global program transformations that exploit producer-consumer locality down the pipeline. For example, computing a first stage on the entire image before processing the second stage causes cache misses when storing and loading the intermediate results; instead, an optimized pipeline might transform the organization of computation with tiling and fusion to compute both stages at the granularity of smaller image tiles that fit in cache.

Image processing exhibits a rich space of possible organizations of computation. The best choice of organization is architecture-specific. Implementations optimized for an x86 multicore and for a modern GPU often bear little resemblance to each other. There is also a tension between parallelism, locality, and storing versus recomputing intermediate values, which can make the ideal organization subtle and unpredictable.

Halide enables simpler programming of high-performance code by separating the intrinsic algorithm of an image processing pipeline from the decisions about how to run efficiently on a particular machine. Programmers may still specify the strategy for execution, since automatic optimization remains hard, but doing so is radically simplified by this split representation, which allows them to concisely express many optimization strategies without obfuscating

The original version of this work appeared in two papers, entitled “Decoupling Algorithms from Schedules for Easy Optimization of Image Processing Pipelines” published in *ACM Transactions on Graphics* 31 4, (July 2012), ACM, and “Halide: A Language and Compiler for Optimizing Parallelism, Locality, and Recomputation in Image Processing Pipelines” published in the *Proceedings of PLDI*, June 2013, ACM.

the code or accidentally modifying the algorithm itself.

This separation of concerns is important, and the ideal code organization nontrivial, even for a problem as simple as a 3×3 box filter implemented as separate horizontal and vertical passes (see Figure 1). We might write this in C++ as a sequence of two loop nests (see Figure 1.a). An efficient implementation on a modern CPU requires Single Instruction Multiple Data (SIMD) vectorization and multithreading. Once we start to exploit parallelism, however, the algorithm becomes bottlenecked on memory bandwidth. Computing the entire horizontal pass before the vertical pass destroys producer-consumer locality: horizontally blurred intermediate values are computed long before they are consumed by the vertical pass, doubling the storage and memory bandwidth required. Exploiting locality requires interleaving the two stages, for example by tiling and fusing the loops. Tiles must be carefully sized for alignment, and efficient fusion requires subtleties such as redundantly computing values on the overlapping boundaries of intermediate tiles. The resulting implementation is 22 times faster on a quad-core CPU, but together these optimizations have fused two simple, independent steps into a single intertwined, architecture-specific mess (see Figure 1.b).

We believe the right answer is to separate the intrinsic algorithm—*what* is computed—from the concerns of efficiently mapping to machine execution—decisions about *storage* and the *ordering of computation*. We call these choices of how to map an algorithm onto resources in space and time the *schedule*.

Functional programming provides a natural basis for separating the *what* from the *when* and *where*. Divorced from explicit storage, images are no longer arrays filled by procedures, but are instead pure functions that define the value at each point in terms of arithmetic, reductions, and the application of other functions (see Figure 1.c). This functional representation also omits boundaries, and the order and extent of iteration, making images functions over an infinite integer domain.

In this representation, the algorithm only defines the value of each function at each point, and the schedule specifies:

1. the order in which points in a function are evaluated, including tiling, the exploitation of parallelism, and mapping onto SIMD execution units;
2. the granularity with which the evaluation of points in one function are interleaved with evaluating points in the functions which call it;
3. the memory locations into which the values of a function are stored, including registers, scratchpad memories, and regions of main memory;
4. whether a value is recomputed, or from where it is loaded, at each place a function is used.

The key challenge in doing this is defining a representation of schedules which is both simple and expressive. Halide’s model (Section 3) decomposes the organization of a pipeline into four major choices for each function, corresponding to the points above, each described as a composition of simple primitives.

Figure 1. The C++ code at the top (a) computes a 3×3 box filter using the composition of a 1×3 and a 3×1 box filter. Using vectorization, multithreading, tiling, and fusion, we can make this algorithm more than 20 times faster on a quad-core x86 CPU (b) However, in doing so we have lost simplicity and portability. Halide (c) separates the algorithm description from its schedule, describing the same optimizations, generating very similar machine code, and achieving the same performance without making these sacrifices. (Benchmarked on an Intel Core i7–4790, from the `BLUR` app in the Halide repository.¹⁴)

```
(a) Clean C++: 6.5ms per megapixel
void blur(const Image<uint16_t> &in, Image<uint16_t> &bv) {
    Image<uint16_t> bh(in.width(), in.height());

    for (int y = 0; y < in.height(); y++)
        for (int x = 0; x < in.width(); x++)
            bh(x, y) = (in(x-1, y) + in(x, y) + in(x+1, y))/3;

    for (int y = 0; y < in.height(); y++)
        for (int x = 0; x < in.width(); x++)
            bv(x, y) = (bh(x, y-1) + bh(x, y) + bh(x, y+1))/3;
}

(b) Fast C++ (for x86): 0.30ms per megapixel
void fast_blur(const Image<uint16_t> &in, Image<uint16_t> &bv) {
    __m128i one_third = _mm_set1_epil6(21846);
    #pragma omp parallel for
    for (int yTile = 0; yTile < in.height(); yTile += 32) {
        __m128i a, b, c, sum, avg;
        __m128i bh[(256/8)*(32+2)];
        for (int xTile = 0; xTile < in.width(); xTile += 256) {
            __m128i *bhPtr = bh;
            for (int y = -1; y < 32+1; y++) {
                const uint16_t *inPtr = &(in(xTile, yTile+y));
                for (int x = 0; x < 256; x += 8) {
                    a = _mm_loadu_si128((__m128i*)(inPtr - 1));
                    b = _mm_loadu_si128((__m128i*)(inPtr + 1));
                    c = _mm_load_si128((__m128i*)(inPtr));
                    sum = _mm_add_epil6(_mm_add_epil6(a, b), c);
                    avg = _mm_mulhi_epil6(sum, one_third);
                    _mm_store_si128(bhPtr++, avg);
                    inPtr += 8;
                }
            }
            bhPtr = bh;
            for (int y = 0; y < 32; y++) {
                __m128i *outPtr = (__m128i *)(&(bv(xTile, yTile+y)));
                for (int x = 0; x < 256; x += 8) {
                    a = _mm_load_si128(bhPtr + (256 * 2) / 8);
                    b = _mm_load_si128(bhPtr + 256 / 8);
                    c = _mm_load_si128(bhPtr++);
                    sum = _mm_add_epil6(_mm_add_epil6(a, b), c);
                    avg = _mm_mulhi_epil6(sum, one_third);
                    _mm_store_si128(outPtr++, avg);
                }
            }
        }
    }
}

(c) Halide: 0.29ms per megapixel
Func halide_blur(Func in) {
    Func bh, bv;
    Var x, y, xi, yi;

    // The algorithm
    bh(x, y) = (in(x-1, y) + in(x, y) + in(x+1, y))/3;
    bv(x, y) = (bh(x, y-1) + bh(x, y) + bh(x, y+1))/3;

    // The schedule
    bv.tile(x, y, xi, yi, 256, 32)
        .vectorize(xi, 8).parallel(y);
    bh.compute_at(bv, x).vectorize(x, 8);

    return bv;
}
```

Halide can most flexibly schedule operations which are data parallel with statically analyzable access patterns (such as stencils), but also supports the bounded iterative

algorithms and irregular access patterns that occur in image processing and general array computation. It imposes a few restrictions on the range of expressible schedules, but is sufficient to concisely express implementations of many image processing algorithms, with state-of-the-art performance on architectures ranging from mobile and server CPUs, to GPUs, to specialized image processors.

Once the programmer has specified an algorithm and a schedule, the Halide compiler combines them into an efficient implementation. Optimizing the execution strategy for a given architecture requires modifying the schedule, but not the algorithm. The representation of the schedule is compact and does not affect the correctness of the algorithm (e.g. Figure 1.c), so exploring the performance of many options is fast and easy. It can be written separately from the algorithm, by an architecture expert if necessary, and we have also shown that good schedules can often be found automatically.^{17,23}

In the rest of this article we will briefly introduce Halide’s language for algorithms (Section 2), discuss its model of schedules and the organizational choices they represent (Section 3), touch on the design of the Halide compiler (Section 4), demonstrate results on several real computational photography algorithms, (Section 5), and conclude by discussing connections with the wealth of related work (Section 6), and our perspective after five years of development and widespread use (Section 7).

2. THE HALIDE ALGORITHM LANGUAGE

Halide describes image processing pipelines in a simple functional style. A straightforward C++ implementation of an algorithm such as local Laplacian filters is described by dozens of loop nests and hundreds of lines of code.² This is not practical to globally optimize with traditional loop optimization systems.¹⁰ The Halide version distills this into 62 lines describing just the essential dataflow and computation in the 99 stage pipeline, and all choices for how the program should be synthesized are described in a separate *schedule* (Section 3).

Halide represents images as pure functions defined over an infinite integer domain, where the value of a function at a point represents the value of the image at the corresponding coordinate. Halide functions can have arbitrary dimensionality (not just two), and may be tuple-valued (they can store a “struct” of values at each point, not just a single number). Pipelines are specified as a directed acyclic graph of functions. The expressions that define functions are side-effect free, and are much like those in any simple functional language, including arithmetic and logical operations, if-then-else expressions, loads from memory buffers, and calls to other functions (including external C ABI functions).

For example, a separable 3×3 unnormalized box filter is expressed as a chain of two functions in x, y :

```
Func bh, bv; Var x, y;
ImageParam in(UInt(8), 2);

bh(x, y) = in(x-1, y) + in(x, y) + in(x+1, y);
bv(x, y) = bh(x, y-1) + bh(x, y) + bh(x, y+1);
```

This representation is simpler than most functional languages. It does not include higher-order functions, dynamic

recursion, or additional data structures such as lists. Functions simply map from integer coordinates to a scalar or tuple-valued result.

Halide is embedded in C++. It uses simple type and operator overloading (not template metaprogramming) to lazily construct programs, rather than eagerly executing expressions as they are written. This “staged” nature makes the Halide front-end easily extensible. Many advanced constructs are expressible by using C++ as a meta-programming layer for Halide. For example, you can simulate higher-order functions by writing a C++ function that takes and returns Halide functions. This provides a powerful tool for structuring code and it does not change the underlying representation of a pipeline.

This representation is sufficient to describe a wide range of image processing algorithms, and these constraints enable flexible analysis and transformation of algorithms during compilation. Critically, this representation is naturally data parallel within the domain of each function. Also, since functions are defined over an infinite domain, boundary conditions can be handled safely and efficiently in two ways. For intermediate pipeline stages, an implementation can compute arbitrary *guard bands* of extra values. For input images, or stages for which specific boundary conditions matter to the meaning of an algorithm, the function may define its own. The compiler will partition the resulting loops so that the boundary conditions have minimal impact on performance.

Update definitions

Functions are typically defined by simple expressions in their arguments, but may additionally have a sequence of bounded *updates* to accommodate reductions (e.g., large-support convolution), scatters (e.g., histograms), and recursive scans (e.g., Infinite Impulse Response (IIR) filters). Sequential iteration within an update can be introduced with an `RDom` (“reduction domain,” a multidimensional iteration domain). The value of the function at each point in the output domain is defined by the final value after all updates are applied. The key constraint relative to arbitrary loops is that the bound of an `RDom` cannot depend on the values computed inside its updates. This guarantees that all iteration bounds are decidable (and the language is therefore not Turing complete).

This pattern can describe a range of algorithms outside the scope of traditional stencil computation but essential to image processing pipelines, in a way that encapsulates side effects. To the rest of the pipeline, a function with updates still acts as stateless pure function that can be evaluated over an arbitrary domain. For example, histogram equalization combines multiple reductions and a data-dependent gather. A scattering reduction computes a histogram, a recursive scan integrates it into a Cumulative Distribution Function (CDF), and a simple point-wise operation remaps the input using the CDF:

```
Func histogram, cdf, out; Var x, y, i;
ImageParam in(UInt(8), 2);

RDom r(0, in.width(),
        0, in.height());
histogram(i) = 0; // initial value
histogram(in(r.x, r.y)) += 1; // update
```

```

RDom ri(0, 255);
cdf(i) = 0; // initial value
cdf(ri) = cdf(ri-1) + histogram(ri); // update

out(x, y) = cdf(in(x, y));

```

3. SCHEDULING IMAGE PROCESSING PIPELINES

A complete Halide algorithm is a DAG of functions over regular grids. Actually evaluating an algorithm requires iterating over and computing all of the required points in each function. But in what order should we compute these points? And where should we store and load the results of intermediate stages to communicate them between stages? Unlike a looping program in a traditional language, the Halide *algorithm* does not specify these choices. Instead, they're specified by a separate *schedule*.

For each stage, we think about these choices in four parts:

1. In what order should we iterate over the points in that stage?
2. In what order should we lay out those points in memory to store their results?
3. At what granularity should we interleave the computation of that stage with the computation of the downstream stages which consume its results?
4. At what granularity should we store blocks of the function for reuse across iterations of its consumers?

These choices affect performance through locality (e.g., cache behavior), exposed parallelism, and the need for recomputation (at grain boundaries). Together, they specify a set of nested loops (invisible to the programmer) that our compiler uses to generate executable code. Scheduling instructions are concise and can be written independently of the definition of the algorithms, as shown in Figure 1(c).

3.1 Scheduling within a function

The order of evaluation of the points within a function is defined by a family of common transformations applied to a default (sequential, row-major) loop nest over the grid of points in the function's domain. Loop dimensions can be split, merged, and reordered. Because the regions computed are simple intervals (axis-aligned bounding boxes within the grid), the result is always a perfect loop nest. In addition, the resulting loops can be unrolled, or mapped to parallel threads, SIMD vector lanes, and GPU kernel dimensions.

The dimensions of the storage layout for a function's results can similarly be reordered, allowing common transformations such as column- versus row-major layouts. Storage can also be *folded* across a dimension into a circular buffer of a fixed size.

The dimensions of an RDom in an update step may also be reordered, parallelized, etc., but only if the compiler can prove that different update iterations across those dimensions do not interact. Recent work has added support for splitting associative reductions to create parallel reduction trees.²⁵

3.2 Scheduling across functions

The more unique part of Halide's model of schedules is how it transforms computation and storage *across* functions.

Consider the simple two-stage blur algorithm. When scheduling across these two stages, we call the first stage, *bh*, the "producer," and the second stage, *bv*, its "consumer." So far, the scheduled order of computation *within* each function (discussed above) defines a perfect loop nest for each function. For example, the default schedule for the output stage *bv* gives a simple row-major loop nest equivalent to:

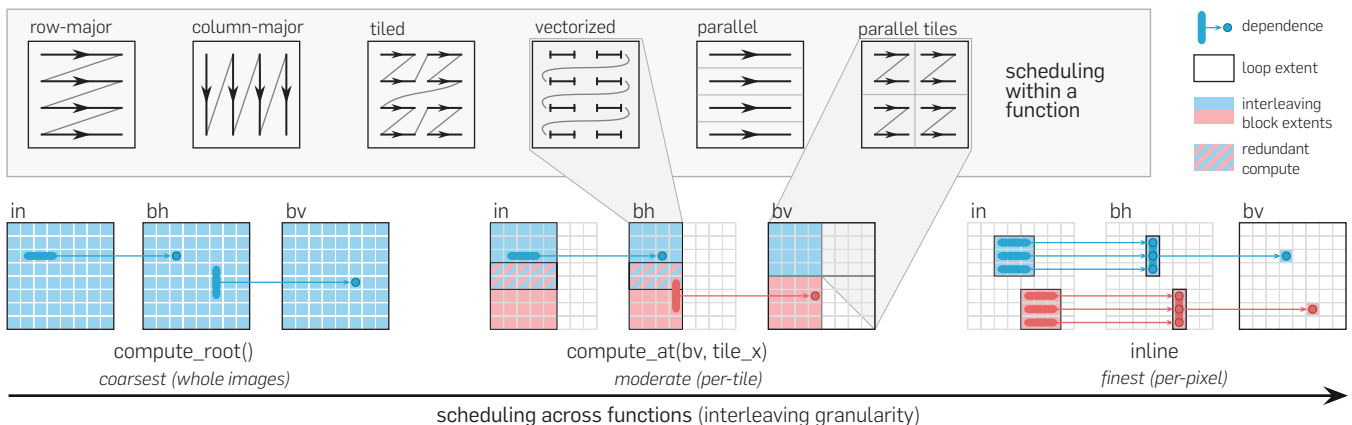
```

for bv.y in bv.min.y to bv.max.y:
  for bv.x in bv.min.x to bv.max.x:
    compute and store bv(bv.x, bv.y)

```

The schedule *across* functions specifies at what level in this

Figure 2. A Halide schedule specifies the order of computation *within* each stage in an image processing algorithm (top, discussed in Sec. 3.1), and the granularity of interleaving *across* producer and consumer stages (bottom, discussed in Sec. 3.2). In the two-stage blur algorithm, coarse-grained interleaving (bottom-left) computes whole stages at a time, sacrificing producer-consumer locality. Finer-grained interleaving improves locality but introduces redundant computation where the stencils for the grains overlap (denoted by hatched regions in the bottom-middle). In the extreme case of per-pixel interleaving (bottom-right), values are immediately consumed as soon as they are computed (maximizing locality), but all values of the producer stages are redundantly recomputed in each place they are reused (here, 3 times in *bh* and 9 times transitively in *in*). Whereas this shows a single shared order and granularity choice made applied throughout a simple pipeline, all of these choices can be made separately for each function in a whole algorithm.



loop nest around the *consumer* stage the program should evaluate the required part of the *producer* stage (here, *bh*). This determines the granularity with which the computation of the different functions is interleaved.

For example, Halide’s default schedule for producers is *inline*, which means that if we leave the default schedule for *bh*, it is evaluated directly everywhere it is called:

```
for bv.y in bv.min.y to bv.max.y:
  for bv.x in bv.min.x to bv.max.x:
    bv(bv.x, bv.y) = // bh(x,y-1)+bh(x,y)+bh(x,y+1) =
      (in(bv.x-1,bv.y-1)+in(bv.x,bv.y-1)+in(bv.x+1,bv.y-1))
      +(in(bv.x-1,bv.y) +in(bv.x,bv.y) +in(bv.x+1,bv.y))
      +(in(bv.x-1,bv.y+1)+in(bv.x,bv.y+1)+in(bv.x+1,bv.y+1))
```

This maximizes locality between the producer and consumer, since each value of *bh* is immediately used as soon as it is computed, so it does not need to be stored and loaded in far-away memory. However, because the stencils of neighboring pixels overlap, every pixel in *bh* is computed three times instead of once: when the outer loop moves to the next row down, each pixel of *bv* recomputes two of the same pixels of *bh* that were just computed by the pixel right above it.

The other obvious choice we could make is to not interleave these two stages at all. Instead, we compute all of *bh* before computing any of *bv*. We express this in Halide by saying that *bh* should be computed at the very outermost, or “*root*” level, outside all loops over its consumer. This is done by calling *bh.compute_root()*, generating the complete equivalent program:

```
// compute bh over slightly enlarged window,
// to fulfill all uses in bv
for bh.y in bv.min.y-1 to bv.max.y+1:
  for bh.x in bv.min.x to bv.max.x:
    bh(bh.x, bh.y) = in(bh.x-1, bh.y)
      + in(bh.x, bh.y)
      + in(bh.x+1, bh.y)

// compute bv using bh
for bv.y in bv.min.y to bv.max.y:
  for bv.x in bv.min.x to bv.max.x:
    bv(bv.x, bv.y) = bh(bv.x, bv.y-1)
      + bh(bv.x, bv.y)
      + bh(bv.x, bv.y+1)
```

This only computes each pixel in each stage exactly once, wasting no work, but it destroys any producer-consumer locality between the two stages: an entire image of intermediate results has to be computed between where values are produced in the first stage and where they are consumed in the second. This schedule is equivalent to the clean C++ as shown in Figure 1(a), which suffers from the same problem.

Looking at these two examples, we see a tension between locality and (excess, or redundant) computation. This tension is fundamental to computations with stencil dependencies, since stencils overlap between iterations and prevent simple loop fusion without duplicating work. However, this tension can be balanced along a continuum between these extremes.

For example, we can change the schedule within *bv* to iterate in a tiled order by splitting its two dimensions by a chosen tile size, and then reordering the four resulting dimensions. We can do this by saying:

```
bv.split(x, tx, xi, tile_size)
  .split(y, ty, yi, tile_size)
  .reorder(xi, yi, tx, ty)
```

which can be abbreviated simply as:

```
bv.tile(x, y, tx, ty, xi, yi, tile_size, tile_size)
```

This generates the loop nest:

```
// simplified to ignore non-evenly divisible tile sizes
for bv.ty in bv.min.y/tile_size to bv.max.y/tile_size:
  for bv.tx in bv.min.x/tile_size to bv.max.x/tile_size:
    for bv.yi in 0 to tile_size-1:
      for bv.xi in 0 to tile_size-1:
        compute bv(bv.tx*tile_size + bv.xi,
          bv.ty*tile_size + bv.yi)
```

Now, by computing *bh* at the granularity of *tiles* of *bv*—at the *bv.tx* loop level—we get a less fine-grained interleaving than individual pixels, but one more fine-grained than the whole image. In Halide’s scheduling language, we specify this by saying *bh.compute_at(bv, tx)*. This is the key scheduling choice in the optimized implementations as shown in Figure 1(b) and (c). This generates the loop nest:

```
for bv.ty in bv.min.y/tile_size to bv.max.y/tile_size:
  for bv.tx in bv.min.x/tile_size to bv.max.x/tile_size:
    // compute tile of bh
    for bh.y in (bv.ty*tile_size-1) to ((bv.ty+1)*tile_size):
      for bh.x in (bv.tx*tile_size) to ((bv.tx+1)*tile_size-1):
        bh(bh.x, bh.y) = in(bh.x-1, bh.y)
          + in(bh.x, bh.y)
          + in(bh.x+1, bh.y)

    // compute tile of bv
    for bv.yi in 0 to tile_size-1:
      for bv.xi in 0 to tile_size-1:
        bv.y = bv.ty*tile_size + bv.yi
        bv.x = bv.tx*tile_size + bv.xi
        bv(bv.x, bv.y) = bh(bv.x, bv.y-1)
          + bh(bv.x, bv.y)
          + bh(bv.x, bv.y+1)
```

This organization recomputes only one row on the top and bottom of each tile, rather than every row, while improving locality to the level of tiles rather than whole images. By increasing or decreasing the size of these tiles, we can easily trade off between locality and computation. Common choices would be sized to fit tiles in a given level of cache. The ideal choice depends on the relative cost of recomputing a pixel of the producer function versus storing and loading it from each level of the memory hierarchy.

It is important to realize that this is not just a constant-factor tweak in the resulting code: if the tile size is a constant, we have *asymptotically* improved the locality relative to the naïve interleaving, by reducing the reuse distance for intermediate values from $O(n)$ to $O(1)$.

We can also choose to compute *bh* at any of the other loop levels in *bv*: the schedule of computation across stages in Halide is specified, for each producer function, as any single level in the scheduled loop nest of the downstream pipeline.

The fourth and final core part of the schedule similarly specifies the granularity of storage across functions. The granularity of storage determines at what loop level in the evaluation of the consuming pipeline the results of a producer function should be stored for reuse. On architectures such as GPUs, different levels are also mapped to

different parts of the specialized memory hierarchy. Using this and other scheduling primitives, Halide can express additional locality-optimizing patterns of computation, such as line-buffering or software pipelining, but these are beyond the scope of this paper. More complete discussion can be found in prior publications.^{21, 23}

In the DAG of functions that makes up a program, we have local ordering choices for each node, and producer-consumer granularity choices for each edge. In the real applications presented in Section 5, the optimized schedules make different but inter-dependent choices for each of dozens of stages and producer-consumer dependencies.

4. THE COMPILER

The Halide compiler takes a graph of Halide functions and *lowers* it to imperative code for a variety of CPU, GPU, and Digital Signal Processor (DSP) architectures. This process is directed by the schedule. It then converts this imperative code to LLVM's intermediate representation, and generates machine code from there.

In contrast to a traditional compiler, lowering makes very few heuristic decisions. Instead, it is designed to be highly deterministic and controllable. Any time there is a choice to be made that might affect performance we defer to the schedule.

Lowering has two essential responsibilities. The first is synthesizing the loop nest specified by the schedule, including any vectorization, unrolling, multi-core parallelism, prefetching, memoization of stages, and offloading work to GPU or DSP accelerators. The second responsibility of lowering is inferring the regions of each function that must be computed to satisfy all producer-consumer dependencies.

Lowering additionally implements optional passes for tracing, profiling, and similar instrumentation, and passes that apply opportunistic optimizations. These include partitioning loops in order to ameliorate the effect of boundary conditions, tightening loop bounds when it can be proved that the loop body does nothing for some of its domain, and

deriving runtime conditions under which the computation of stages can be skipped entirely.

For a full description of the compiler see our original PLDI paper,²¹ and the source code itself.¹⁴

5. RESULTS & EVALUATION

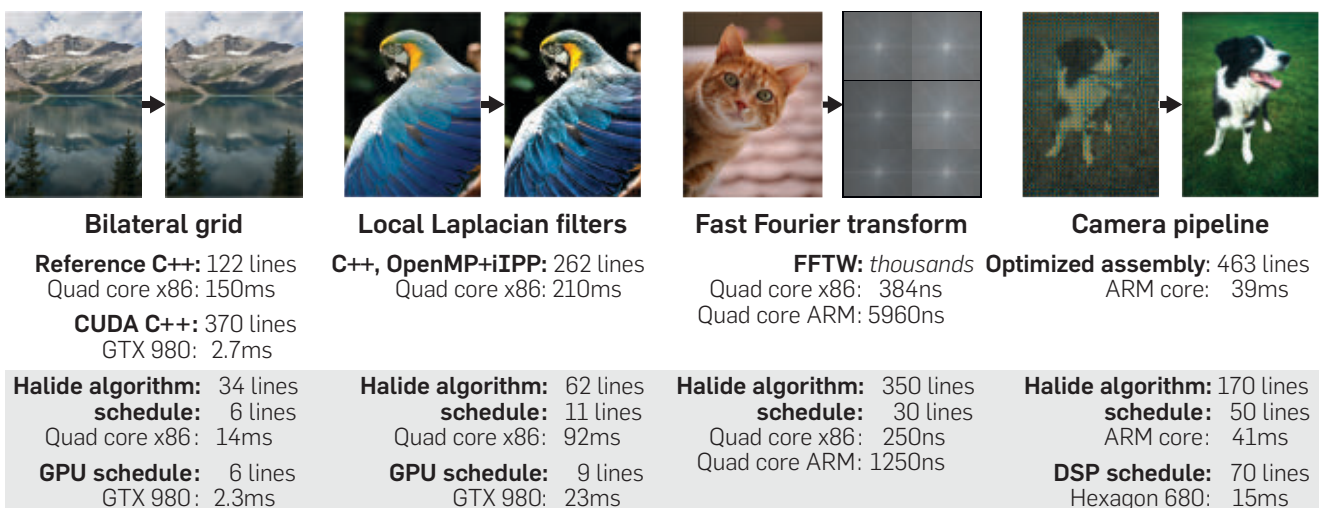
Halide has been open source since it was released by MIT in 2012, Halide source repository.¹⁴ The largest user of and contributor to Halide is Google. The largest Halide pipeline we are aware of is Google's HDR+¹⁵ which produces every photograph taken on their flagship Pixel phones. Halide pipelines are also run at scale within Google Photos, YouTube, and elsewhere. Outside of Google, Halide code is shipping in products from Instagram, Facebook, Adobe, and others.

In the original Halide publications,^{17, 22, 23} we evaluated the Halide representation and compiler by implementing a variety of modern image processing pipelines in Halide and comparing the result to the best previously-published expert implementations we could find. The pipelines exhibit a wide variety of data types, access patterns, and pipeline lengths. Here, we highlight four representative pipelines that approximately span this space (Figure 3). These pipelines also approximately correspond to the major algorithms in Google's HDR+ pipeline, albeit at a more modest scale. Our more recent work and open source code also includes competitive matrix-matrix multiply (GEMM) and convolutional neural network implementations in very little code.^{14, 17} We generally find that Halide code is many times faster than equivalently simple reference code, many times more concise than equivalently fast code, or some tradeoff between those two extremes.

Benchmark applications

Bilateral grid is an efficient implementation of the bilateral filter, which smoothes an image while preserving its main edges.⁷ It first scatters the image data into a 3D grid, effectively building a windowed histogram in each column of the grid, then blurs the grid along each of its axes with three

Figure 3. Summary of the code size and running time Halide implementations compared to handwritten reference implementations of four representative computational photography algorithms. The time measurements are reported in milliseconds to process the representative image (4–6 megapixels), except for the fast Fourier transform. The FFT experiment is for the tiled real-to-complex transform, and reports time in nanoseconds per 16 × 16 tile.



5-point stencils. Finally, the output image is constructed by trilinear interpolation within the grid at locations determined by the input image.

The CPU reference code is a tuned but clean implementation from the original authors in 122 lines of C++. It is partially autovectorized by GCC, but is nontrivial to multithread (a naive OpenMP parallelization of major stages results in a slowdown on our benchmark CPU), so the reference is single-threaded. The Halide algorithm is 34 lines, and compiles to an implementation 11 times faster than the original. The speedup comes from a combination of parallelism, tilelevel fusion of some stages, and careful reordering of dimensions to control parallel grain size in the grid.

We also compared the Halide version to a hand-tuned GPU implementation from the original authors, written in 370 lines of CUDA code. The same Halide algorithm, paired with a different schedule, was 17% faster than the hand-written CUDA, but about $\frac{1}{10}$ th the total code size. The Halide compiler generates similar GPU code to the reference, but with Halide we quickly found a new point in the schedule space. It sacrifices some parallelism in the grid construction step to reduce synchronization overhead, and uses a tiled fusion strategy which passes intermediate results through GPU scratchpad memory to improve locality through the blur steps at the expense of redundant computation. These tradeoffs were counter-intuitive to the original author, and much harder to express in CUDA, but are easily described by our schedule representation.

Local Laplacian filters uses a multi-scale approach to tone map images and enhance local contrast in an edge-respecting fashion.² It is used in the clarity, tone mapping, and other filters in Adobe Photoshop and Lightroom. The algorithm builds and manipulates several image pyramids. The filter output is produced by a data-dependent resampling from the processed pyramids. With the parameters we used, the pipeline contains 99 different stages, operating at many scales, and with several different computational patterns.

The reference implementation is 262 lines of C++, developed at Adobe. It is carefully parallelized with OpenMP, and offloads most intensive kernels to tuned assembly routines from Intel Performance Primitives (IPP). It has very similar performance to a version deployed in their products, which took several months to develop, including 2–3 weeks dedicated to optimization. It is 10 times faster than an algorithmically identical reference version written by the authors in pure C++, without IPP or OpenMP.

The Halide version was written in one day, in 52 lines of code. It compiles to an implementation which is 2.3 times faster than the highly optimized expert implementation (at least 20 times faster than the clean C++ without IPP and OpenMP). The resulting schedule is complex, mixing different fusion, tiling, vectorization, and multithreading strategies throughout the 99 stage graph. In C, it would correspond to hundreds of loops over thousands of lines of code, but in Halide it is just 11 lines.

The same program compiles with a different schedule to a hybrid CPU/GPU program with 23 unique GPU kernels, each representing a different subset of the overall graph. It runs 9.1 times faster than the hand-tuned Adobe implementation, and is four times faster than the best parallel and vectorized implementation on the CPU.

Fast Fourier transform implements small 2D FFTs, based on an algorithm adapted for GPUs.¹³ Small 2D tiled FFTs are used in many image processing workloads. The strategy in this implementation is to perform Cooley-Tukey FFTs on columns, which is convenient for vectorizing along the rows. The data is transposed repeatedly such that all of the FFTs are performed on columns; after the FFT is complete, the data is transposed back to the original orientation. Real FFTs are implemented by computing pairs of columns of FFTs simultaneously.

We compare the performance of our FFT implementation to FFTW.¹¹ For 16×16 , 32×32 , and 48×48 complex-to-complex FFTs, the Halide FFT is about 1.3 times faster than FFTW. For similarly sized real FFTs, Halide is between 1.5 times and 2.1 times faster than FFTW on x86, and up to 4.6 times faster than FFTW on ARM. For larger FFTs, the Halide FFT performance begins to drop relative to FFTW, becoming comparable to FFTW at 64×64 and dropping further from there, as we have not implemented a vectorization strategy more suitable for these larger FFTs.

In addition to the raw performance, the Halide FFT has a number of other advantages over FFT libraries. Because the FFT is expressed in pure Halide, the operations being performed in the Fourier domain can be fused into the FFT itself, improving locality. The implementation is also more easily modified, for example to compute fixed point or zero-padded FFTs. Because the FFT is expressed at a high level, with the optimizations expressed in the schedule, it can deliver state-of-the-art performance across a variety of platforms.

Camera pipeline transforms the raw data recorded by a camera sensor into a photograph. It performs four tasks: hot pixel suppression, demosaicking, color correction, and global tone mapping (i.e., gamma correction and contrast enhancement). These steps contain a variety of operations including stencils, color space transformations, and table lookups. The demosaicking alone is a combination of 21 inter-dependent stencils.

The reference comparison is a single carefully tiled and fused loop nest from the Frankencamera, expressed in 463 lines of C++ with ARM NEON intrinsics and inline assembly.¹ All producer-consumer communication is staged through scratch buffers. The Halide implementation is 170 lines describing 32 functions and 22 different stencils, literally translated from the pseudocode in the comments explaining the original source. With 70 lines of combined schedule code, the Halide algorithm can be compiled to target 32- and 64-bit x86 and ARM, is trivially parallelized across CPU cores, and can also be compiled to run on a Qualcomm Hexagon DSP.

We benchmarked this workload on a Google Pixel smartphone, which contains a Hexagon 680 DSP with HVX (Hexagon Vector Extensions), a standard feature of the popular Snapdragon 820 SoC. The Halide CPU code performs almost identically to the ARM reference implementations on a single core. Using all four CPU cores, the Halide implementation is 2.7 times faster than the single-core reference implementation. (The CPU cores on this processor are asymmetric, with a theoretical linear speedup of 3.5 times across all four.) The Halide Hexagon implementation performs similar to the four core CPU implementation, which is in line with the theoretical throughput of two HVX processing clusters. However, while we expect that to be the case, it delivers that

throughput using up to 10 times less power.

Porting the camera pipeline to Hexagon required modifying the schedule to use a sliding window on the rows of the image, instead of tiling. Hexagon has extremely large vectors, and also benefits heavily from loop pipelining, which means that efficient 2D tile sizes are too large to fit in cache. Halide allowed us to easily rewrite the camera pipeline from a tiled schedule to a sliding window schedule, without changing the algorithm description. In contrast, rewriting the handwritten ARM implementation to use a sliding window would be a serious undertaking, as the tiling logic is inextricably linked to the algorithm description, all the way into the indexing arithmetic in the assembly inner loops.

Scheduling new algorithms

To better understand how long it takes an expert Halide developer to schedule an algorithm when starting from scratch, we recruited two professional Halide developers (authors on this paper) into a scheduling “race.” They selected three new, non-trivial applications they had never scheduled before (LENSBLUR, NLMEANS, and MAXFILTER) and implemented the original Halide algorithm for these programs. For each benchmark, each programmer independently developed a schedule in a single programming session. The programmer stopped optimizing after converging on a solution they considered their reasonable best. While developing the schedules the developers documented their progress by measuring the performance of their current schedule at various points in the session.

In each case, the developers started out with relatively simple baseline schedules, but which generally still included at least SIMD vectorization and multicore parallelism. Nonetheless, the speedup from this baseline to their best optimized performance was always many times.

Results of the race are as shown in Figure 4. The X-axis in each of the graphs indicates development time (in minutes) for the schedules. The Y-axis shows the performance of the benchmark (measured as pixel throughput, so higher is better). The yellow and gray lines each correspond to one of the programmers.

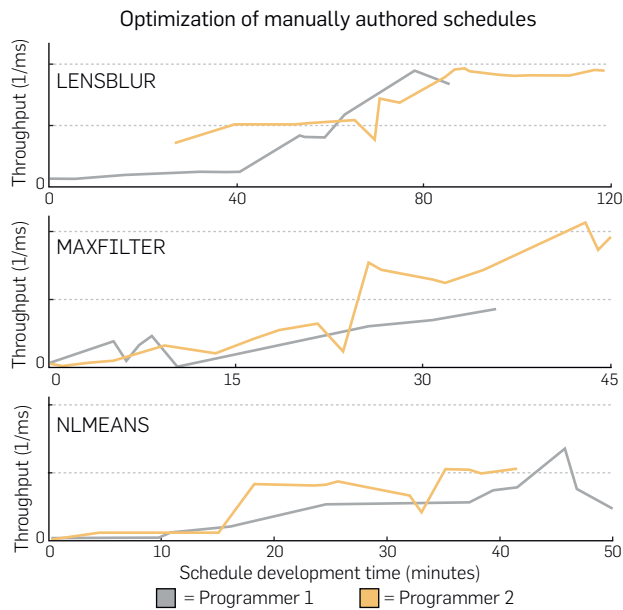
Arriving at a good schedule requires significant optimization effort, even for experts. For these applications, it took on the order of an hour of iteration and experimentation for the developers to feel their optimization had reasonably converged. The resulting schedules, however, are only a few lines of code, and dramatically simpler than equivalent code in explicit C++ loops.

Automatically determining good schedules

To further accelerate development of high-performance code, and to make Halide accessible to programmers without optimization experience, we have also worked on ways to *automatically* optimize Halide programs.

In the original version of this paper we demonstrated a prototype *autotuner* which automatically searched the (exponentially large) space of schedules for a program using stochastic search with a genetic algorithm.²³ For each generated schedule, the autotuner benchmarked its actual runtime by compiling and running a complete test program given by the user. In a few hours to days of tuning, it was able to generate schedules competitive with hand-tuned results

Figure 4. Two professional Halide developers were tasked with developing schedules for three new programs. The graphs plot the runtime of their schedules, over time as they developed them. The developers converged to what they considered “reasonable” performance in on the order of an hour per program.



for several applications, including some presented above.

To converge in reasonable time on real programs, we found that higher-level *heuristics* are critical to guide the search, because of the high cost of compiling and running each new test. Drawing on this experience, we have since found that using a simple cost model in place of recompiling and benchmarking each configuration, and restricting the search to greedily consider variants of common interleaving patterns, gave a much faster and more scalable automatic scheduling method.¹⁷ The cost model is based directly on the tradeoff we observe between parallelism, locality, and redundant work. The tool is also easier to use since it directly analyzes the algorithm, rather than requiring a runnable benchmark program and data set.

6. RELATED WORK

Graphics & image processing languages

Domain-specific languages for image processing go back at least as far as Bell Labs’ Pico and POPI.¹⁶ Most prior image processing languages have focused on efficient expression of individual kernels. Pan introduced a functional model for image processing much like our own, in which images are functions from coordinates to values.⁸ Recently, others have demonstrated automatic optimization of image processing code in a subset of the Halide’s algorithm model using the polyhedral model.¹⁸

Elsewhere in graphics, the real-time graphics pipeline has been a hugely successful abstraction precisely because the *schedule* is separated from the specification of the shaders.⁴ This allows GPUs and drivers to efficiently execute a wide range of programs with little programmer control over parallelism and memory management. This separation of concerns is extremely effective, but it is specific to the design of a single pipeline.

Loop transformation

Most compiler optimizations for numerical programs are based on loop analysis and transformation, including auto-vectorization, loop interchange, fusion, and tiling.³ The polyhedral model is a powerful tool for modeling and transforming looping imperative programs.¹⁰ Halide's model considers only axis-aligned bounding regions, not general polytopes—a practical simplification for image processing and many other applications. Interval analysis is simpler than modern polyhedral analysis, but it can effectively analyze through a wide range of expressions, and it is trivial to generate high quality loop nests from intervals. This simple generality is essential for Halide's design in which all bounds are inferred from context. Most traditional loop optimizations also do not consider *recomputation* of values, but in image processing this can be a large performance win and is central to the choices we consider during optimization.

Parallel languages

Many data-parallel languages have been proposed. Chapel and its predecessors focus on computations over regions of regular grids, much like Halide, with explicit primitives to control distribution of grid regions across memory and processors.⁶ Popular today, CUDA and OpenCL expose an imperative, single program-multiple data programming model which can target both GPUs and multicore CPUs with SIMD units.^{5,19} Like C, these languages allow the specification of very high performance implementations for many algorithms, but because their semantics closely model the underlying machine, they also deeply conflate algorithms with optimization.

Streaming languages encode data and task parallelism in graphs of kernels, which compilers automatically schedule using tiling, fusion, fission, and 1D stencil optimization.¹² Halide's model of scheduling addresses the problem of multidimensional stencils with parallelism, where recomputation versus storage becomes a critical but complex choice.

A separate line of research creates explicit languages for choices of how problems are mapped into physical execution, much like Halide's decoupling of schedules from algorithms. SPIRAL uses a domain-specific language to specify linear signal processing operations at the level of mathematical operators,²⁰ and a separate algebra of rewrite rules describes ways these operations can be turned into efficient code for a particular architecture. Sequoia defines a model where a user-defined “mapping file” describes how to execute tasks on a tree-like memory hierarchy.⁹ The CHILL framework exposes polyhedral program transformations and code generation choices via a dedicated scripting language.²⁴

7. CONCLUSION AND PERSPECTIVE

Our initial hope in designing Halide was to allow expert-level manual control over the organization of computation in image processing code, which was at once far more controllable and powerful than traditional compiler flags and auto-vectorization, and far more productive than hand coding in low-level languages. After more than 4 years in production, we found this works even better than we first hoped. Across a range of applications and target architectures, we find that Halide's scheduling representation is able to model, and its


compiler is able to generate, implementations which deliver state-of-the-art performance from surprisingly little code.

This performance comes from careful navigation of the high dimensional space of tradeoffs between locality, parallelism, and redundant recomputation in image processing pipelines. Expressing these tradeoffs in traditional languages is challenging enough, as shown by the much greater complexity of hand-written implementations, but finding the ideal balance is daunting when each change a programmer might want to try can require completely rewriting a complex loop nest hundreds of lines long. The performance advantage of the Halide implementations is a direct result of simply *making it easy to test many more points in the space* than a human programmer could manually describe using explicit loops. We have found this design to generalize well from its initial CPU and GPU targets to new, more specialized (and notoriously hard-to-program) image processing accelerators and DSPs. We think it helps point the way towards the “OpenGL” of future specialized image processing hardware.

Still, while Halide's model of schedules is powerful and productive in the hands of experts, we have found it challenging for novices and those unfamiliar with high performance programming to master. Even for experts, optimized schedules grow difficult to maintain for algorithms beyond moderately sized blocks, and there's no complete answer for what should be done if those blocks are meant to be reused in different places or get moved around a larger pipeline (as with a library of reusable components). Fundamentally, schedules describe how to interleave computation across composition boundaries. Decoupling them from the fundamental algorithm gives simpler, more modular algorithm code, but modularizing scheduling choices without sacrificing performance remains an open problem. This led to our attempts to automate the process of scheduling in Halide,^{17,23} which have shown promise but remain ongoing work. An automatic scheduling system could both remove the burden from novice programmers, and schedule globally across composition boundaries even in large applications.

Finally, we have also found that, while the constraints and style of the Halide language were motivated by examples in modern image processing, the programming model is significantly more general: it expresses arbitrary bounded computations over multidimensional regular grids, and it has been applied to algorithms in linear algebra, scientific simulation, and machine learning. Really, Halide should be thought of as specific to the “data structure” of regular grids or multidimensional arrays, not to the “domain” of image processing.

Acknowledgments

As an open source project, Halide has received contributions from many people. Most notably, Zalman Stern, Steven Johnson, and Patricia Suriana are full-time developers on the project at Google and are responsible for a large amount of the current code. The Hexagon backend was developed with Pranav Bhandarkar, Ron Lieberman, Dan Palermo, and Anshuman Dasgupta at the Qualcomm Innovation Center. Eric Chan provided feedback and inspiration throughout the original design of Halide. This work was supported by DOE Award DE-SC0005288, NSF grant 0964004, grants from Intel and Quanta, and gifts from Cognex and Adobe. 

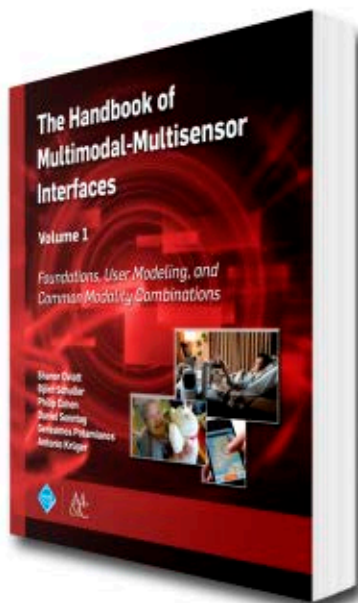
References

- Adams, A., Talvala, E., Park, S.H., Jacobs, D.E., Ajdin, B., Gelfand, N., Dolson, J., Vaquero, D., Baek, J., Tico, M., Lensch, H.P.A., Matusik, W., Pulli, K., Horowitz, M., Levoy, M. The Frankencamera: An experimental platform for computational photography. *ACM Trans. Graph.* 29, 4 (2010), 29:1–29:12.
- Aubry, M., Paris, S., Hasinoff, S.W., Kautz, J., Durand, F. Fast local Laplacian filters: Theory and applications. *ACM Trans. Graph.* 33, 5 (2014), 167.
- Bacon, D.F., Graham, S.L., Sharp, O.J. Compiler transformations for high-performance computing. *ACM Comput. Surv.* 26, 4 (Dec. 1994).
- Blythe, D. The Direct3D 10 system. *ACM Trans. Graph.* 25, (2006), 724–734.
- Buck, I. GPU computing: Programming a massively parallel processor. In *Proceedings of the International Symposium on Code Generation and Optimization* (Tessellations Publishing, Phoenix, Arizona, 2007).
- Chamberlain, B., Callahan, D., Zima, H. Parallel programmability and the Chapel language. *Int. J. High Perform. Comput. Appl.* 21, (2007), 291–312.
- Chen, J., Paris, S., Durand, F. Real-time edge-aware image processing with the bilateral grid. *ACM Trans. Graph.* 26, 3 (2007), 103:1–103:9.
- Elliott, C. Functional image synthesis. In *Proceedings of Bridges 2001, Mathematical Connections in Art, Music, and Science* (IEEE Computer Society, Washington, DC, USA, 2001).
- Fatahalian, K., Horn, D.R., Knight, T.J., Leem, L., Houston, M., Park, J.Y., Erez, M., Ren, M., Aiken, A., Dally, W.J., Hanrahan, P. Sequoia: Programming the memory hierarchy. In *ACM/IEEE conference on Supercomputing* (ACM, New York, NY, 2006).
- Feautrier, P. Dataflow analysis of array and scalar references. *Int. J. Parallel Program.* 20, 1 (1991), 23–53.
- Frigo, M., Johnson, S.G. The design and implementation of FFTW3. *Proc. IEEE* 93, 2 (2005).
- Gordon, M.I., Thies, W., Karczmarek, M., Lin, J., Meli, A.S., Leger, C., Lamb, A.A., Wong, J., Hoffman, H., Maze, D.Z., Amarasinghe, S. A stream compiler for communication-exposed architectures. In *International Conference on Architectural Support for Programming Languages and Operating Systems* (ACM, New York, NY, 2002).
- Govindaraju, N., Lloyd, B., Dotsenko, Y., Smith, B., Manferdelli, J. High performance discrete Fourier transforms on graphics processors. In *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*. IEEE (Washington, DC, January 2008).
- Halide source repository, <http://github.com/halide/Halide>.
- Hasinoff, S.W., Sharlet, D., Geiss, R., Adams, A., Barron, J.T., Kainz, F., Chen, J., Levoy, M. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Trans. Graph.* 35, 6 (2016).
- Holzmann, G. *Beyond Photography: The Digital Darkroom*. Prentice Hall, Englewood Cliffs, NJ, 1988.
- Mullapudi, R.T., Adams, A., Sharlet, D., Ragan-Kelley, J., Fatahalian, K. Automatically scheduling halide image processing pipelines. *ACM Trans. Graph.* 35, 4 (2016).
- Mullapudi, R.T., Vasista, V., Bondhugula, U. PolyMage: Automatic optimization for image processing pipelines. In *ACM SIGPLAN Notices* (ACM, New York, NY, 2015), volume 50, 429–443.
- The OpenCL specification, version 1.2. <http://www.khronos.org/registry/cl/specs/opencl-1.2.pdf>, 2011.
- Püschel, M., Moura, J.M.F., Johnson, J., Padua, D., Veloso, M., Singer, B., Xiong, J., Franchetti, F., Gacic, A., Voronenko, Y., Chen, K., Johnson, R.W., Rizzolo, N. SPIRAL: Code generation for DSP transforms. *Proceedings of the IEEE, special issue on "Program Generation, Optimization, and Adaptation"* 93, 2 (2005), 232–275.
- Ragan-Kelley, J. Decoupling algorithms from the organization of computation for high performance image processing. PhD thesis, Massachusetts Institute of Technology (2014).
- Ragan-Kelley, J., Adams, A., Paris, S., Levoy, M., Amarasinghe, S., Durand, F. Decoupling algorithms from schedules for easy optimization of image processing pipelines. *ACM Trans. Graph.* 31, 4 (2012).
- Ragan-Kelley, J., Barnes, C., Adams, A., Paris, S., Durand, F., Amarasinghe, S. Halide: A language and compiler for optimizing parallelism, locality, and recomputation in image processing pipelines. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation* (ACM, New York, NY, 2013).
- Rudy, G., Khan, M.M., Hall, M., Chen, C., Chame, J. A programming language interface to describe transformations and code generation. In *Proceedings of the 23rd International Conference on Languages and Compilers for Parallel Computing LCPC'10*, (Springer-Verlag, Berlin, Heidelberg, 2011), 136–150.
- Suriana, P., Adams, A., Kamil, S. Parallel associative reductions in halide. In *Proceedings of the 2017 International Symposium on Code Generation and Optimization* (ACM, New York, NY, 2017).

Andrew Adams, and Dillon Sharlet ([abadams, dsharlet]@google.com), Google.
Saman Amarasinghe, and Frédo Durand ([saman, fredod]@csail.mit.edu), MIT CSAIL.
Connelly Barnes (connelly@cs.virginia.edu), University of Virginia.

Marc Levoy (levoy@google.com), Stanford University & Google.
Sylvain Paris (sparis@adobe.com), Adobe.
Jonathan Ragan-Kelley (jrk@eecs.berkeley.edu), UC Berkeley.

© 2018 ACM 0001-0782/18/1 \$15.00



The FIRST authoritative resource.

EDITED BY

Sharon Oviatt, *Incaa Designs*

Björn Schuller, *University of Passau, Imperial College London*

Philip Cohen, *VoiceBox Technologies*

Daniel Sonntag, *German Research Center for Artificial Intelligence*

Gerasimos Potamianos, *University of Thessaly*

Antonio Krüger, *German Research Center for Artificial Intelligence*



ISBN: 978-1-970001-64-8 DOI: 10.1145/3015783

<http://books.acm.org>

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

CAREERS

Boston College

Assistant Professor of the Practice or Lecturer

The Computer Science Department of Boston College aims to grow substantially over the next several years, and will seek to fill faculty positions at all levels. We invite applications for a full-time, non-tenure-track faculty position, beginning in the 2018-2019 academic year. Candidates should be committed to excellence in undergraduate education, and should be able to both teach a broad variety of undergraduate computer science courses, and to participate in the development of new courses that reflect the evolving landscape of the discipline.

Minimum requirements for the title of Assistant Professor of the Practice include a Ph.D. in Computer Science or closely related discipline. Candidates who have only attained a Master's degree would be eligible for the title of Lecturer.

Application review begins October 1, 2017. See www.cs.bc.edu for more information.

To apply go to

<http://apply.interfolio.com/44984>.

Boston College

Associate or Full Professor of Computer Science

The Computer Science Department of Boston College aims to grow substantially over the next several years, and will seek to fill faculty positions at all levels. We invite applications for a senior-level (Associate or Full Professor) position, starting in the 2018-2019 academic year. The successful candidate is expected to play a leadership role in the creation of a Data Science program in conjunction with the new interdisciplinary Institute for Integrated Science and Society, recently announced at Boston College, and will also participate in shaping the future of our growing department.

Applicants must have a Ph.D. in Computer Science or closely related discipline, a record of strong research accomplishment and external funding, and a commitment to quality in undergraduate and graduate education. Preference will be given to candidates whose research is in the areas of high-performance data mining / machine learning or data visualization, particularly those with a history of interdisciplinary collaboration, but outstanding candidates in all research areas will be considered.

Application review begins October 1, 2017. See www.cs.bc.edu for more information.

To apply go to

<http://apply.interfolio.com/44982>

Boston College

Tenure Track Assistant Professor in Computer Science

The Computer Science Department of Boston College aims to grow substantially over the next several years and will seek to fill faculty positions

at all levels. We invite applications for one or more tenure-track faculty positions at the rank of Assistant Professor, beginning in the 2018-2019 academic year. Successful candidates will be expected to develop strong research programs that can attract external research funding. The search will focus on candidates who can participate in cross-disciplinary research in conjunction with the new Institute for Integrated Science and Society recently announced at Boston College, in the areas of high-performance data mining / machine learning, systems / networks, data visualization, and human-computer interaction. However, outstanding candidates in all research areas will be considered.

Minimum requirements for all positions include a Ph.D. in Computer Science or closely related discipline, an energetic research program that promises to attract external funding, and a commitment to quality in undergraduate and graduate education.

Application review begins October 1, 2017. See www.cs.bc.edu for more information.

To apply go to

<http://apply.interfolio.com/44980>

California State University San Bernardino

Tenure Track Position - Assistant Professor

The School of Computer Science and Engineering at California State University San Bernardino invites applications for a tenure track position at the Assistant Professor level beginning in Fall 2018. Candidates must have a Ph.D. in Computer Science or a closely related field. All areas of computer science will be considered. The position is to support primarily the B.S. in Computer Science (ABET accredited), the BA in Computer Systems and the M.S. in Computer Science programs. In addition, the school offers the degrees B.S. in Computer Engineering (ABET accredited) and B.S. in Bioinformatics.

The candidate must display potential for excellence in teaching and scholarly work. The candidate is expected to supervise student research at both the undergraduate and graduate levels, and to actively participate in other types of academic student advising. The candidate will actively contribute to the School's curriculum development. The candidate will serve the School, College and University, as well as the community and the profession.

Women and underrepresented minorities are strongly encouraged to apply. For more information about the School of Computer Science and Engineering, please visit <http://cns.csusb.edu/cse>.

DEADLINE AND APPLICATION PROCESS:

Applicants should submit a curriculum vitae, statement of teaching philosophy, description of research interests, copies of transcripts of all post-secondary degrees (official transcripts will be required prior to appointment), contact information for three references at

<https://www.governmentjobs.com/careers/csusb/jobs/1918429/computer-science-engineering-assistant-professor-tenure-track?pagetype=transferJobs>.

Please have three letters of recommendation sent on your behalf to facultyrecruitment@csusb.edu.

Review of applications will begin February 1, 2018, and will continue until the position is filled.

Questions about this position can be directed to Dr. Haiyan Qiao, Director of School of Computer Science and Engineering, at hqiao@csusb.edu

Case Western Reserve University

Tenure-Track Faculty Positions in Computer Science

The Department of Electrical Engineering and Computer Science at Case Western Reserve University invites applications for tenure-track positions in Computer Science. Preference will be given to candidates at the assistant professor level, but applications at other ranks will be considered commensurate with experience and qualifications. While exceptional candidates in all areas of Computer Science will be considered, our priority areas include Database, Data Mining, Data Science and Analytics, and Computer systems (Networks, Cyber-Security, Distributed Computing). Current departmental strengths in Computer Science include the Internet of Things, Bioinformatics, Software Engineering, Networks and Distributed Systems, and Artificial Intelligence and Machine Learning, and we expect a successful candidate to be synergistic with these strengths.

Applicants should have potential for excellence in innovative research. All successful candidates are expected to develop a vibrant, high-quality externally sponsored research program, supervise graduate students, and interact and collaborate with faculty across the department and campus. Applicants should have a strong commitment to high quality teaching at the undergraduate and graduate levels. Candidates must have a Ph.D. in Computer Science or a closely related field.

Applicants must submit (i) a cover letter, (ii) current curriculum vita, (iii) statement of research interests, and (iv) statement of teaching interests and (v) recommend at least three references. Applications will be reviewed starting immediately and will continue to be reviewed until the position is filled.

Application materials may be sent by email to YoLonda Stiggers at yxs307@case.edu or via mail to: Computer Science Faculty Search Committee Dept. of Electrical Engineering and

Computer Science
Case Western Reserve University
c/o YoLonda Stiggers
10900 Euclid Avenue, Glennan 321
Cleveland, OH 44106-7071

Founded in 1826, Case Western Reserve University is a highly-ranked private research uni-

versity located in Cleveland, Ohio. As a vibrant and up-and-coming city, Cleveland was named one of the top 15 best places to live in the US by timeout.com in 2016. The campus is in the heart of University Circle, a world-renowned area for its cultural vibrancy, hosting the Cleveland Museum of Art (the second highest ranked art museum in the country), Cleveland Orchestra, the Museum of Natural History, Cleveland Institute of Music, and the Cleveland Botanical Garden, as well as two world-class health institutions, The Cleveland Clinic and University Hospitals of Cleveland.

In employment, as in education, Case Western Reserve University is committed to Equal Opportunity and Diversity. Women, veterans, members of underrepresented minority groups, and individuals with disabilities are encouraged to apply.

George Mason University Department of Computer Science Assistant Professor Tenure-Track Faculty Positions

The Department of Computer Science in the Volgenau School of Engineering at George Mason University, invites applications for tenure-track assistant professor positions beginning fall 2018. Exceptionally strong senior candidates may also be considered, and must have an established record of outstanding research and excellent teaching. Such candidates will be eligible for tenured associate professor or professor positions. The university has an institutional commitment to achieving excellence and diversity among its faculty and staff and encourages candidates who will enrich its academic and culturally inclusive environment to apply.

Responsibilities:

Successful candidates will be expected to teach at the undergraduate and graduate levels; develop an independent, externally funded research program; advise students; participate in all aspects of the department's mission; and serve the profession.

Required Qualifications:

Applicants must have received a PhD in Computer Science or a related field by the start date of the position and should have demonstrated potential for excellence and productivity in research and a commitment to high-quality teaching.

Preferred Qualifications:

While applicants in all areas of Computer Science will be given serious consideration, we are particularly interested in candidates in the areas of Computer Graphics, Human Computer Interaction, Machine Learning, Artificial Intelligence, Robotics, and Databases.

About Us:

The Computer Science Department has over 40 faculty members with wide-ranging research interests including artificial intelligence, algorithms, autonomic computing, computational biology, computer graphics, computer vision, databases, data mining, parallel and distributed systems, real-time systems, robotics, security, software engineering, and wireless and mobile computing. The department has more than \$8 million in annual research funding and 11 recipients of the National Science Foundation's prestigious CAREER award.

For more information on the department, visit our website: <http://cs.gmu.edu/>.

For full consideration applicants must apply for position number F196AZ at <http://jobs.gmu.edu>; complete and submit the online application; attach a complete C.V. with publications, statement of teaching interests, statement of research interests, and the names of three professional references. The review of applications will begin January 15, 2018 and continue until the position is filled.

George Mason University is an equal opportunity/affirmative action employer, committed to promoting inclusion and equity in its community. All qualified applicants will receive consideration for employment without regard to race, color, religion, sex, gender identity, sexual orientation, national origin, disability, or protected veteran status.

The Harvard John A. Paulson School of Engineering and Applied Sciences Tenure Track Faculty Position in Computer Science

The Harvard John A. Paulson School of Engineering and Applied Sciences (SEAS) seeks applicants for a position at the tenure-track level in Computer Science, with an expected start date of July 1, 2018.

We are accepting applications in all areas of Computer Science, especially machine learning, programming languages, and human-computer interaction.

We seek candidates who have a strong research record and a commitment to undergradu-

ate and graduate teaching and training. We particularly encourage applications from historically underrepresented groups, including women and minorities.

Computer Science at Harvard benefits from outstanding undergraduate and graduate students, world-leading faculty, an excellent location, significant industrial collaboration, and substantial support from the Harvard Paulson School. Information about Harvard's current faculty, research, and educational programs in computer science is available at <http://www.seas.harvard.edu/computer-science>.

The associated Institute for Applied Computational Science (<http://iacs.seas.harvard.edu>) and Data Science Initiative (<https://datascience.harvard.edu/>) foster connections among computer science, applied math, data science, and various domain sciences at Harvard through its graduate programs and events.

A doctorate or terminal degree in Computer Science or a related field is required by the expected start date.

Required application documents include a cover letter, CV, a statement of research interests, a teaching statement, and up to three representative papers. Candidates are also required to submit the names and contact information for at least three and up to five references, and the application is complete only when three letters have been submitted. We encourage candidates to apply by December 1, 2017, but will continue to review applications until the position is filled. Applicants can apply online at <http://academicpositions.harvard.edu/postings/7928>



香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

The Chinese University of Hong Kong, Shenzhen

The Hopcroft Institute for Advanced Study in Information Sciences

The Hopcroft Institute for Advanced Study in Information Sciences was established on June 2017 in the School of Science and Engineering at The Chinese University of Hong Kong, Shenzhen. Prof. John E. Hopcroft, the Turing Award Winner in 1986, serves as the Director of the Institute. This institute is committed to excellence and innovation in research and education in all areas of information sciences and technology, including computer science, computer engineering and information engineering. It is positioned to 1) carry out world class researches, 2) train outstanding young talents, and 3) educate next generation technical leaders in all areas of information sciences.

Post Specifications:

The Institute invites applications for multiple faculty positions at both senior and junior levels in all areas of Computer Science and Information Science. Junior applicants should have (i) a PhD degree (by the time of reporting for duty) in related fields (CS, EE, IE, Data Science); and (ii) high potential in teaching and research. Candidates for senior post (Associate and Full Professor) are expected to have demonstrated academic leadership and strong commitment to excellence in teaching, research, and services. Junior appointments will normally be made on contract basis for up to three years initially, leading to longer-term appointment or tenure later subject to review. Exceptional appointments with tenure will be considered for candidates of proven excellence. Applicants are encouraged to check out the details about the university at <http://www.cuhk.edu.cn>

Salary and Fringe Benefits

Salary will be comparable to international standards, commensurate with experience and accomplishments. Appointments will be made under the establishment of CUHK(SZ), and employee benefits will be provided according to the relevant labor laws of Mainland China as well as CUHK(SZ) regulations. The university will also sponsor eligible faculty members to apply for various government talent programs: <http://www.cuhk.edu.cn/UploadFiles/talentsprogramoutline.pdf>

Application package, including CV and contacts of three referees, as well as personal statements in teaching, research, and service, should be emailed to: Talents4SSE@cuhk.edu.cn. Applicants are required to specify the rank of the position in their letter of application. Applicants also need to ask three referees to send the letters directly to Talents4SSE@cuhk.edu.cn upon submitting application materials.

We are an equal opportunity employer and all qualified applicants will receive consideration for employment without regard to race, color, religion, sex, sexual orientation, gender identity, national origin, disability status, protected veteran status, or any other characteristic protected by law.

Hua Loo-Keng Center for Mathematics Sciences

Post-Doctorate Fellow Positions

Hua Loo-Keng Center for Mathematics Sciences (HCMS), Academy of Mathematics and Systems Science (AMSS), Chinese Academy of Science invites outstanding young researchers in mathematical sciences to apply for 2 post-doctorate fellow positions in 2018. The research areas include but not limited to the following: (algebra) computational theory, symbolic and symbolic-numerical computation, quantum algorithms and cryptography. Successful applicants are expected to conduct research full-time in the above mentioned areas.

The position is for two years. The amount of annual salary plus fringe benefits is 300K RMB (fringe benefits are about 20% for Chinese citizens and 5% for non-Chinese citizens). Other welfare benefits will be in accordance with the rules of AMSS.

HCMS is supported by the project of National Natural Science Foundation of China, "Geometry, analysis, and computation on manifolds". HCMS has 45 faculty members, including 5 Fellows of Chinese Academy of Sciences and 18 recipients of the National Science Fund for Distinguished Young

Scholars from NSFC. The main purpose of HCMS is to carry out research of highest level on some of the main directions of Mathematical Sciences through cooperation and exchange. HCMS runs regular academic programs throughout the whole year.

According to relevant rules, the applicant's age should not exceed 35 years, and the time of obtaining a doctorate should not exceed three years. To apply, please send a message to: msec@amss.ac.cn with the following information:

- ▶ a detailed resume;
- ▶ a research statement;
- ▶ a copy of diploma certificate if available;
- ▶ two letters of recommendation (including one from your PhD thesis supervisor).

Applicants are encouraged to submit their applications before Feb. 28, 2018. The positions will open until filled.

The Max Planck Center for Visual Computing and Communication Junior Research Group Program

The Max Planck Center for Visual Computing and Communication (MPC-VCC) was established as a joint program by the Max Planck Society for the Advancement of Science (MPG) and Stanford University in 2003 (<http://www.mpc-vcc.org>).

With this call the Max Planck Center for Visual Computing and Communication (MPC-VCC) invites applications for its **Junior Research Group Program**.

Our Junior Research Group program offers young scientists in information technology the opportunity to develop their own research program addressing important problems in areas such as

- ▶ image communication
- ▶ computer graphics
- ▶ geometric computing
- ▶ imaging systems
- ▶ computer vision
- ▶ human machine interface
- ▶ distributed multimedia architectures
- ▶ multimedia networking
- ▶ visual media security

The center includes an outstanding group of faculty members at Stanford's Computer Science and Electrical Engineering Departments, the Max Planck Institute for Informatics, and Saarland University.

The program begins with a preparatory 1-2 year postdoc phase (**Phase P**) at the Max Planck Institute for Informatics, followed by a two-year appointment at Stanford University (**Phase I**) as a visiting assistant professor, and then a position at the Max Planck Institute for Informatics as a junior research group leader (**Phase II**). However, the program can be entered flexibly at each phase, commensurate with the experience of the applicant.

Applicants to the program must have completed an outstanding PhD. Exact duration of the preparatory postdoc phase is flexible, but we typically expect this to be about 1-2 years. Applicants who completed their PhD in Germany may enter Phase I of the program directly. Applicants for Phase II are expected to have completed a post-



CEMSE Computer, Electrical and Mathematical Sciences and Engineering

The Computer, Electrical, and Mathematical Sciences and Engineering Division at King Abdullah University of Science and Technology (KAUST) invites applications for a faculty position in Data Science at all levels (Assistant, Associate, and Full Professor) beginning in the Fall of 2018.

KAUST is an international, graduate research university, located in Saudi Arabia on the shores of the Red Sea.

Applicants should apply here <https://stat.kaust.edu.sa/employment>
Inquiries on the position should be sent to datasci@kaust.edu.sa

Applications received by January 31, 2018, are guaranteed consideration.

Faculty
Position
in Data Science 2018

doc stay abroad and must have demonstrated their outstanding research potential and ability to successfully lead a research group.

Reviewing of applications will commence on 01 Jan 2018. The final deadline is **31 Jan 2018**. Applicants should submit their CV, copies of their school and university reports, list of publications, reprints of five selected publications, names of 3-5 references, a brief description of their previous research and a detailed description of the proposed research project (including possible opportunities for collaboration with existing research groups at Saarbrücken and Stanford) to:

Prof. Dr. Hans-Peter Seidel
Max Planck Institute for Informatics,
Campus E 1 4, 66123 Saarbrücken, Germany;
Email: mpc-vc@mpi-inf.mpg.de

The Max Planck Center is an equal opportunity employer and women are encouraged to apply.

Additional information is available on the website www.mpc-vc.org.

North Dakota State University Tenure-Track Assistant/Associate Professor Positions

The Department of Computer Science seeks to fill two tenure-track Assistant/Associate Professor positions, one in Computational Biology and the other in Cyber Security and Big Data starting Fall 2018. See https://www.ndsu.edu/cs/policies_and_information/cs_department_positions/ for more information about the department and positions.



ADVERTISING IN CAREER OPPORTUNITIES

How to Submit a Classified Line Ad: Send an e-mail to acmm mediasales@acm.org. Please include text, and indicate the issue/or issues where the ad will appear, and a contact name and number.

Estimates: An insertion order will then be e-mailed back to you. The ad will be typeset according to ACM guidelines. NO PROOFS can be sent. Classified line ads are NOT commissionable.

Deadlines: 20th of the month/2 months prior to issue date. For latest deadline info, please contact: acmm mediasales@acm.org

Career Opportunities Online: Classified and recruitment display ads receive a free duplicate listing on our website at: <http://jobs.acm.org>

Ads are listed for a period of 30 days.

**For More Information Contact:
ACM Media Sales
at 212-626-0686 or
acmm mediasales@acm.org**

NDSU offers degrees at all levels in Computer Science and Software Engineering. Research and teaching excellence is expected. Normal teaching loads are three courses per year. Salary is competitive.

The Computer Science Department has 13 faculty, 7 lecturers and approximately 150 graduate (Ph.D. and M.S.) students, and nearly 500 B.S./B.A. students.

Assistant/Associate Professor – Computational Biology

Ph.D. in Computer Science, Computational Biology, Bioinformatics, Health Informatics, or a closely related field. See <https://jobs.ndsu.edu/postings/8780> for more information and to apply.

Assistant/Associate Professor – Cyber Security

Ph.D. in Computer Science or Software Engineering with focus on Cybersecurity, Big Data, Data Mining or a closely related topic. See <https://jobs.ndsu.edu/postings/8777> for more information and to apply.

Minimum requirements for both positions are: Experience or other evidence of potential for excellence in undergraduate and graduate teaching, research and service. Effective oral and written communication skills are necessary. Experience or other evidence of potential to conduct research in ecosystems, sustainability or health applied to rural populations is desirable.

NDSU is a top national research university. Fargo is a clean, growing metropolitan area of 250,000 that consistently ranks near the top in national quality-of-life surveys. We have low levels of crime and pollution, excellent schools, short commutes, and proximity to the Minnesota lake country. The community has a symphony, an opera, a domed stadium, a community theater, three colleges, a research technology park and many other amenities. NDSU is an equal opportunity institution.

Review of applications begins January 15, 2018. Applications will be accepted until positions are filled.

NDSU is an EEO/AA-MF/Vet/Disability.

University of Maine Tenure-Track Faculty Position in Computer Science

The School of Computing and Information Science seeks applicants for a tenure-track Assistant Professor position with research foci in the broad areas of Secure Computing and/or Software Engineering.

While strong candidates in all related areas will be considered, expertise in cybersecurity is desired, either in a sub-area (e.g., trustworthy computing, development and architecture of secure software systems, formal methods, and software systems verification) or as it pertains to another area (e.g., networks, Internet of Things, software engineering, operating systems, trust/verification of AI, and deep learning systems). Regardless of the candidate's research area, teaching courses in cybersecurity is required. We are particularly interested in candidates that complement the School's existing research strengths in Artificial Intelligence, Data Management, Distributed Systems, Human Computer Interaction, and Spatial Informatics with potential for collaborations within the School. The ability to contribute to campus-wide Signature and Emerging areas of excellence in Data Science and STEM Education Research (with focus on CS) also would

be viewed favorably. Candidates should have a strong research profile such as demonstrated by relevant and recent contributions in top ranked conferences and journals, presentations at significant conferences, awards, and similar evidence. Successful teaching experience is also desired. A PhD in computer science or a closely related discipline is required by date of hire, expected to be 9/1/2018.

The successful candidate will be expected to establish a dynamic research program in his or her fields of expertise, to become an engaging teacher, adviser, and mentor at both the undergraduate and graduate levels, and to make a strong commitment to curricula development. The typical teaching load is three courses per year, including both undergraduate and graduate courses in the School's academic programs. Service to the School, College, and University is expected.

Increasing diversity of the computing profession is one of our strategic priorities, and women and traditionally underrepresented people in computing fields are particularly encouraged to apply.

To apply, visit <https://bit.ly/UMaineCSPosition> (full position announcement). Also arrange for three letters of recommendation to be sent to Roy Turner, Chair, CS Faculty Search Committee, University of Maine, 5711 Boardman Hall, Orono, ME 04469-5711 or rturner@maine.edu. Incomplete applications cannot be considered. Review of applications will begin 1/15/2018 and continue until the position is filled.

The University of Maine, an EO/AA employer, seeks to employ outstanding people who contribute to the rich cultural diversity expected in a university setting. All qualified individuals are encouraged to apply.

The University of Texas at San Antonio Faculty Position in Computer Science

The Department of Computer Science at The University of Texas at San Antonio (UTSA) invites applications for two tenured/tenure-track positions, starting in Fall 2018. The first position is for a tenure-track Assistant or tenured/tenure-track Associate Professor in Game-related areas. The focus is on Computer Graphics, especially 3D animation, 3D modeling, and real-time rendering; and/or Human Computer Interaction, especially human computer interfaces, virtual reality, augmented reality, and game analytics. The second position is for a tenured/tenure-track Associate Professor in Data Science and Artificial Intelligence, focusing on cybersecurity, Internet of things, bioinformatics, natural language processing, speech recognition, language understanding, computer vision, or machine learning. This position is part of UTSA's focused cluster hiring plan under the Gold Star Initiative to recruit top-tier researchers over a four-year period.

See <http://www.cs.utsa.edu/fsearch> for information on the Department and application instructions. Screening of applications will begin immediately. The search will continue until the positions are filled or the search is closed.

The University of Texas at San Antonio is an Affirmative Action/Equal Opportunity Employer.

Department of Computer Science

RE: Faculty Search

The University of Texas at San Antonio

One UTSA Circle

San Antonio, TX 78249-0667

Phone: 210-458-4436



DOI:10.1145/3157090

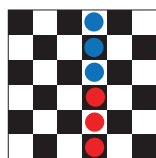
Dennis Shasha

Upstart Puzzles

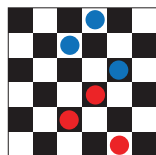
Polychromatic Choreography

A CERTAIN MODERN dance choreographer has her dancers wear k different-colored leotards. For example, when k is 2, half the dancers wear red and the other half wear blue. In general, there are k colors with n dancers wearing each color. The basic algorithmic problem she has to solve is how to instruct her dancers to move from some given configuration to a configuration in which the dancers form disjoint vertical or horizontal line segments, with each line segment consisting of one dancer from each of the k colors in any order—a “perfect lineup.” A movement of one dancer consists of a horizontal or vertical step.

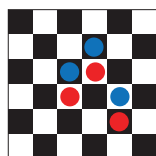
Warm-Up. Consider the following configuration of six dancers on a grid, where three wear blue leotards and three wear red leotards. Can you achieve a perfect lineup in just two moves?



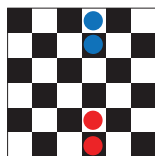
Solution to Warm-Up.



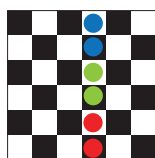
and, moving vertically



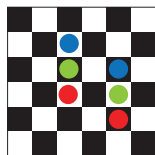
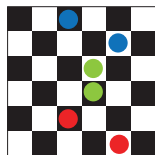
Challenge. Starting with two red and two blue dancers, now add two green dancers. We want to create two disjoint segments, each with a red, blue, and green dancer in any order, constituting the perfect lineup in this case. Note the blues and reds are two spaces apart. Where should the two greens start in order to create a perfect lineup in two steps? Show those two steps.



Solution.



Here are the two steps



Upstart 1. Given an initial configuration of k colors, each with an equal number n of dancers of each color on a grid, design an algorithm that uses as few steps as possible to achieve a perfect lineup.

Upstart 2. Given k colors and n dancers of each color and a board of size $B \times B$,



Consider k groups of dancers, where the dancers in each group wear leotards of the same color. The choreographer’s goal is to get them to move on the grid in parallel, producing a set of disjoint line segments, each including a dancer of each color.

find a maximally hard configuration of the dancers; a configuration c is maximally hard if c requires m parallel steps to achieve a perfect lineup and no other configuration requires more than m steps to achieve a perfect lineup.

Upstart 3. Given a maximally hard configuration with cost m , is there any way to add a $k+1^{\text{st}}$ color of n dancers to reduce the number of steps required to achieve a perfect lineup?

Upstart 4. How would these upstarts change if diagonal (and counter-diagonal) segments were allowed?

All are invited to submit their solutions to upstartpuzzles@cacm.acm.org; solutions to upstarts and discussion will be posted at <http://cs.nyu.edu/cs/faculty/shasha/papers/cacmpuzzles.html>

Dennis Shasha (dennisshasha@yahoo.com) is a professor of computer science in the Computer Science Department of the Courant Institute at New York University, New York, as well as the chronicler of his good friend the omniheurist Dr. Ecco.

Copyright held by the author.

PHOTO BY GURVANDY ANDREY

ACM Code of Ethics: A Guide for Positive Action

THE ACM CODE of Ethics and Professional Conduct (the Code) is being updated by the Code Update Task Force^a in conjunction with the ACM's Committee on Professional Ethics. The Code was initially written in 1992, and this is the first update since then. In previous articles we detailed the motivations for updating the Code,^b gave our responses to feedback on the initial draft, and produced an updated version, which we presented for feedback through the ACM Discourse site, email, and focus groups and workshops at ETHICOMP and SIGCSE. We thank everyone who took part in this public consultation round. Their insights, both positive and negative, were invaluable. We have deliberated extensively on the numerous suggestions for additions, changes, and deletions. Based on those deliberations, we produced Draft 3 of the Code.

There are some significant changes made in Draft 3. Some principles have been removed entirely or completely rewritten, and some new principles were added in response to recommendations by several respondents. This article explains the significant changes that were made, and a few changes that were suggested but not made. For the most part, the suggestions that were not explicitly incorporated are ideas that we consider covered by existing aspects of the Code. Some of these suggestions were excellent, and because of them, explanatory materials that will supplement the Code are being designed. These include examples, cases, and more detailed explanations of the Code.

This article is part of the final round of public consultation associated with the 2018 update to the ACM Code of Ethics. ACM members agree to abide by the Code. Please encourage other computing professionals around you to

read and contribute to this effort.

We have provided two opportunities to comment on this draft and suggest ways it might be improved. We have provided a space for open discussion of Draft 3 among interested parties at the ACM Code 2018 Discussion website <https://code2018.acm.org/discuss>. In addition, ACM members are encouraged to take an online survey about the specific principles of the Code at <https://www.acm.org/code-2018-survey>. Both comment systems close **Feb. 10, 2018**.

The ACM is a professional society whose goals include promotion of the highest standards “to advance the profession and make a positive impact.” Thus, the ACM Code of Ethics and Professional Conduct ought to reflect the conscience of the computing profession, understood in the broadest sense. A successful code of ethics should reflect the values of the computing profession in a way that can help ACM members make appropriate ethical decisions. The Code should also inspire members, future members, and other professionals by highlighting the aspirations of the profession.

The ACM Code of Ethics and Professional Conduct is a guide to proactive action that helps us, as a profession, promote good. Because of this, it also applies to those aspiring to be computing professionals, including students. Members of ACM student chapters are also invited to take the survey and comment on the Code.

In the Code we identify global ethical principles that reflect the highest standards of computing professionals. The Code is designed to inform ACM members and others of what society should expect from computing professionals, and what computer professionals should expect of themselves.

In the next section we identify specific changes we made in response to suggestions from the reviewers. The section after that addresses some of the thoughtful suggestions that did not directly lead to changes. The article

concludes with Draft 3 of the Code for your review.^c

I. Changes from Draft 2 to Draft 3

1.2 Harm

There were several comments about Principle 1.2: Avoid Harm. Some respondents suggested that this principle was inconsistent with work in the military sector or law enforcement where some systems are designed, in part, to cause harm. The Task Force modified the guidance for this Principle to deal more effectively with that perception. Another concern expressed was that with most systems harm of some degree almost always happens. In response, we sharpened the definitions of intentional and unintentional harms, and we added language to encourage professionals to take care to minimize unintended harm.

1.3 Transparency and Honesty

Commenters were concerned that certain technological developments such as algorithmic transparency and systems that learn were not addressed by the Code. We agreed. However, since one goal of this update process is to craft language that will apply to new technologies as they emerge, we did not include these specific technologies explicitly in the Code. Instead, we tried to use language that would implicitly include them and future developments. Certain aspects of algorithmic transparency are covered by the principles regarding nondiscrimination and privacy, but the concept of transparency was not addressed in Draft 2, except with respect to the actions of people. By changing the guidance for Principle 1.3 on honesty to include explicit discussion of transparency, especially with respect to system limitations

^a For a list of current taskforce members see <http://ethics.acm.org/code-2018>.

^b <http://cacm.acm.org/magazines/2016/12/210366-the-acm-code-of-ethics/fulltext>

^c A complete track changes version of Draft 3 showing additions and deletions to draft 2 is available at <http://ethics.acm.org/code-2018>.

and problems, the Code now addresses technologies that are opaque even to their developers. Suggestions about how to manage the release of self-modifying systems are also now made in the guidance for Principle 2.5.

1.4 Harassment

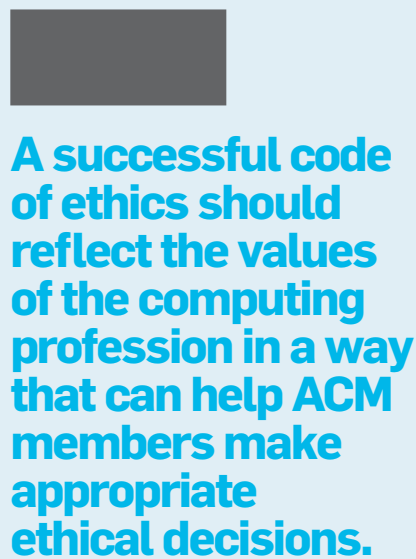
The harassment principle generated much discussion, both for and against a new emphasis in Draft 2 on discouraging harassment. The Task Force consensus is that a strong clause about harassment should be included in the Code since it is to be a modern statement of the ethical responsibilities of the computing profession. As an update to the previous draft, we added virtual spaces to physical spaces as places where harassment can take place. We also broadened the harassment definition to encompass cyber bullying.

Many existing harassment policies focus exclusively on prohibiting negative actions. Draft 3 of the Code now includes a proactive call to create open and inclusive spaces. We wanted to clarify that when people feel disrespected, this can also be prohibitive to certain spaces. The new language explicitly encourages building diversity and safe environments that enable all people to feel respected.

1.6 Data collection and informed consent

In Principle 1.6, Respect Privacy, we have shifted the focus away from opting in or out of data collection, and moved to a more general requirement for informed consent procedures. The stronger emphasis on informed consent requires that users not only understand what data are being collected and what they are being used for, but that they have the ability to consent to, or to withhold consent from the data collection. This is consistent with broader international standards that are being implemented worldwide, such as the European Union's General Data Protection Regulation (GDPR),^d which we endorse.

It is important for professionals to understand that informed consent is not just about disclosure of information about data collection (for example, in a lengthy, practically unintelligible Privacy Policy), but is ideally a proactive



A successful code of ethics should reflect the values of the computing profession in a way that can help ACM members make appropriate ethical decisions.

agreement with the user about the type, content, and use of data that are being collected about them. Users should have the ability to view and update their data, and to withdraw from data collection procedures. In many circumstances, users should also be able to remove their data entirely, particularly on social media or other user-generated content platforms.

Legislation is constantly changing to catch up with technology, and different countries have different ways of approaching the issues raised by technical developments. One suggestion that was made was that we include the “right to be forgotten” in our privacy clause. This issue is a significant aspect of the EU's GDPR regulation that is coming into effect in 2018, and which has been debated in other jurisdictions as well. While we are generally supportive of this idea, we felt that for a code of ethics, the use of this term was too specific to particular legislation, and would require too nuanced a definition to be useful in this Code. Instead, as part of the privacy clause, we have required computing professionals to allow for the user's removal of data where appropriate - this captures the essence of the “right to be forgotten” in a way that we deemed to be more generalizable.

2.6 Evaluation of work and skills

Principle 2.6 clarifies the computing professional's responsibility to evaluate potential work assignments. When potential tasks are assigned, the professional should be able to evaluate the advisability and feasibility of the assignment; if these evaluations are beyond the computing professional's skill, then he or she ought to seek help in these evaluations. Professionals should further evaluate if their skill level is currently adequate to complete the assignment or if they are capable of gaining the required skills.

New Principles and Concepts

To address some of the more recent changes in computing and society we added some new principles. These principles bring attention to the professional's responsibility to a broader range of stakeholders.

^d <http://www.eugdpr.org/>

2.9 Security

Computing professionals have a responsibility to ensure that the systems they create are secure. Principle 2.9 is new, and instructs professionals to “Design and implement systems that are robustly and usably secure.” Computer hacking is a growing problem, and developments like ubiquitous computing and the “Internet of things” surround us with new vulnerabilities. True security requires usability—security features are of no practical use if users cannot or will not use them.

3.6 Legacy Systems Retirement

Principle 3.6 is new, and includes: “Retire legacy systems with care.” This principle was added to address a fundamental tension mentioned by responders: sometimes software companies must end support for systems; however, what should they do if there are users who still depend on those systems? Discontinuing support causes harm, but sometimes is necessary. This is particularly challenging because the users of legacy systems often reside in the developing world or in areas that are economically less advantaged. This new principle says that this process should be undertaken with care, and states that it is critical to notify users of the risks (especially with regard to security) of continuing to use unsupported systems.

3.4 Leadership principle changes

In the new draft version of Principle 3.4, we consolidated Principles 3.4 and 3.5 from the previous draft. Based on respondents’ comments, the Task Force decided that policies for the use of organizational computing resources are no longer such a central a concern so as to require an entire principle to itself. Instead, we amended the original principle 3.5, which was about creating policies that protect dignity, to be broader. Leaders are now expected to create and support policies and processes that not only protect dignity, but that reflect all the principles in the Code. This subsumed the original 3.4 principle, so 3.4 was eliminated, and the remaining principles were renumbered appropriately.

4 Compliance

The primary functions of a code of ethics are to state a profession’s values and

to present professionals’ commitment to those values; but many codes also allude to consequences when professionals do not comply with the code. The principles in the ACM Code provide numerous behavioral targets. The 1992 Code had a single consequence for missing any of these behavioral targets: expulsion from the ACM. The compliance section of the updated ACM Code is designed to be more flexible, to inspire and educate, as well as punish when appropriate. The new version recognizes degrees of violation, and includes opportunities of remediation less severe than expulsion.

Must or Should?

Some commenters expressed concern about the use of “must” and “should” in the Code. In Draft 3, the word “must” appears three times, and the word “should” appears 76. The impetus to use “should” stems from the aspirational nature of the Code. This is the same motivation for replacing “moral imperative” with “ethical imperative” in the development of Draft 2. The use of the word “should” is also important because during ethical deliberations, the principles in the Code can come into conflict. When this occurs, thoughtful ethical analysis may require one of the principles to yield to others. If a computing professional “must” adhere to two principles and the particular situation does not allow adhering to both simultaneously, the person necessarily violates the Code, even when a course of action is ethically justified. By using “should” professionals are given the opportunity to articulate their analysis and be transparent about their ethical reasoning.

What about those three uses of “must”? The first is in the Preamble where it says “computing professionals must always support the public good.” This reminds us that the public good is our paramount concern and is given more weight when principles in the Code conflict. The other two uses of “must” appear in the guidance for Principle 2.3, which speaks to following rules and laws. The guidance is clear: computing professionals must obey rules. The guidance articulates that it is possible for rules or laws to be unethical and when that is the case,

they ought to be challenged. Further, the computing professionals “must” accept responsibility for their actions when they challenge rules.

II. Requested changes not specifically included

Many useful ideas were suggested that were not specifically included in the newest version of the Code. Perhaps most significantly is that some respondents thought that the Code’s references to “the good of society” or “the public good” are so vague as to be meaningless. Suggestions were made to amend the Code to reflect the reality that there are many different societies, with important differences between them.

The Task Force agrees that “society” and “public” are indeed very broad, and that this breadth can be a symptom of insensitivity to the nuances of different groups and people. However, in writing a code of ethics for a global audience, authors can use a broad generalized term, focus on a single or a few particular societies, try to make a comprehensive list of relevant societies, or abandon any mention of “society.” The Task Force decided that for practical reasons, the broad general references were the proper choice for this code. Thus, in the Code the “public” or “society” is meant to encompass all affected people, and is not meant to homogenize their diversities. In the Code, a term such as “the public good” implicitly acknowledges that individuals and subsets within the public may differ about what is good in a particular situation, but we contend that there is a notion of “good” that resonates with people. This broad sense of good can be embraced, and a broad sense of evil can be shunned, without denying the importance of diversity.

Additionally, a complete cyber security standard and a complete due process standard for Code violators were suggested to be a part of the updated Code, but were not added. Although these suggested additions reflect important concerns, the Task Force decided that the additions would be more appropriately placed in different documents. For example, they could be added as independent standards supporting the Code, such as ACM Bylaws; and they could be added in supplemental

materials, such as teaching materials. There are many items that are important, even crucial, that are nonetheless not appropriate in the Code.

It is important, and tricky, to get the length of the Code right. There were some calls for the Code to be made shorter, possibly short enough to fit on a business card. There are legitimate concerns about someone choosing not to read the Code because it is too long. Rather than opt for that kind of brevity, we have targeted a middle ground. The Code must reflect the diversity of the activities computing professionals are involved in. Broader impacts of technology are not always clear or immediate, and the Code contains language to remind the reader to consider those broader impacts. Furthermore, the Code is intended to serve as a tool to use during ethical analysis. The guidance helps the professional to a deeper understanding of the principles. We hope that the Code is written in a way that facilitates a quick scan, as well as rewarding a more careful reading.

Call to action

After reading Draft 3 of the ACM Code of Ethics, please take the opportunity to make it better as a standard for the computing profession. We have provided two opportunities for you to share your comments. There is a general discussion board <https://code2018.acm.org/discuss> providing an opportunity for interested parties to discuss the suggested updates and ACM members are invited to take an online survey about the specific elements of the Code at <https://www.acm.org/code-2018-survey>. Both comment systems close **Feb. 10, 2018**.

We look forward to your comments.

III. ACM Code of Ethics and Professional Conduct: Draft 3

Draft 3 was developed by The Code 2018 Task Force. (It is based on the 2018 ACM Code of Ethics and Professional Conduct: Draft 2).

Preamble

The actions of computing professionals directly impact significant aspects of society. In order to meet their responsibilities, computing professionals must

always support the public good. The ACM Code of Ethics and Professional Conduct (“the Code”) reflects this obligation by expressing the conscience of the profession and provides guidance to support ethical conduct of all computing professionals.

The Code is designed to support all computing professionals, including current and aspiring computing practitioners, instructors, influencers, and anyone who uses technology in an impactful way. Additionally, the Code serves as a basis for remediation when violations occur. The Code includes principles formulated as statements of responsibility, based on the understanding that the public good is always the primary consideration. Each principle is supplemented by guidelines, which provide explanations to assist computing professionals in understanding and applying the principle.

Section 1 outlines fundamental ethical principles that form the basis for the remainder of the Code. Section 2 addresses additional, more specific considerations of professional responsibility. Section 3 pertains to individuals who have a leadership role, whether in the workplace or in a volunteer professional capacity. Commitment to ethical conduct is required of every ACM member, and principles involving compliance with the Code are given in Section 4.

The Code as a whole is concerned with how fundamental ethical principles apply to a computing professional’s conduct. The Code is not an algorithm for solving ethical problems; rather it serves as a basis for ethical decision making. When thinking through a particular issue, a computing professional may find that multiple principles should be taken into account, and that different principles will have different relevance to the issue. Questions related to these kinds of issues can best be answered by thoughtful consideration of the fundamental ethical principles, understanding that the public good is the paramount consideration. The entire computing profession benefits when the ethical decision making process is accountable to and transparent to all stakeholders. Open discussions about ethical issues promotes this accountability and transparency.

1. GENERAL MORAL PRINCIPLES.

A computing professional should...

1.1 Contribute to society and to human well-being, acknowledging that all people are stakeholders in computing.

This principle, concerning the quality of life of all people, affirms an obligation of computing professionals to use their skills for the benefit of society, its members, and the environment surrounding them. This obligation includes promoting fundamental human rights and protecting each individual’s right to autonomy in day-to-day decisions. An essential aim of computing professionals is to minimize negative consequences of computing, including threats to health, safety, personal security, and privacy.

Computing professionals should consider whether the results of their efforts respect diversity, will be used in socially responsible ways, will meet social needs, and will be broadly accessible. They are encouraged to actively contribute to society by engaging in pro bono or volunteer work. When the interests of multiple groups conflict, the needs of the least advantaged should be given increased attention and priority.

In addition to a safe social environment, human well-being requires a safe natural environment. Therefore, computing professionals should promote environmental sustainability both locally and globally.

1.2 Avoid harm.

In this document, “harm” means negative consequences to any stakeholder, especially when those consequences are significant and unjust. Examples of harm include unjustified physical or mental injury, unjustified destruction or disclosure of information, and unjustified damage to property, reputation, and the environment. This list is not exhaustive.

Well-intended actions, including those that accomplish assigned duties, may lead to harm. When that harm is unintended, those responsible are obligated to undo or mitigate the harm as much as possible. Avoiding harm begins with careful consideration of potential impacts on all those affected by decisions. When harm is an intentional part of the system, those responsible are obligated to ensure that the harm is

tential impacts on all those affected by decisions. When harm is an intentional part of the system, those responsible are obligated to ensure that the harm is ethically justified and to minimize unintended harm.

To minimize the possibility of indirectly harming others, computing professionals should follow generally accepted best practices. Additionally, the consequences of emergent systems and data aggregation should be carefully analyzed. Those involved with pervasive or infrastructure systems should also consider Principle 3.7.

A computing professional has an additional obligation to report any signs of system risks that might result in harm. If leaders do not act to curtail or mitigate such risks, it may be necessary to “blow the whistle” to reduce potential harm. However, capricious or misguided reporting of risks can itself be harmful. Before reporting risks, a computing professional should thoroughly assess all relevant aspects.

1.3 Be honest and trustworthy.


Honesty is an essential component of trust. A computing professional should be transparent and provide full disclosure of all pertinent system limitations and potential problems. Making deliberately false or misleading claims, fabricating or falsifying data, and other dishonest conduct are violations of the Code.

Computing professionals should be honest about their qualifications, and about any limitations in competence to complete a task. Computing professionals should be forthright about any circumstances that might lead to conflicts of interest or otherwise tend to undermine the independence of their judgment.


Computing professionals often belong to organizations associated with their work. They should not misrepresent any organization’s policies or procedures, and should not speak on behalf of an organization unless authorized to do so.

1.4 Be fair and take action not to discriminate.

The values of equality, tolerance, respect for others, and justice govern this principle. Computing professionals should strive to build diverse teams



The Code is designed to support all computing professionals, including current and aspiring practitioners, instructors, influencers, and anyone who uses technology in an impactful way.



and create safe, inclusive spaces for all people, including those of underrepresented backgrounds. Prejudicial discrimination on the basis of age, color, disability, ethnicity, family status, gender identity, labor union membership, military status, national origin, race, religion or belief, sex, sexual orientation, or any other inappropriate factor is an explicit violation of the Code. Harassment, including sexual harassment, is a form of discrimination that limits fair access to the virtual and physical spaces where such harassment takes place.

Inequities between individuals or different groups of people may result from the use or misuse of information and technology. Technologies and practices should be as inclusive and accessible as possible. Failure to design for inclusiveness and accessibility may constitute unfair discrimination.

1.5 Respect the work required to produce new ideas, inventions, creative works, and computing artifacts.

Developing new ideas, inventions, creative works, and computing artifacts creates value for society, and those who expend this effort should expect to gain value from their work. Computing professionals should therefore provide appropriate credit to the creators of ideas or work. This may be in the form of respecting authorship, copyrights, patents, trade secrets, license agreements, or other methods of assigning credit where it is due.

Both custom and the law recognize that some exceptions to a creator’s control of a work are necessary for the public good. Computing professionals should not unduly oppose reasonable uses of their intellectual works. Efforts to help others by contributing time and energy to projects that help society illustrate a positive aspect of this principle. Such efforts include free and open source software and other work put into the public domain. Some work contributes to or comprises shared community resources. Computing professionals should avoid misappropriation of these resources.

1.6 Respect privacy.

The responsibility of respecting privacy applies to computing professionals in a particularly profound way. Therefore, a computing professional should be

come conversant in privacy's various definitions and forms.

Technology enables the collection, monitoring, and exchange of personal information quickly, inexpensively, and often without the knowledge of the people affected. Computing professionals should only use personal data for legitimate ends and without violating the rights of individuals and groups. This requires taking precautions to prevent unauthorized data collection, ensuring the accuracy of data, and protecting it from unauthorized access and accidental disclosure. Computing professionals should establish transparent policies and procedures that allow individuals to give informed consent to automatic data collection, review their personal data, correct inaccuracies, and, where appropriate, remove data.

Only the minimum amount of personal information necessary should be collected in a system. The retention and disposal periods for that information should be clearly defined, enforced, and communicated to data subjects. Personal information gathered for a specific purpose should not be used for other purposes without the person's consent. Computing professionals should take special care for privacy when data collections are merged. Individuals or groups may be readily identifiable when several data collections are merged, even though those individuals or groups are not identifiable in any one of those collections in isolation.

1.7 Honor confidentiality.

Computing professionals should protect confidentiality unless required to do otherwise by a bona fide requirement of law or by another principle of the Code.

User data observed during the normal duties of system operation and maintenance should be treated with strict confidentiality, except in cases where it is evidence of the violation of law, of organizational regulations, or of the Code. In these cases, the nature or contents of that information should not be disclosed except to appropriate authorities, and a computing professional should consider thoughtfully whether such disclosures are consistent with the Code.



The responsibility of respecting privacy applies to computing professionals in a particularly profound way.

2. PROFESSIONAL RESPONSIBILITIES.

A computing professional should...

2.1 Strive to achieve high quality in both the process and products of professional work.

Computing professionals should insist on high quality work from themselves and from colleagues. This includes respecting the dignity of employers, colleagues, clients, users, and anyone else affected either directly or indirectly by the work. Computing professionals have an obligation to keep the client or employer properly informed about progress toward completing the work. Professionals should be cognizant of the serious negative consequences affecting any stakeholder that may result from poor quality work and should resist any inducements to neglect this responsibility.

2.2 Maintain high standards of professional competence, conduct, and ethical practice.

High quality computing depends on individuals and teams who take personal and group responsibility for acquiring and maintaining professional competence. Professional competence starts with technical knowledge and with awareness of the social context in which the work may be deployed. Professional competence also requires skill in reflective analysis and in recognizing and navigating ethical challenges. Upgrading necessary skills should be ongoing and should include independent study, conferences, seminars, and other informal or formal education. Professional organizations and employers should encourage and facilitate those activities.

2.3 Know, respect, and apply existing rules pertaining to professional work.

"Rules" here includes regional, national, and international laws and regulations, as well as any policies and procedures of the organizations to which the professional belongs. Computing professionals must obey these rules unless there is a compelling ethical justification to do otherwise. Rules that are judged unethical should be challenged. A rule may be unethical when it has an inadequate moral basis, it is superseded by another rule, or it

causes recognizable harm that could be mitigated through its violation. A computing professional who decides to violate a rule because it is unethical, or for any other reason, must consider potential consequences and accept responsibility for that action.

2.4 Accept and provide appropriate professional review.

High quality professional work in computing depends on professional review at all stages. Whenever appropriate, computing professionals should seek and utilize peer and stakeholder review. Computing professionals should also provide constructive, critical reviews of other's work.

2.5 Give comprehensive and thorough evaluations of computer systems and their impacts, including analysis of possible risks.

Computing professionals should strive to be perceptive, thorough, and objective when evaluating, recommending, and presenting system descriptions and alternatives. Computing professionals are in a position of trust, and therefore have a special responsibility to provide objective, credible evaluations to employers, clients, users, and the public. Extraordinary care should be taken to identify and mitigate potential risks in self-changing systems. A system for which future risks cannot be reliably predicted requires frequent re-assessment of risk as the system evolves in use, or it should not be deployed. Any issues that might result in major risk should be reported.

2.6 Have the necessary expertise, or the ability to obtain that expertise, for completing a work assignment before accepting it. Once accepted, that commitment should be honored.

A computing professional is accountable for evaluating potential work assignments.

Once it is decided that a project is feasible and advisable, the professional should make a judgment about whether the work assignment is appropriate to the professional's expertise. If the professional does not currently have the expertise necessary to complete the assignment, the professional should disclose this shortcoming to the employer or cli-

ent. The client or employer may decide to pursue the assignment with the professional after time for additional training, to pursue the assignment with someone else who has the required expertise, or to forego the assignment. A computing professional's ethical judgment should be the final guide in deciding whether to work on the assignment.

2.7 Improve public awareness and understanding of computing, related technologies, and their consequences.

Computing professionals should share technical knowledge with the public, foster awareness of computing, and encourage understanding of computing. Important issues include the impacts of computer systems, their limitations, their vulnerabilities, and opportunities that they present. Additionally, a computing professional should counter false views related to computing.

2.8 Access computing and communication resources only when authorized to do so.

No one should access another's computer system, software, or data without permission. A computing professional should have appropriate approval before using system resources unless there is an overriding concern for the public good. To support this principle, a computing professional should take appropriate action to secure resources against unauthorized use. Individuals and organizations have the right to restrict access to their systems and data so long as the restrictions are consistent with other principles in the Code.

2.9 Design and implement systems that are robustly and usably secure.

Breaches of computer security cause harm. It is the responsibility of computing professionals to design and implement systems that are robustly secure. Further, security precautions are of no use if they cannot or intentionally will not be used appropriately by their intended audience in practice; for example, if those precautions are too confusing, too time consuming, or situationally inappropriate. Therefore, the design of security features should make usability a priority design requirement.

3. PROFESSIONAL LEADERSHIP PRINCIPLES.

In this section, "leader" means any member of an organization or group who has influence, educational responsibilities, or managerial responsibilities. These principles generally apply to organizations and groups, as well as their leaders.

A computing professional acting as a leader should...

3.1 Ensure that the public good is the central concern during all professional computing work.

The needs of people—including users, those affected directly and indirectly, customers, and colleagues—should always be a central concern in professional computing. Tasks associated with requirements analysis, design, development, testing, validation, deployment, maintenance, retirement, and disposal should have the public good as an explicit criterion for quality. Computing professionals should keep this focus no matter which methodologies or techniques they use in their practice.

3.2 Articulate, encourage acceptance of, and evaluate fulfillment of the social responsibilities of members of an organization or group.

Technical organizations and groups affect broader society, and their leaders should accept the associated responsibilities. Organizational procedures and attitudes oriented toward quality, transparency, and the welfare of society reduce harm to the public and raise awareness of the influence of technology in our lives. Therefore, leaders should encourage full participation of all computing professionals in meeting social responsibilities and discourage tendencies to do otherwise.

3.3 Manage personnel and resources to enhance the quality of working life.

Leaders should ensure that management enhances, not degrades, the quality of working life. Leaders should consider the personal and professional development, accessibility requirements, physical safety, psychological well-being, and human dignity of all workers. Appropriate human-computer ergonomic standards should be used in the workplace.

3.4 Articulate, apply, and support policies and processes that reflect the principles in the Code.

Leaders should ensure that organizational policies are consistent with the ethical principles in the Code, are clearly defined, and are effectively communicated to all stakeholders. In addition, leaders should encourage and reward compliance with those policies, and take appropriate action when policies are violated.

Leaders should verify that processes used in the development of systems protect the public good and promote the dignity and autonomy of users. Designing or implementing processes that deliberately or inadvertently violate, or tend to enable the violation of, the Code's principles is ethically unacceptable.

3.5 Create opportunities for members of the organization or group to learn and be accountable for the scope, functions, limitations, and impacts of systems.

Educational opportunities are essential for all organization and group members. Leaders should ensure that opportunities are available to computing professionals to help them improve their knowledge and skills in professionalism, in the practice of ethics, and in their technical specialties. These opportunities should include experiences that familiarize computing professionals with the consequences and limitations of particular types of systems. Computing professionals should be fully aware of the dangers of oversimplified models, the improbability of anticipating every possible operating condition, the inevitability of software errors, the interactions of systems and the contexts in which they are deployed, and other issues related to the complexity of their profession.

3.6 Retire legacy systems with care.

Computing systems should be retired when it is judged impractical to continue supporting them. System developers should take care when discontinuing support for systems on which people still depend. Developers should thoroughly investigate viable alternatives to removing support for a legacy system. If these alternatives are not practical

or unacceptably risky, the developer should assist stakeholders' graceful migration from the system to an alternative. When system support ends, stakeholders should be notified of the risks of their continued use of the unsupported system.

System users should continually monitor the operational viability of their computing systems, accepting the timely replacement of inappropriate or outdated systems. The primary consideration must be the impact on stakeholders, who should be kept informed at all times.

3.7 Recognize when a computer system is becoming integrated into the infrastructure of society, and adopt an appropriate standard of care for that system and its users.

When organizations and groups develop systems that become an important part of the infrastructure of society, their leaders have a responsibility to be good stewards of these socially integrated systems. Part of that stewardship requires establishing policies for fair system access, including for those who may have been excluded. That stewardship also requires that computing professionals monitor the level of integration of their systems into the infrastructure of society. Continual monitoring of how society is using a system will allow the organization or group to remain consistent with their ethical obligations outlined in the Code. As the level of adoption changes, there are likely to be changes in the ethical responsibilities of the organization or group. When appropriate standards of care do not exist, computing professionals have a duty to ensure they are developed.

4. COMPLIANCE WITH THE CODE.

A computing professional should...

4.1 Uphold, promote, and respect the principles of the Code.

The future of computing depends on both technical and ethical excellence. Computing professionals should adhere to the principles of the Code. Each ACM member should encourage and support adherence by all computing professionals regardless of ACM membership.

4.2 Treat violations of the Code as inconsistent with membership in the ACM.

Computing professionals who recognize breaches of the Code should take actions to resolve the ethical issues they recognize, including, when reasonable, expressing their concern to the person or persons thought to be violating the Code. Possible actions also include reporting the violation to the ACM, which may result in remedial action by the ACM up to and including termination of the violator's ACM membership.

Authors

Don Gotterbarn (chair@Ethics.acm.org gotterbarn@acm.org) is chair of the ACM Committee on Professional Ethics and Professor Emeritus in the Department of Computing at East Tennessee State University, Johnson City.

Amy Bruckman (asb@cc.gatech.edu) is a professor of Interactive Computing at Georgia Institute of Technology, Atlanta.

Catherine Flick (cflick@dmu.ac.uk) is a Senior Lecturer in Computing and Social Responsibility at De Montfort University, Leicester, U.K.

Keith Miller (millerkei@ums.edu) is the Orthwein Endowed Professor for Lifelong Learning in the Sciences College of Education, University of Missouri, St. Louis.

Marty J. Wolf (mjwolf@acm.org) is a professor of Computer Science at Bemidji State University, Bemidji, MN.



CELEBRATING

50

YEARS OF
SOFTWARE ENGINEERING

GOTHENBURG
MAY 27 - JUNE 3, 2018

SWEDEN

STILL VALID SUBMISSION DATES

Posters Feb 5, 2018
Workshop papers Feb 5, 2018

www.icse2018.org



CHALMERS
UNIVERSITY OF TECHNOLOGY



UNIVERSITY OF
GOTHENBURG



City of
Gothenburg



INTERNATIONAL CONFERENCE
ON SOFTWARE ENGINEERING

2018 Artificial Intelligence · Blockchain · Cloud

— CREATING THE FUTURE —

June 25 - June 30, Seattle, USA

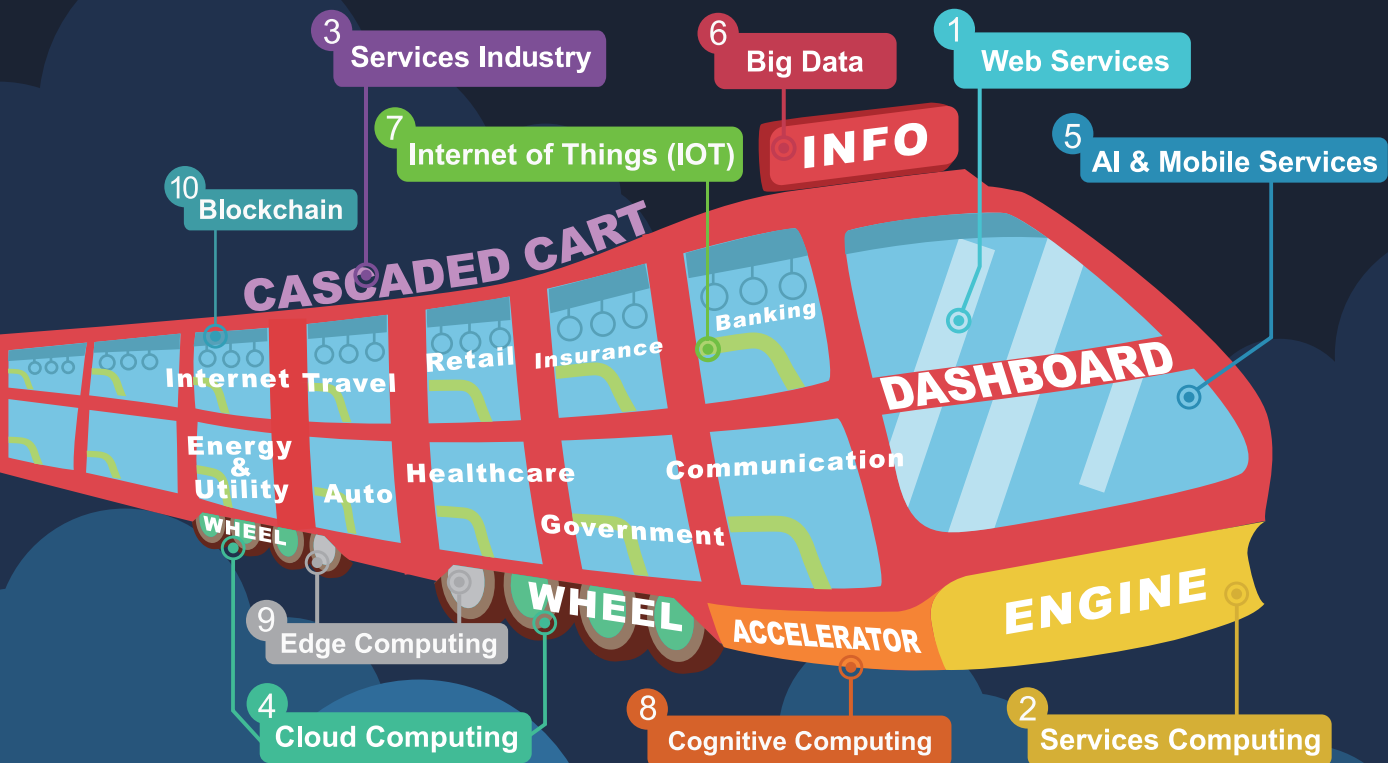
CELEBRATING THE 25th GATHERING OF ICWS



SCF

SERVICES
CONFERENCE
FEDERATION

- 1 The 25th International Conference on Web Services (**ICWS 2018**)
- 2 The 15th International Conference on Services Computing (**SCC 2018**)
- 3 The 14th World Congress on Services (**SERVICES 2018**)
- 4 The 11th International Conference on Cloud Computing (**CLOUD 2018**)
- 5 The 7th International Conference on AI & Mobile Services (**AIMS 2018**)
- 6 The 7th International Congress on Big Data (**BigData Congress 2018**)
- 7 The 3rd International Conference on Internet of Things (**ICIOT 2018**)
- 8 The 2nd International Conference on Cognitive Computing (**ICCC 2018**)
- 9 The 2nd International Conference on Edge Computing (**EDGE 2018**)
- 10 The 1st International Conference on Blockchain (**ICBC 2018**)



Submission Deadlines

2/6/2018: ICWS 2018 (<http://icws.org>)
 2/21/2018: SCC 2018 (<http://theSCC.org>)
 2/25/2018: SERVICES 2018 (<http://ServicesCongress.org>)
 2/6/2018: CLOUD 2018 (<http://theCloudComputing.org>)
 2/21/2018: AIMS 2018 (<http://ai1000.org>)

2/28/2018: BigData Congress 2018 (<http://BigDataCongress.org>)
 3/17/2018: ICIOT 2018 (<http://iciot.org>)
 3/17/2018: ICC 2018 (<http://theCognitiveComputing.org>)
 3/17/2018: EDGE 2018 (<http://theEdgeComputing.org>)
 3/17/2018: ICBC 2018 (<http://Blockchain1000.org>)

Contact: ZHANGLJ@ACM.ORG
 Dr. Liang-Jie (LJ) Zhang
 Steering Committee Chair



Largest not-for-profits organization (501(c)(3))
 dedicated for serving 30,000+ worldwide
 services computing professionals



ICWS.ORG