

COMMUNICATIONS

CACM.ACM.ORG

OF THE

ACM

07/2019 VOL.62 NO.07

On The Hourglass Model

Good Algorithms
Make Good Neighbors
Internet of Things Search Engine
Halfway Round! Growing the
Regional Special Sections

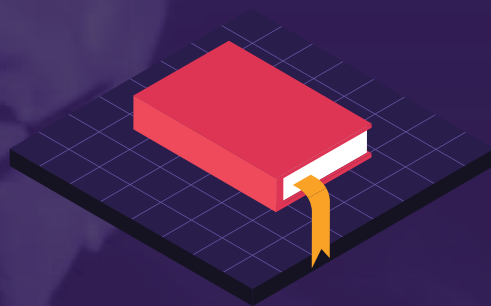


thrive

SIGGRAPH2019

LOS ANGELES • 28 JULY – 1 AUGUST

DISCOVER THE MOST BRILLIANT RESEARCH



Register today at s2019.siggraph.org/register

Heterogenous Computing

Hardware and Software Perspectives

We're introducing the big picture of heterogenous computing so you are prepared to tackle the big wave that is here to stay.

This book brings researchers and engineers to the forefront of the research frontier in the new era which began a few years ago and is expected to continue for decades.

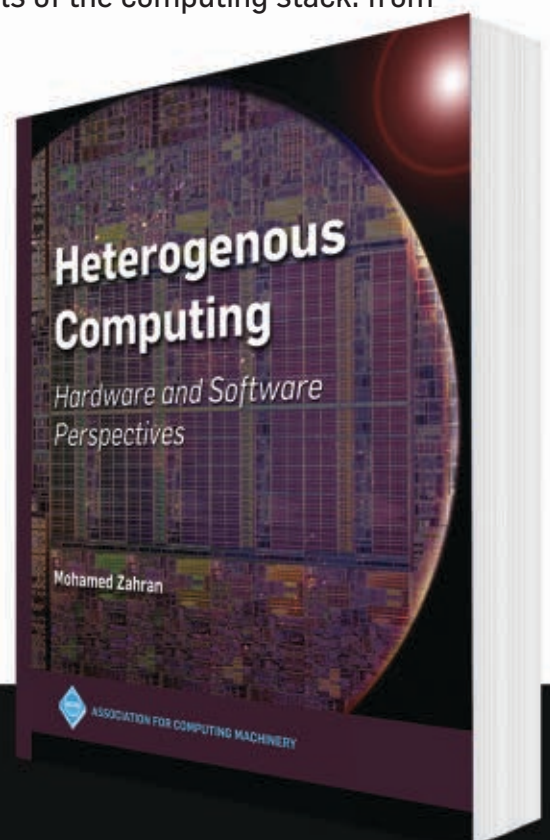
Heterogeneous computing results in both challenges and opportunities. This book discusses both. It shows that we need to deal with these challenges at all levels of the computing stack: from algorithms all the way to process technology. We discuss the topic of heterogeneous computing from different angles: hardware challenges, current hardware state-of-the-art, software issues, how to make the best use of the current heterogeneous systems, and what lies ahead.

Mohamed Zahran

ISBN: 978-1-4503-6233-7

DOI: 10.1145/3281649

<http://books.acm.org>



ACM BOOKS

Departments

- 5 **Editor's Letter**
Halfway Round! Growing the Regional Special Sections
By Andrew A. Chien
-
- 6 **Cerf's Up**
Back to the Future, Part II
By Vinton G. Cerf
-
- 7 **Vardi's Insights**
To Serve Humanity
By Moshe Y. Vardi
-
- 8 **BLOG@CACM**
Bringing More Women, Immigrants, to Computer Science
 Gloria Townsend on encouraging women to pursue CS, and Sheldon Waite on supporting immigrants to fill STEM jobs.
-
- 25 **Calendar**
-
- 94 **Careers**

Last Byte

- 96 **Upstart Puzzles**
Opioid Games
By Dennis Shasha

News



- 11 **Good Algorithms Make Good Neighbors**
 Many computer scientists doubted ad hoc methods would ever give way to a more general approach to finding nearest neighbors. They were wrong.
By Erica Klarreich
-
- 14 **The Edge of Computational Photography**
 Smartphones and consumer cameras increasingly give professional photographers a run for their money.
By Keith Kirkpatrick
-
- 17 **Protecting the 2020 Census**
 A new framework is being used to secure the 2020 U.S. Census from database reconstruction attacks.
By Logan Kugler

Viewpoints



- 20 **Legally Speaking**
API Copyrights Revisited
 Deliberating on the main arguments in recent sets of briefs filed in support of Google's U.S. Supreme Court petition.
By Pamela Samuelson
-
- 23 **Computing Ethics**
Who Benefits?
 Considering the case of smart cities.
By Susan J. Winter
-
- 26 **Broadening Participation**
A New Labor Market for People with 'Coolabilities'
 How the unique perspective and enhanced strengths accompanying disabilities can benefit the workforce.
By David Nordfors, Chally Grundwag, and V.R. Feroze
-
- 29 **Viewpoint**
GOTO Rankings Considered Helpful
 Seeking to improve rankings by utilizing more objective data and meaningful metrics.
By Emery Berger, Stephen M. Blackburn, Carla Brodley, H.V. Jagadish, Kathryn S. McKinley, Mario A. Nascimento, Minjeong Shin, Kuansan Wang, and Lexing Xie

Practice



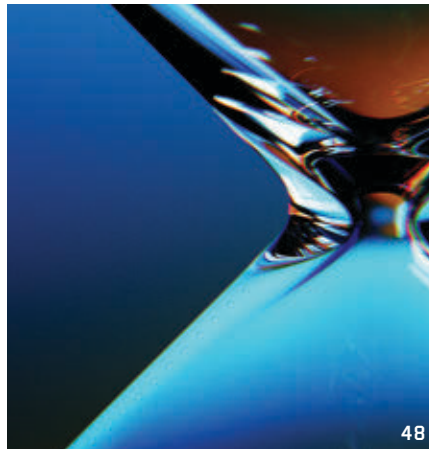
32

- 32 **Extract, Shoehorn, and Load**
Data doesn't always fit nicely into a new home.
By Pat Helland
-
- 34 **The Top 10 Things Executives Should Know About Software**
Software acumen is the new norm.
By Thomas A. Limoncelli
-
- 41 **Access Controls and Healthcare Records: Who Owns the Data?**
A discussion with David Evans, Richard McDonald, and Terry Coatta.



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

Contributed Articles



48

- 48 **On The Hourglass Model**
Used in the design of the Internet and Unix, the layered services of the hourglass model have enabled viral adoption and deployment scalability.
By Micah Beck



Watch the author discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/on-the-hourglass-model>

- 58 **Ways of Thinking in Informatics**
An innovative, entry-level informatics course enables students to ponder CS problems in different ways, from different perspectives.
By Christopher Frauenberger and Peter Purgathofer

Review Articles

- 66 **Internet of Things Search Engine**
Tracing the complicated yet still relatively unripe area of the Internet of Things search engine—from concepts, to classification, and open issues.
By Nguyen Khoi Tran, Quan Z. Sheng, M. Ali Babar, Lina Yao, Wei Emma Zhang, and Shahram Dustdar



Watch the authors discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/iot-search-engine>

- 74 **Unifying Logical and Statistical AI with Markov Logic**
Markov logic can be used as a general framework for joining logical and statistical AI.
By Pedro Domingos and Daniel Lowd

Research Highlights

- 85 **Technical Perspective**
Do You Know Why Your Web Pages Load Faster?
By Costin Raiciu
-
- 86 **Taking a Long Look at QUIC: An Approach for Rigorous Evaluation of Rapidly Evolving Transport Protocols**
Arash Molavi Kakhki, Samuel Jero, David Choffnes, Cristina Nita-Rotaru, and Alan Mislove

**About the Cover:**

This month's cover story spotlights the hourglass model used in the design of the Internet and Unix. Micah Beck explores the model's powerful elements, which combine visually to form an hourglass shape, beginning on p. 48. Cover illustration by Peter Crowther Associates.



Association for Computing Machinery
Advancing Computing as a Science & Profession



ACM, the world's largest educational and scientific computing society, delivers resources that advance computing as a science and profession. ACM provides the computing field's premier Digital Library and serves its members and the computing profession with leading-edge publications, conferences, and career resources.

Executive Director and CEO

Vicki L. Hanson

Deputy Executive Director and COO

Patricia Ryan

Director, Office of Information Systems

Wayne Graves

Director, Office of Financial Services

Darren Ramdin

Director, Office of SIG Services

Donna Cappel

Director, Office of Publications

Scott E. Delman

ACM COUNCIL

President

Cherri M. Pancake

Vice-President

Elizabeth Churchill

Secretary/Treasurer

Yannis Ioannidis

Past President

Alexander L. Wolf

Chair, SGB Board

Jeff Jortner

Co-Chairs, Publications Board

Jack Davidson and Joseph Konstan

Members-at-Large

Gabriele Anderst-Kotis; Susan Dumais;

Renée McCauley; Claudia Bauzer Medeiros;

Elizabeth D. Mynt; Pamela Samuelson;

Theo Schlossnagle; Eugene H. Spafford

SGB Council Representatives

Sarita Adve and Jeanna Neefe Matthews

BOARD CHAIRS

Education Board

Mehran Sahami and Jane Chu Prey

Practitioners Board

Terry Coatta

REGIONAL COUNCIL CHAIRS

ACM Europe Council

Chris Hankin

ACM India Council

Abhiram Ranade

ACM China Council

Wenguan Chen

PUBLICATIONS BOARD

Co-Chairs

Jack Davidson and Joseph Konstan

Board Members

Phoebe Ayers; Edward A. Fox; Chris Hankin;

Xiang-Yang Li; Nenad Medvidovic;

Tulika Mitra; Sue Moon; Michael L. Nelson;

Sharon Oviatt; Eugene H. Spafford;

Stephen N. Spencer; Divesh Srivastava;

Robert Walker; Julie R. Williamson

ACM U.S. Technology Policy Office

Adam Eisgrau,

Director of Global Policy and Public Affairs

1701 Pennsylvania Ave NW, Suite 200,

Washington, DC 20006 USA

T (202) 580-6555; acmpo@acm.org

Computer Science Teachers Association

Jake Baskin

Executive Director

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

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

STAFF

DIRECTOR OF PUBLICATIONS

Scott E. Delman

cacm-publisher@cacm.acm.org

Executive Editor

Diane Crawford

Managing Editor

Thomas E. Lambert

Senior Editor

Andrew Rosenbloom

Senior Editor/News

Lawrence M. Fisher

Web Editor

David Roman

Editorial Assistant

Danbi Yu

Art Director

Andrij Borys

Associate Art Director

Margaret Gray

Assistant Art Director

Mia Angelica Balaquiot

Production Manager

Bernadette Shade

Intellectual Property Rights Coordinator

Barbara Ryan

Advertising Sales Account Manager

Ilia Rodriguez

Columnists

David Anderson; Michael Cusumano;

Peter J. Denning; Mark Guzdial;

Thomas Haigh; Leah Hoffmann; Mari Sako;

Pamela Samuelson; Marshall Van Alstyne

CONTACT POINTS

Copyright permission

permissions@hq.acm.org

Calendar items

calendar@cacm.acm.org

Change of address

acmhelp@acm.org

Letters to the Editor

letters@cacm.acm.org

WEBSITE

<http://cacm.acm.org>

WEB BOARD

Chair

James Landay

Board Members

Marti Hearst; Jason I. Hong;

Jeff Johnson; Wendy E. McKay

AUTHOR GUIDELINES

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

ACM ADVERTISING DEPARTMENT

1601 Broadway, 10th Floor

New York, NY 10019-7434 USA

T (212) 626-0686

F (212) 869-0481

Advertising Sales Account Manager

Ilia Rodriguez

ilia.rodriguez@hq.acm.org

Media Kit acmm mediasales@acm.org

Association for Computing Machinery (ACM)

1601 Broadway, 10th Floor

New York, NY 10019-7434 USA

T (212) 869-7440; F (212) 869-0481

EDITORIAL BOARD

EDITOR-IN-CHIEF

Andrew A. Chien

eic@cacm.acm.org

Deputy to the Editor-in-Chief

Lihan Chen

cacm.deputy.to.eic@gmail.com

SENIOR EDITOR

Moshe Y. Vardi

NEWS

Co-Chairs

Marc Snir and Alain Chesnais

Board Members

Monica Divitini; Mei Kobayashi;

Rajeev Rastogi; François Sillion

VIEWPOINTS

Co-Chairs

Tim Finin; Susanne E. Hambrusch;

John Leslie King; Paul Rosenbloom

Board Members

Michael L. Best; Judith Bishop;

James Grimmelmann; Mark Guzdial;

Haym B. Hirsch; Richard Ladner;

Carl Landwehr; Beng Chin Ooi;

Francesca Rossi; Len Shustek; Loren Terveen;

Marshall Van Alstyne; Jeannette Wing;

Susan J. Winter

PRACTICE

Co-Chairs

Stephen Bourne and Theo Schlossnagle

Board Members

Eric Allman; Samy Bahra; Peter Bailis;

Betsy Beyer; Terry Coatta; Stuart Feldman;

Nicole Forsgren; Camille Fournier;

Jessie Frazelle; Benjamin Fried; Tom Killalea;

Tom Limoncelli; Kate Matsudaira;

Marshall Kirk McKusick; Erik Meijer;

George Neville-Neil; Jim Waldo;

Meredith Whittaker

CONTRIBUTED ARTICLES

Co-Chairs

James Larus and Gail Murphy

Board Members

William Aiello; Robert Austin; Kim Bruce;

Alan Bundy; Peter Buneman; Jeff Chase;

Andrew W. Cross; Yannis Ioannidis;

Gal A. Kaminka; Ben C. Lee; Igor Markov;

Lionel M. Ni; Adrian Perrig; Doina Precup;

Marie-Christine Rousset; Krishan Sabnani;

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

Sebastian Uchitel; Hannes Werthner;

Reinhard Wilhelm

RESEARCH HIGHLIGHTS

Co-Chairs

Azer Bestavros and Shriram Krishnamurthi

Board Members

Martin Abadi; Amr El Abbadi;

Animashree Anandkumar; Sanjeev Arora;

Michael Backes; Maria-Florina Balcan;

David Brooks; Stuart K. Card; Jon Crowcroft;

Alexei Efros; Bryan Ford; Alon Halevy;

Gernot Heiser; Takeo Igarashi; Sven Koenig;

Greg Morrisett; Tim Roughgarden;

Guy Steele, Jr.; Robert Williamson;

Margaret H. Wright; Nikolai Zeldovich;

Andreas Zeller

SPECIAL SECTIONS

Co-Chairs

Sriram Rajamani, Jakob Rehof,

and Haibo Chen

Board Members

Tao Xie; Kenjiro Taura; David Padua

ACM Copyright Notice

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

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

Subscriptions

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

ACM Media Advertising Policy

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

Single Copies

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

COMMUNICATIONS OF THE ACM

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

POSTMASTER

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

Printed in the USA.



Association for
Computing Machinery





DOI:10.1145/3338465

Andrew A. Chien

Halfway Round! Growing the Regional Special Sections

IN NOVEMBER 2017, we launched *Communications'* Regional Special Sections with goals to 1) bring all ACM members a deep, insightful window into the most exciting trends and changes in regions around the world, and 2) include diverse, leading computing professionals as contributors to the magazine.^a A year and half later, I'm happy to report we're halfway around the world and gaining momentum!

Our vision is to represent the best of computing leadership and distinctive development for each region, and bring a sharp focus on:

- ▶ Leading technical and research advances and activities;
- ▶ Leading and emerging industry and research players;
- ▶ Innovation and shape of computing in the region; and,
- ▶ Unique challenges and opportunities.

The special sections have emerged as polychromatic windows into each region with widely varied topics and contributors. The China Region special section in the November 2018 issue included 13 articles, mixed short and long, covering topics ranging from tech role models to data markets, from quantum communication to extreme cloud bursting, and fintech and killer-apps for autonomous vehicles. Contributors were split across companies and universities, with 30 of the 40 total authors never previously contributing to *Communications*. The April 2019 Europe Region special section included 14 articles covering topics ranging from data privacy law to Europe's plans for ICT leadership as well as insightful, distinctive views



Bengaluru Workshop participants, February 2019.

on the central importance of informatics and what constitutes responsible computing in the modern world. Of the 28 authors, 23 had never previously contributed to *Communications*, and the authors reflected a diverse mix of corporate, government policy, and university affiliations. The nascent India Region special section for November 2019 is projected at 15–17 articles with nearly all of the contributors likely to have never previously contributed to *Communications*.

While each workshop to kick off regional section teams has been a dynamic, diverse, and creative gathering, the Bengaluru Workshop for the India Region was particularly dramatic as it included participants from Sri Lanka, Pakistan, and India, and was held on February 23, 2019—right in the midst of international conflict between Pakistan and India over a terrorist attack, aerial bombardment response, and more! Nothing could better underscore the importance of international scientific collaboration, computing professionalism, and human relationships founded on mutual respect.

The Regional special sections will circle the globe—so plans for future sections are exploring how to best cover East Asia, Oceania, North Africa and Arabia, as well as the Americas before circling back around. At current cadence, our period will be three years (a little

slower than Verne's Phileas Fogg!^b)

These sections are thriving because of the extraordinary creativity and enthusiasm of the Regional co-leaders and article contributors. But we now have a strong international team of co-chairs on *Communications'* Editorial Board: Sriram Rajamani (Microsoft), Jakob Rehof (Dortmund and Fraunhofer), and Haibo Chen (SJTU and Huawei) are driving the effort. Thanks also to David Padua, Kenjiro Taura, and Tao Xie who contribute as members of the editorial board. If you have a passion to join this effort, we can always use more talent and energy! Finally, the special sections are thriving because the extraordinary efforts of Lihan Chen, my deputy, and the stellar *Communications'* publication team, led by Diane Crawford. Thanks to all!

The international computing profession's importance has never been greater than in this time of growing distrust and international tension;^c there is much to be done!

Andrew A. Chien, EDITOR-IN-CHIEF

Andrew A. Chien is the William Eckhardt Distinguished Service Professor in the Department of Computer Science at the University of Chicago, Director of the CERES Center for Unstoppable Computing, and a Senior Scientist at Argonne National Laboratory.

^b J. Verne. *Le tour du monde en quatre-vingts jours* (*Around the World in Eighty Days*), 1873.

^c A.A. Chien. Open Collaboration in an Age of Distrust. *Commun. ACM* (Jan. 2019).

^a A.A. Chien. Here Comes Everybody ... to *Communications*. *Commun. ACM* (Mar. 2018).



Vinton G. Cerf

DOI:10.1145/3338516

Back to the Future, Part II

This year is the 50th anniversary of the activation of the Arpanet project. The first packet switches (called Interface Message Processors or IMPs) were installed at UCLA,

SRI International [then Stanford Research Institute], UC Santa Barbara, and University of Utah. Host computers at UCLA and SRI made their first connection through the Arpanet on October 29, 1969.

Tom Standage's book, *The Victorian Internet*,^a highlights the drama of the arrival of the telegraph in the mid-19th century. In its earliest incarnation, telegraph operators coded text messages in Morse Code^b and sent them to the next operator whose receiver was connected on a dedicated, point-to-point wire. The transcribed message was then re-sent by the receiving operator to the next one, proceeding hop-by-hop until it reached the final operator who transcribed the message and delivered it to the recipient, typically by courier. This was called a "store-and-forward" message-switching system because at each hop, messages were temporarily stored at the intermediate telegraph operator's office until they could be re-sent to the next operator.^c

When teletype machines were invented, they punched holes in paper tape coded with 5 bits to represent up to 32 characters or symbols. Eventually an 8-bit coding scheme was used to expand the available character set to include uppercase and lowercase

as well as other special characters. An operator would enter a message on the teletype and a paper tape would be punched. The operator would tear off the paper tape and hang it on a peg next to the teletype that would be used to send the message to the next hop. This was called a "torn tape" system since the punched paper tapes would be torn from the originating teletype and fed into a teletype that would automatically read the tape and send the signals to the next teletype. The receiving teletype, which was connected by a dedicated circuit, would then punch out a copy of the message on a tape which would again be torn from the machine ...

Interestingly, the telephone emerged from research attempting to allow one wire to carry more than one telegraph message at the same time. Telephone circuits were built manually with patch panels to connect the caller to the called party. Eventually automatic switching was possible and circuits were built automatically from source to destination. This allowed teletypes to be directly connected end-to-end without requiring intermediate queueing. No more store-and-forward was necessary as the two end-points were directly connected by a circuit.

Returning to the Arpanet, the challenge there was to find an alternative to connecting the machines by way of dial-up circuits. Such a process would have been painfully slow since each connection would take seconds to complete while the computers,

even in the 1960s, were operating at microsecond speeds. Ironically, the solution was to abandon the circuit switching concept that had been so effective for the teletype system and introduce a store-and-forward packet switching system! Since the circuits connecting the packet switches were fixed and dedicated, there was no circuit switching delay. Rather, packets from the host computers could be sent as soon as they were ready to the directly connected packet switch, which would forward them in milliseconds to the next packet switch, completing the hop-by-hop journey in tens of milliseconds. In the original Arpanet, the dedicated circuits were operated at 50 kilobits/second, so a 1,000-bit packet would take 20 milliseconds to transmit, per hop. A five-hop transmission would consume 100 milliseconds.

The subsequent Internet made use of circuits running at a minimum of 1.5 megabits/second in the early 1980s and today at 100–400 gigabits/second. The transmission delays are negligible, per hop, and are dominated by speed of light and/or queueing delays in the event of congestion. So we have come full circle back to the store-and-forward days of the telegraph, but at sub-nanoseconds of transmission delay per bit. We've gone back to the future! 

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 T. Standage. *The Victorian Internet*. Walker & Company, 1998; ISBN-13: 978-162040592, ISBN-10: 162040592X, ISBN 0-8027-1342-4 (hardcover), ISBN 978-0-8027-1604-0 (paperback).

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

c For more information, see my June 2019 Cerf's Up column.



DOI:10.1145/3338092

Moshe Y. Vardi

To Serve Humanity

TO SERVE MAN," a short science fiction story written by Damon Knight in 1950, was the basis for a 1962 episode of the classic TV series "The Twilight Zone." The story opens at a special session of the United Nations where three alien emissaries are testifying that the purpose of their mission to Earth is to bring humans "the peace and plenty which we ourselves enjoy, and which we have in the past brought to other races throughout the galaxy." The title of the story uses the dual meanings of the verb to serve: "to assist" or "to provide as a meal." At the conclusion of the story, the narrator realizes to his horror that an alien book titled *How to Serve Man* is a cookbook!

This story came back to mind when I recently attended the First International Workshop on Digital Humanism^a (organized by the Faculty of Informatics of Technical University of Vienna, Austria). The workshop, attended by a highly multidisciplinary audience, was motivated by a deep sense of frustration with the current relationship between technology and society. There is no doubt that we are in the midst of a profound transformation of our society, with computer science and its artifacts as a major driver of change. Whereas this development opens enormously positive possibilities for our future, it also raises serious questions and has dramatic downsides—as was expressed by Tim Berners-Lee in 2017 in his anguished declaration "The system is failing." Instead of technology assisting humanity, technology sometimes seems to "eat" humanity.

Technology has always been a two-edged sword. We discovered fire approximately one million years ago; a discovery so crucial to human progress

and development that Greek mythology attributes it to Prometheus stealing it from the gods to give it to humans, but people still die from fire regularly. In fact, according to the classical Greek poet Hesiod, when Prometheus stole fire from heaven, Zeus, the king of the gods, took vengeance by sending Pandora with a gift box to Prometheus' brother. The box contained sickness, death, and many other unspecified evils, which were then released into the world. Greek mythology, it seems, is telling us that technology is never free from adverse consequences.

Other scientific disciplines have had their moment of abrupt realization of the dual nature of technology. Chemists gave us the first "weapons of mass destruction" in World War I. The German chemist Otto Hahn, a future Nobel laureate, was recruited to the German chemical weapons program. Hahn went to the eastern front to see for himself the capabilities of this new weapon. "I was very ashamed and deeply agitated," he wrote later. The American physicist Robert Oppenheimer was the wartime head of the Los Alamos Laboratory and is among those credited with being the "father of the atomic bomb" for his role in the Manhattan Project, the American World War II effort to develop nuclear weapons. When the first atomic bomb was successfully detonated in July 1945 in New Mexico, Oppenheimer recited the words from the Sanskrit scripture Bhagavad Gita: "Now I am become Death, the destroyer of worlds." He went on later to oppose the development of the fusion bomb.

Biologists had the foresight to be proactive. The Asilomar Conference on Recombinant DNA held in 1975 at the Asilomar Conference Center in California was an influential event organized to discuss the potential biohazards

and regulation of biotechnology. The conference brought together approximately 140 biologists, lawyers, and physicians to draw up voluntary guidelines^b to ensure the safety of recombinant DNA technology.

The recent Vienna workshop focused on the broad societal role of digital technology. Its basic premise was that technology is for people and not the other way round. We need to put "humanity" at the center of our work. The goal of the workshop was to raise questions, rather than provide answers. It was acknowledged that computer science alone cannot provide answers to the challenges raised by the digital transformation. Yet the participants were convinced it is possible to influence the future of science and technology and, in consequence, society. They were also aware of their joint responsibility for the current situation and the future—both as professionals and citizens.

The workshop launched the drafting of the Vienna Manifesto on Digital Humanism,^c which was proposed and discussed at the workshop, and finalized in a cooperative online mode afterward. The Manifesto is a call to reflect and act on current and future technological development, to provide input to future discussions, and to influence societal and policy decision making. We have unleashed the "information revolution." Its outcome depends on us!

Follow me on Facebook and Twitter. 

^b <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC432675/>

^c <https://www.informatik.tuwien.ac.at/dighum/manifesto/>

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

Copyright held by author.

^a <https://www.ec.tuwien.ac.at/dighum2019>

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:10.1145/3329705

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

Bringing More Women, Immigrants, to Computer Science

Gloria Townsend on encouraging women to pursue CS, and Sheldon Waite on supporting immigrants to fill STEM jobs.



**Gloria Townsend
Women Now
Outnumber Men in
Medical Schools.
Computer Science
Should Be Next.**

<http://bit.ly/2PHmm80>

April 26, 2019

In March we celebrated Women's History month, but there were few female computer scientists to celebrate. Women receive only 16% of U.S. bachelor's degrees in pure computer science (CS). In an age when women outnumber men in medical schools, we scratch our heads when we see such a small number. What's going on? The National Center for Women & IT (NCWIT) reports: "By 2026, 3.5 million computing-related job openings are expected. At the current rate, only 17% of these jobs could be filled by U.S. computing bachelor's degree recipients." More women graduating in CS will reduce the magnitude of the looming crisis.

ACM launched a pioneering effort to address plummeting graduation rates in the early 1990s with the creation of a

Council on Women in Computing (ACM-W). ACM-W seeks to recruit, retain, and celebrate women in computing.

I joined ACM-W in 2000, bringing with me a community-building idea for women in computing. In 1996, I used the fledgling Internet to count the numbers of female computer science majors in Indiana. Small pockets of women dotted the state. I dreamed of uniting these small groups of Indiana women by inviting them to attend a regional conference where each woman would find role models and a peer community. Attendees could build confidence by giving short "lightning talks" and poster presentations. The conference would dispel the myth of the lonely programmer hidden away in a cubicle by offering keynotes and panel presentations that share accurate career information. Women could find job and internship opportunities offered by industry and graduate school sponsors, who also serve as role models.

My colleagues and I organized the first conference, called an ACM Celebration, in Indiana in 2004. We imag-

ined organizing Celebrations all over the world, so that no woman would feel isolated. Fast-forward 15 years, and ACM Celebrations now span the globe: Serbia, Chile, Ukraine, Canada, Philippines, Pakistan, Ireland, Turkey, Spain, and India, to name a few.

ACM-W Student Chapters sustain energy after one Celebration ends and before another begins. The ACM-sponsored organizations provide local activities on a smaller scale, but with the same Celebration mission to recruit, retain, and build community for women.

Beyond Celebrations and Chapters, what can institutions do? DePauw University awarded 47% of its computer science degrees to women in 2017—almost three times the national average. How? A lineup of traditional methods such as mentoring and role-modeling added new computer science majors, as did more specialized techniques like our CS Try-out. We invite every first-year woman (immediately before registration) to a preview of the introductory class, where third- and fourth-year female majors (and role models) sit alongside attendees to teach all that is needed to complete the first laboratory. The student teachers also talk briefly about their computing opportunities and career plans. Many women have zero computing experience, so the event removes the mystery surrounding computing classrooms and careers, as the older students describe their internship and research opportunities and their classroom projects—especially their impressive senior projects.

Finally, what can individuals do? We underestimate the power of encouragement, in my opinion. Once I heard a young woman tell an audience, “I am a computer scientist today, because of three words that a professor wrote on my exam.” In response to a recent post on my Facebook page, a woman wrote, “And here, just this past week, I once again happened upon my first CS1 exam upon which Gloria had written, ‘CS Major????’” (Ashley had saved the exam for 15 years). I encourage all of my talented students—both men and women. Woman after woman later tells me (as these two did) how my words influenced her. The number of men who have expressed the same sentiment? Zero!

This story is bittersweet. It’s sad that women hunger for words of encouragement and value notes that take seconds to write. At the same time, this story about encouragement tells us how we—as sisters, brothers, mothers, fathers, partners, teachers, and other mentors—can change girls’ and women’s lives.

An important time to encourage girls is in elementary school. These very young girls have as much interest in computing and technology as young boys do. There’s no unimportant time to encourage, because girls begin to lose interest as they progress through middle school and high school. Support the many worthwhile programs that target girls and college women. Above all, use *only* supportive language when talking with girls and young women about computing.

Sheldon Waite **Immigrants Help Solve the** **Looming STEM Worker Shortage**

<http://bit.ly/2Y0APQh>

April 12, 2019

As an engineering hardware manager working in the rapidly growing automotive electronics industry, I’ve been baffled by politicians who champion anti-immigration policies. If we want our economy to prosper, we should eagerly welcome the world-class talent that’s knocking at our door.

I should know. I’ve witnessed firsthand the excellence newcomers bring to this country. About half of my 30-person engineering team is comprised of foreign-born workers or children of recent immigrants. As a hiring manager, I have recruited the best, assembling

a whip-smart, talented group that keeps us on the cutting edge of a highly competitive field.

Car companies like Mercedes-Benz and Ford hire us to make high-tech accessories for their cars, such as screens, radios, embedded cell-phones, and Wi-Fi devices. As connectivity devices become more integral to the car industry, demand for our work continues to rise. I have worked in this industry for more than 15 years and see how important diversity is to staying ahead of the competition. The type of work we do is highly technical, but also creative, because we are always trying to solve problems. That’s why the diverse perspectives on my team are so critical to helping us find out-of-the-box solutions faster. Additionally, many of our bilingual employees give us a competitive advantage in the global economy. I’m proud to work alongside colleagues who work every day to make safer cars that improve consumers’ lives.

I have heard people say they want to restrict immigration because they fear immigrants will take our jobs. But in my experience, there are not enough American-born workers to fill all these jobs. I just looked through a stack of résumés for summer internships, and the vast majority of applicants were immigrants or first- and second-generation Americans. This is emblematic of a broader trend: immigrants play a large role in science, technology, engineering, and math—or “STEM”—fields. In my home state of Illinois, for example, 24.1% of STEM workers were born in another country, according to a New American Economy analysis of various 2017 datasets (<https://www.newamericaneconomy.org/locations/illinois/>).

It is difficult to overstate this importance because employment in STEM jobs has grown significantly, exceeding overall job growth (<https://pewrsr.ch/2UUpYPA>). From 1990 to 2016, STEM occupations have grown 79%, with computer jobs increasing 338% over that same period, according to the Pew Research Center (<https://pewrsr.ch/2v5gBN>). These fields are expected to play a critical role in future U.S. economic growth.

In my state—Illinois—the labor market for tech talent is tighter than that on the coasts. That is why I support

policies that help keep these skilled workers right here, where they can contribute to the workforce and the economy. In Illinois alone, immigrants contribute \$17.6 billion in taxes, include 118,055 entrepreneurs, and employ 390,685 people (<http://bit.ly/2DJtvRr>). Nationally, they pay \$405.4 billion in taxes, account for nearly 3.2 million entrepreneurs, and create approximately 8 million jobs (<http://bit.ly/2vtCYb5>). If our country’s policies send the message “we don’t want you here,” then where will all this talent go? To foreign competitors.

I am lucky to have a team of talented engineers, but I know the tech industry as a whole struggles to find the skilled workers it needs. The recent spate of anti-immigration policies doesn’t help their case. The H-1B visa program, which is the main way U.S. companies hire high-skilled foreign workers, is capped at 65,000 visas, plus an additional 20,000 visas for foreign applicants with a U.S. graduate degree (<http://bit.ly/2VDzoK4>). Demand for these workers in recent years far exceeds the available number of visas. Last year, nearly 200,000 people applied (<http://bit.ly/2vyg5De>), and for the past six consecutive years, the H-1B visa cap has been reached within a week of the application period opening (<http://bit.ly/2PCMS3a>). And existing H-1B visa holders are grappling with the looming possibility the program allowing their spouses to work is on the chopping block (<http://bit.ly/2J8teuO>). This presents just another incentive to move to a country that is more inclusive.

Immigrants play a critical role in filling labor gaps; we should be embracing them. That means reversing policies that deter the hiring of foreign-born workers, creating a more streamlined immigration process, and cultivating a more supportive environment for newcomers.

Increasing immigration is the key to keeping our economy thriving. And the great thing about America is that people want to come here. So let’s welcome them with open arms.

Gloria Townsend is a professor, and department chair of computer science, at DePauw University in Greencastle, IN, USA. **Sheldon Waite** is an engineering hardware manager in Chicago’s Northwest suburbs.

© 2019 ACM 0001-0782/19/7 \$15.00

SHAPE THE FUTURE OF COMPUTING. JOIN ACM TODAY.

www.acm.org/join/CAPP

SELECT ONE MEMBERSHIP OPTION

ACM PROFESSIONAL MEMBERSHIP:

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

ACM STUDENT MEMBERSHIP:

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

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

PAYMENT INFORMATION

Name

Mailing Address

City/State/Province

ZIP/Postal Code/Country

- ☐ Please do not release my postal address to third parties

Email Address

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

- ☐ AMEX ☐ VISA/MasterCard ☐ Check/money order

Credit Card #

Exp. Date

Signature

Purposes of ACM

ACM is dedicated to:

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

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

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

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

BE CREATIVE. STAY CONNECTED. KEEP INVENTING.



Association for
Computing Machinery

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

1-800-342-6626 (US & Canada)
1-212-626-0500 (Global)
Hours: 8:30AM - 4:30PM (US EST)

Fax: 212-944-1318
acmhelp@acm.org
www.acm.org/join/CAPP

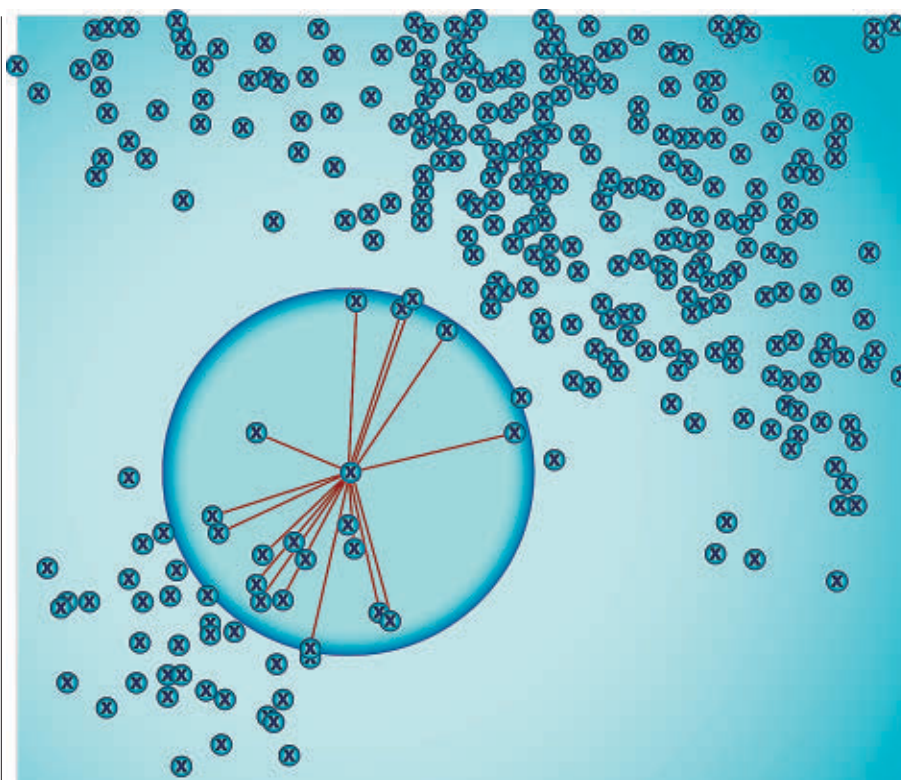
Good Algorithms Make Good Neighbors

Many computer scientists doubted ad hoc methods would ever give way to a more general approach to finding nearest neighbors. They were wrong.

A HOST OF DIFFERENT tasks—such as identifying the song in a database most similar to your favorite song, or the drug most likely to interact with a given molecule—have the same basic problem at their core: finding the point in a data set that is closest to a given point. This “nearest neighbor” problem shows up all over the place in machine learning, pattern recognition, and data analysis, as well as many other fields.

Yet the nearest neighbor problem is not really a single problem. Instead, it has as many different manifestations as there are different notions of what it means for data points to be similar. In recent decades, computer scientists have devised efficient nearest neighbor algorithms for a handful of different definitions of similarity: the ordinary Euclidean distance between points, and a few other distance measures.

However, “every time you needed to work with a new space or distance measure, you would kind of have to start from scratch” in designing a nearest neighbor algorithm, said Rasmus Pagh, a computer scientist



at the IT University of Copenhagen. “Each space required some kind of craftsmanship.”

Because distance measures are so varied, many computer scientists

doubted these ad hoc methods would ever give way to a more general approach that could cover many different distance measures at once. Now, however, a team of five computer

scientists has proven the doubters—who originally included themselves—were wrong.

In a pair of papers published last year (in the *Proceedings of the ACM Symposium on Theory of Computing* and the IEEE Annual Symposium on Foundations of Computer Science, respectively), the researchers set forth an efficient approximation algorithm for nearest neighbor search that covers a wide class of distance functions. Their algorithm finds, if not the very closest neighbor, then one that's almost as close, which is good enough for many applications.

The distance functions covered by the new algorithm, called norms, “encompass the majority of interesting distance functions,” said Piotr Indyk, a computer scientist at the Massachusetts Institute of Technology.

The new algorithm is a big leap forward, Pagh said, who added, “I wouldn’t have guessed such a general result was possible.”

An Unbelievable Tangle

Four of the paper’s five authors—Alexandr Andoni and Erik Waingarten of Columbia University, Aleksandar Nikolov of the University of Toronto and Ilya Razenshteyn of Microsoft Research Redmond—originally set out to prove the opposite of what they finally proved. Namely, they tried to show there are some normed spaces, and some datasets within these spaces, for which no nearest neighbor algorithm can be significantly faster than the laborious process of comparing a given point to every single dataset point to find the closest one.

It seemed plausible to the four researchers that some such norms should exist, because norms are extremely diverse. A norm is simply a distance function in ordinary space (of any dimension) that has the property that whenever you scale the coordinates of two points by a given factor, the distance between them scales by the same factor. Ordinary Euclidean distance is a norm, and so is “Manhattan” distance, which measures the number of city blocks between two points.

But there are many other norms. In fact, given any convex shape that is symmetric around the origin, there is

some corresponding norm for which that shape is the “unit ball”—the ball of radius 1 around the origin. In high-dimensional spaces, the assortment of different norms “is a wild world, but also very expressive,” said Assaf Naor, a mathematician at Princeton University and the new paper’s fifth author.

“There’s a lot of freedom in designing a normed space,” Andoni said. Because of this, he, Nikolov, Razenshteyn, and Waingarten believed at first that it should be possible to cook up some normed space that contains an “expander graph”—an object that is nearest neighbor search’s worst enemy.

An expander graph is a network (a collection of nodes and edges) that has a counterintuitive combination of sparseness and connectivity. Each node in an expander connects to relatively few other nodes; nevertheless, there’s no way to separate the graph into reasonably large chunks without cutting through many edges.

“Somehow globally it creates an unbelievable tangle that can never be separated,” Naor said. “It’s completely nonobvious that such objects even exist.”

Yet they do exist, and since every graph comes with a natural notion of distance—simply declare the length of each edge to be 1—it makes sense to talk about the nearest neighbor problem in the context of graphs. Expander graphs, because of their connectivity patterns, defy the kind of data-structuring approaches at the heart of nearest neighbor algorithms.

Expander graphs, because of their connectivity patterns, defy the kind of data-structuring approaches at the heart of nearest neighbor algorithms.

Computer scientists designing a nearest neighbor algorithm generally start by trying to partition the dataset into sections in such a way that points in different sections tend to be far from each other. That way, once you’ve decided that your chosen point belongs in a particular section, you can feel confident that its nearest neighbor—or at least an approximate nearest neighbor—lives within that section, too.

However, in an expander graph, it’s impossible to carry out an effective partition, since all but the most lopsided partitions are going to separate many points that are close to each other. Andoni, Nikolov, Razenshteyn, and Waingarten were able to show that there can be no efficient nearest neighbor algorithm for an expander graph.

It should be possible, the four researchers conjectured in a 2016 paper, to extend this result to the realm of norms, by creating an appropriate normed space that contains an expander graph. “If you can do that, then this tangle sitting inside the norm will fool any nearest neighbor data structure,” Naor said.

The researchers looked for some normed space that contains an expander. “But all our attempts failed,” Andoni said. “In retrospect, it was for good reason.”

A General Framework

When Naor saw the four researchers’ paper, he realized he could explain why their attempts had failed. A body of mathematics he had been developing for over a decade for other reasons had the power to settle the researchers’ conjecture—but with the opposite answer from what they had expected. Naor wrote a paper proving that expanders cannot be embedded in any reasonable way in normed spaces (except for trivial embeddings that do not create obstacles to efficient nearest neighbor search).

His paper, Naor said, “revived hope that maybe the community was wrong all this time” about whether there could be an efficient general method for nearest neighbor search that would cover all normed spaces. Naor teamed up with the other four researchers, and they came up with

a method for organizing datasets inside any normed space so that finding an approximate nearest neighbor is a quick process.

Roughly speaking, their method starts by converting the dataset into a graph, by connecting every close pair of points with an edge. Naor's result tells us this graph cannot be an expander, so one of two things must be true: either the graph contains some large cluster of highly-connected points, or it can be partitioned into two sets with few edges crossing from one set to the other.

In the former case, since the cluster points are all close to each other, you can collapse them all down to a single point without losing much information. In the latter case, the partition gives you a way to greatly narrow down your search. Either way, you've simplified the set of points you need to consider. Next, you repeat this process on the smaller graphs that you obtained by collapsing and partitioning, and keep doing so until you've collapsed and partitioned the dataset down to single points.

This recursive process creates a tree-like data structure. Then, when you want to find the (approximate) nearest neighbor for a given point, you can simply follow the tree downward to quickly obtain your answer.

"Now we have a method saying that whatever issue humanity may be faced with in the future, that uses some new norm I can't even imagine, we have an off-the-shelf way to create a nearest neighbor data structure," Naor said.

Significant further work will be needed to make the algorithm practical—improving the approximation factor, for instance, and making the tree structure faster to construct. Researchers also plan to examine whether the result can be extended to some distance functions that are not norms, such as the "edit distance," which measures how many insertions, deletions, and substitutions it takes to convert one string of DNA into another.

Already, though, the new algorithm offers a path forward for norms that had no fast nearest neighbor method, such as the "nuclear" norm on matrices, which played a role in

The new algorithm offers a path forward for norms that had no fast nearest neighbor method, such as the "nuclear" norm on matrices.

the Netflix Prize competition (an open competition approximately a decade ago to find the best algorithm for predicting user ratings of movies).

"This result is the first time that a general framework for high-dimensional spaces has been developed," Indyk said. "Now that we have a framework, we can do lots of things with it."

And the result opens up broader vistas of discovery about norms. "The world of norms is very complicated," Naor said. "But the main picture is, we have more structure than we thought. I'm definitely intending to use this structure for more." **C**

Further Reading

Andoni, A., Naor, A., Nikolov, A., Razenshteyn, I., and Waingarten, E. Data-dependent hashing via nonlinear spectral gaps, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, June 25-29, 2018, pp. 787-800. <https://dl.acm.org/citation.cfm?id=3188846>

Andoni, A., Naor, A., Nikolov, A., Razenshteyn, I., and Waingarten, E. Hölder Homeomorphisms and Approximate Nearest Neighbors, *Proceedings of the 2018 IEEE 59th Annual Symposium on Foundations of Computer Science*, October 7-9, 2018, pp. 159-169. <http://ieeefocs.org/FOCS-2018-Papers/pdfs/59f159.pdf>

Andoni, A., and Indyk, P. Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. *Communications of the ACM*, January 2008, Vol. 51, No. 1, pp. 117-122. <https://dl.acm.org/citation.cfm?id=1327494>

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

© 2019 ACM 0001-0782/19/7 \$15.00

ACM Member News

BRINGING TOGETHER MACHINE LEARNING, PLATFORM SECURITY



Nadarajah Asokan is a professor of computer science at Finland's Aalto University. In

September, Asokan will become Cheriton Chair at the University of Waterloo's David R. Cheriton School of Computer Science in Ontario, Canada.

Asokan received a bachelor of technology, honors, in computer science and engineering from the Indian Institute of Technology at Kharagpur, India. He then earned a master's degree in computer and information science from Syracuse University in New York, USA, and his doctorate in computer science from the University of Waterloo in Ontario, Canada.

After completing his education, Asokan spent the first 15 years of his career working in industrial research laboratories. He designed and built secure systems, first at the IBM Zurich Research Laboratory in Switzerland, and then at Nokia Research Center, which was located in Helsinki, Finland, at that time.

"Even today, the way I look at research is colored by this industry experience," Asokan says. "I want to do things that will end up being useful."

Asokan's primary research focus today is in systems security, including topics like the development and use of novel platform security features, applying cryptographic techniques to design secure protocols for distributed systems, applying machine learning techniques to security and privacy problems, and understanding and addressing the security and privacy of machine learning applications themselves.

"I see these interests coming together," Asokan explains. "If you want to make machine learning systems more secure, you will actually need to use platform security techniques. Down the line, I see this as my primary research focus."

—John Delaney

The Edge of Computational Photography

Smartphones and consumer cameras increasingly give professional photographers a run for their money.

SINCE THEIR INTRODUCTION more than a decade ago, smartphones have been equipped with cameras, allowing users to capture images and video without carrying a separate device. Thanks to the use of computational photographic technologies, which utilize algorithms to adjust photographic parameters in order to optimize them for specific situations, users with little or no photographic training can often achieve excellent results.

The boundaries of what constitutes computational photography are not clearly defined, though there is some agreement that the term refers to the use of hardware such as lenses and image sensors to capture image data, and then applying software algorithms to automatically adjust the image parameters to yield an image. Examples of computational photography technology can be found in most recent smartphones and some

Computational photography uses hardware to capture image data, and software to adjust image parameters to yield an image.

standalone cameras, including high dynamic range imaging (HDR), auto-focus (AF), image stabilization, shot bracketing, and the ability to deploy various filters, among many other features. These features allow amateur photographers to produce pictures that can, at times, rival photographs taken by professionals using significantly more expensive equipment.

One of the key computational photography techniques that has been employed across smartphones and standalone cameras is HDR, a technique that is designed to reproduce a greater dynamic range of luminosity, or brightness, than is possible with standard digital imaging or photographic techniques.

The human eye adjusts constantly to adapt to a broad range of luminance present in the environment via changes in the iris, and the brain continuously processes this data so a viewer can see in a wide range of lighting conditions. Today's complementary metal oxide semiconductor (CMOS) image sensors can capture a high dynamic range (bright and dark areas) from a single exposure, or from multiple frames of the same image taken within milliseconds of each other. By using tuned algorithms to process this information, the images are combined so a final image can display a wider dynamic range without requiring any image compres-



IMAGE BY HEDVIKA MICHNOVA

sion. Furthermore, most smartphones are now designed to allow HDR to be turned on automatically.

“The value and the benefit to the user is that [they] don’t need to turn this mode on; the software just takes care of it for them,” says Josh Haftel, principal product manager with Adobe Systems, which makes image-processing software, including Adobe Lightroom (a family of image organization and image manipulation software). Haftel observes that HDR is an example of one of the first computational photography technologies that really resonated with the public, because it provided real value to users by allowing them to produce brilliant-looking pictures without requiring any significant user decisions.

Another technology related to HDR that has been incorporated into Google’s Pixel smartphone is Night Sight. Night Sight is a feature of the Pixel Camera app that allows users to take photographs in dimly lit or dark situations, and actually makes them brighter than they are in reality, without any graininess or blurriness in the background. Before a picture is taken, the software uses motion metering to account for camera movement, the movement of objects in a scene, and the amount of light available, to decide how many exposures to take and how long these should be. Night Sight then segments the image exposure into a burst of consecutively shot frames, which are then reassembled into a single image using an algorithm trained to discount and discard the tints cast by unnatural light, thereby allowing for proper reproduction colors of objects. The software’s tone-map was adjusted to bring out the colors in a low-light image that can’t be perceived by the human eye in low-light situations. The results are hyper-real images that maintain the dark background of the surroundings, but feature more brilliant colors and detail than the human eye can process in real life.

“Google has recently been doing a great job promoting their Night Sight mechanism,” Haftel says. “That’s a big problem that customers have, which is ‘how do I take a photo at nighttime that’s neither grainy nor

blurry, [while also ensuring] I can see people’s faces?’”

Another key technology that has been deployed is autofocus (AF), which uses sophisticated pattern, color, brightness, and distance detection to understand subjects and track them. The goal of AF is to help camera sensors recognize these objects, and then adjust the camera’s focus settings automatically and quickly to allow them to track their typical movement, ensuring faster and more accurate focus tracking. “[Autofocus makes] focus easier for everything from sports to weddings to parents wanting to shoot their toddlers and kids,” says Rishi Sanyal, science editor at *Digital Photography Review*. “They’re even using machine learning to teach their AF systems to recognize faces, eyes, animals, and [objects] like trains and motorcycles.”

Computational photography can also be used to create images taken from a camera’s data sensors to produce a photo that would be impossible to capture with more conventional tools. Examples include the ability to capture multiple frames or multiple camera inputs and then fuse them into a single image, allowing for crisper or richer images in a single shot. Incorporating a synthetic zoom view that looks nearly as good as one produced via the traditional external lens used on professional cameras allows elements from both a wide shot and a telephoto shot to be combined automatically.

“You could take a photo with 100 people in the picture, but if you want to take your friend in the center of the picture and are using a telephoto sensor, her face or his face will be very clear,” explains Zack Zhou, senior director of engineering at Qualcomm, Inc.

This type of computational technology has become somewhat commonplace in the market today. “There are many smartphone OEMs that are using multiple sensors with actual multiple depth-to-field lenses to get just a few different planes of focus, up to many, many planes of focus so that you could refocus [the photo] after the fact,” says Judd Heape, senior director of product management for cameras, computer vision, and video at Qualcomm. Qualcomm supplies the Snapdragon Mobile Platform, a hardware platform that supports a



ON THE MOVE!

**ACM headquarters
is relocating effective
July 1, 2019 to:**

**1601 Broadway
10th Floor
New York, NY 10019**

wide range of computational photography techniques and technologies and is used in virtually all smartphones (except for Apple's iPhones).

Still, the best computational techniques are not yet able to outperform top professional photographers using professional-level digital single-lens reflex cameras (DSLRs, which feature larger lenses and better sensors that still yield better large-format images than consumer cameras or smartphone cameras, due to their ability to capture more light). Nevertheless, there is significant recognition that computational photography is here to stay, and will be a point of technological investment and improvement in the years to come in both smartphones and standalone camera bodies.

One example is the Light L16, a standalone multi-lens, multi-sensor camera released in July 2017. When the user shoots a picture using the Light L16, the camera captures 10 or more images simultaneously, each with a slightly different perspective of the same scene. The L16 uses algorithms to choose a combination of its 28mm, 70mm, and 150mm modules to use in each shot, depending on the level of zoom. These individual shots are then computationally fused together to create a high-resolution 52-megapixel (MP) photograph.

Though a Light representative did not wish to comment on the camera or anticipated future developments, the company's November 2018 press release indicated improvements to the L16 were imminent, such as allowing the user to adjust the aperture, or depth effect, after a photo has been captured, using images from five camera modules. The L16 is also being updated to allow video recording at 1080p resolution and 30 frames per second.

"You have companies like Light who are going out there and utilizing multiple lenses to try and overcome the idea that you can't have a long lens on a smartphone because of physics," says Adobe's Haftel, who says sensors, mirrors, and algorithms are used to mimic the look and feel of an image being taken on a high-end professional camera that has its subject in focus, and the background out of focus, known as a bouquet.

The best computational techniques are not yet able to outperform professional photographers with professional-grade camera equipment.

Some more traditional high-end standalone cameras also have incorporated computational photographic features. The Canon 5D Mark IV includes a DIGIC 6+ Image Processor, which uses a noise-processing algorithm to help keep noise at a minimum at high ISO settings, an automatic AF selection mode, and a Digital Lens Optimizer that can automatically apply a variety of aberration and diffraction corrections, as well as other corrective measures specific to the lens in use.

Meanwhile, Nikon announced in January its CoolPix B600 camera, which also includes computational photography-based features, such as its 19 scene modes; the user only needs to select the most appropriate mode for the scene, and the camera automatically applies the appropriate settings. The CoolPix B600's Creative mode offers 36 effects, designed to provide optimal combinations of exposure, contrast, and color reproduction.

High-end lens maker ZEISS introduced in 2018 the ZX1, a camera built on an Android platform using the Qualcomm Snapdragon processor, and outfitted with lots of RAM, a graphics processing unit (GPU), and a large hard drive, feature sets typically found on smartphones. The camera was slated to be available by early this year, and could be a game changer, given that unlike the Light L16, the ZX1 features a high-performance 35mm f/2 ZEISS Distagon mirrorless lens system, as well as classic dials to control aperture, shutter speed, and sensitivity, providing a traditional, comfortable way for photographers to adjust settings.

"ZEISS is the first company that we've seen do this in a rather meaningful way," Haftel says, noting that the ZX1's combination of classic features and computational photography elements allow photographers to "do the similar kinds of computational photography that you're seeing with hand-set manufacturers."

Ultimately many of the technical improvements on the horizon will be focused on incorporating multiple lenses combined with the image stacking and super high-resolution techniques, to provide a wide zoom range, Sanya says. Further, dedicated cameras, which can shoot at 20 to 60 frames per second, will allow a multitude of sophisticated image stacking possibilities in the future, given the large amount of high-quality image data that is provided by the high frame rate. Further, high-definition images that are fed into specific algorithms can also be used to help render photographs with a surprising amount of depth, such as those found in Facebook's three-dimensional photos.

"I think a lot of the OEMs are working with software partners out there that are dreaming up all kinds of computational use cases, many of which we probably haven't even thought of yet," Qualcomm's Heape says, highlighting work being done to address the processing of multiple planes of focus at once, as well as the ability to freeze a part of an image while other parts of the image remain in motion, and segmentation techniques that allow users to highlight certain parts of an image and colorize them while the rest goes to black and white or goes into a bouquet. "These are some common use cases, but I think there will be even more in the coming couple years that we haven't even really thought of yet." C

Further Reading

Computational Photography: <https://www.andplus.com/blog/what-is-computational-photography>

Explanation of Computational Photography: <https://www.youtube.com/watch?v=WsAdG6wIAaM>

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

© 2019 ACM 0001-0782/19/7 \$15.00

Protecting the 2020 Census

A new framework is being used to secure the 2020 U.S. Census from database reconstruction attacks.

IN 2020, THE people of the U.S. will stand up and be counted, according to the provisions in the U.S. Constitution that stipulate a census may take place every decade. It's a tradition dating back to 1790, when the first national census was conducted.

This tradition is turning to a newer technique to stay secure in the 21st century.

Back in 2003, researchers Irit Dinur and Kobbi Nissim of the NEC Research Institute published a paper explaining how they had identified theoretical vulnerabilities in the summary data published with confidential databases. In some cases, the researchers found, the summary data—a high-level picture of the data from individual records in a database—could be used to reconstruct the private database. That meant attackers could use the public summary of the data to reconstruct what people had disclosed privately.

On paper, these types of database reconstruction attacks presented a possible threat to confidential databases that published summary data. The U.S. Census is a prime example of such a database.

For a long time, the paper remained a warning about a theoretical threat; until the last decade, when a dramatic increase in both computer speed and the efficiency of NP-hard problem solvers turned the theoretical threat into a practical peril, according to research published by U.S. Census Bureau employees.

One of those employees, John Abowd, associate director for research and methodology at the Bureau, worked with a team to investigate whether advances in computing power could enable database reconstruction attacks on the U.S. Census.

The results were shocking.

Abowd and his team retroactively used database reconstruction tech-



niques on these public data summaries, and found they could use advanced computational power and techniques to recreate private data that was never meant to be public.

In fact, Abowd and his team found they could reconstruct all the records contained in the database with approximately 50% accuracy. When they allowed a small error in the age of an individual, the accuracy with which they could associate public data with individuals went up to 70%. And if they allowed getting *one* piece of personal information like race or age wrong, but everything else right, their reconstruction was more than 90% accurate.

“The vulnerability is not a theoretical one; it’s an actual issue. The systems being used [for the census] were vulnerable,” says Abowd.

The solution, it turns out, was just as modern as the problem.

A Modern Solution to a Modern Problem

By law, the U.S. Census Bureau is prohibited from identifying “the data furnished by any particular establishment or individual.” That is why the Census Bureau publishes summary data, or a high-level view of the sex, age, race, and other household details of Americans by state.

The main data product that comes out of the Census is Summary File 1, which constitutes the “main dissemination of census results,” says Abowd. Summary File 1 contains a lot of data that demographers use, like age, race, and ethnicity segmented by gender, as well as household composition statistics.

According to the Census Bureau, Summary File 1 “includes population and housing characteristics for the total population, population totals

for an extensive list of race ... and Hispanic or Latino groups, and population and housing characteristics for a limited list of race and Hispanic or Latino groups.”

Abowd and his team took Summary File 1 data from the 2010 Census and subjected it to a database reconstruction attack, and found they were able to uncover privately disclosed data with some accuracy.

When Abowd presented his findings to senior executives at the Census Bureau, the agency interpreted the ability to reconstruct private data as a breach of the confidentiality obligation it had under law. In that context, it was decided that action needed to be taken to correct the vulnerability before the 2020 Census.

The Bureau’s executive team discussed the issue, then made the decision to use a statistical method called differential privacy to secure the Census process.

Explains Jonathan Ullman, an assistant professor of computer science

The Bureau’s executive team made the decision to use a statistical method called differential privacy to secure the Census process.

at Northeastern University with specialties in cryptography and privacy, differential privacy is a way to prevent attackers from reconstructing databases by adding statistical “noise” to those databases.

Statistical noise refers to altering the aggregate results that come from a database like the Census, so it is more difficult to use these aggregate results to identify the original data

collected. Ullman offers an example: rather than reporting the median income of a resident of a town in the U.S. as \$66,500, you could choose a random number between \$66,000 and \$67,000 to add noise.

“Adding this noise makes it harder for someone to reconstruct the database or otherwise breach privacy by combining many statistics,” says Ullman.

Ideally, the amount of noise should be pretty small, so the statistics can still be used by researchers, thousands of whom rely on Census data for their work. After all, statisticians and researchers are “already used to thinking of their data as containing various sources of error,” such as sampling error and response bias, according to Ullman.

However, Ullman cautions, “We have to be careful about how much noise we add and how we do it,” so the data strikes the right balance between confidential and useful. Adding the right amount of noise can make it more difficult to reconstruct the data-

ACM News

A Healthy Dose of Wearables

The idea of carrying a small device to monitor your health is not new. From the 1960s television show “Star Trek,” which featured a fictional handheld scanning device called the Tricorder, to more recent smartwatches and wearable devices, portable technology that aims to improve medicine and healthcare has advanced steadily.

Wearables are reshaping medicine in significant ways. Last September, for example, Apple received approval from the U.S. Food and Drug Administration (FDA) to include medical-grade heart monitoring in its Series 4 Apple Watch. In addition, the watch detects falls, and can alert emergency responders if such an event takes place.

Manufacturers are also introducing glucose and blood pressure monitoring, breast cancer detection, and more advanced sleep monitoring and feedback into wearables. Embedded sensors, along with wireless technology and the Internet of Things, are pushing the boundaries of medicine into new frontiers.

Smartwatches and wearables can spot potential medical problems, improve patient behavior, and boost compliance. The Apple Watch is perhaps the highest-profile smart wearable device, but other manufacturers are now streaming into the market with wearables that address an array of health challenges.

For example, iSono Health has introduced a three-dimensional (3D) ultrasound system that uses a bra to detect unusual lumps and masses in breasts, and transmits data to a smartphone or tablet.

Medical device maker Omron has received FDA approval on a blood pressure monitor that looks like a smartwatch and connects to a smartphone.

“These devices change healthcare in significant ways,” says Arielle Trzcinski, a senior analyst at Forrester Research. “An individual gains greater insight into what is happening inside his or her body without having to visit a doctor and sit in an exam room.”

Connected wearables offer other advantages. One of the most significant is access to data not previously available—or feasible to collect. For instance, the Apple Heart Study, conducted by Stanford Medicine, now has more than 400,000 participants, making it the largest screening study ever conducted for atrial fibrillation.

Wearable medical devices and the data they collect may lead to different treatment approaches, and introduce insurance rates at least partially based on patient incentives. “These devices and technology in general can aid in patient care and help alleviate burnout for clinicians. But these systems must be designed so that clinicians don’t spend hours reviewing meaningless data,” says Adrienne Boissy, a neurologist and chief experience officer for Cleveland Clinic Healthcare systems.

Not surprisingly, the quality of data and whether it can be used to accurately identify and diagnose specific issues is critically important. Unlike

fitness devices, medical devices must be precise. But security and privacy concerns also exist, including who controls and owns data as it streams across devices, systems, and companies. Finally, says Trzcinski, organizations must reexamine “the dynamics of how clinicians and patients interact” and how billing and reimbursements take place.

Nevertheless, the ability to integrate these devices into people’s lives could change thinking, behavior, and actions—and lead to healthier, happier people.

Concludes Trzcinski, “We’re going to continue to see remarkable innovation. We’re going to see devices that are smaller and smarter. As the technology improves and underlying algorithms become more precise and accurate through machine learning, the chance to actually improve people’s lives will grow.”

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

base, while also leaving the data sufficiently useful for researchers.

Differential privacy can make sure you're drawing the right balance between noise in your data and the usefulness of your data. Researchers Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith presented a paper at the 2006 Theory of Cryptography Conference, "Calibrating Noise to Sensitivity in Private Data Analysis," showing how to set up a mathematical system that allows parametric control over a risk that can be quantified, while formalizing the amount of noise needed to be added to protect the data and proposing a generalized mechanism for doing so.

"It was specifically designed to provide mathematical assurances that you had controlled the risk of database reconstruction, specifically that you controlled the potential harm from re-identification caused by an attacker building too accurate an external image of your data," says Abowd.

This is why differential privacy was picked by the Census Bureau to defend its data.

"It's a mathematical framework for understanding what 'ensuring privacy' means," says Ullman. "The framework was specifically tailored to understanding how to protect privacy in statistical analysis of large datasets, which is exactly the problem the Census faces."

Abowd began experimenting with differential privacy frameworks in 2008 as part of other work for the Census Bureau, which produces a number of data products aside from the Census itself. However, it wasn't until 2016, after he conducted a database reconstruction attack on past Census data, that the need to use differential privacy on all Census data became apparent.

Census Bureau management agreed with Abowd that differential privacy was the solution to the problem, so Abowd and a team of computer scientists and engineers got to work implementing it.

Balancing Privacy with Usability

Abowd put together a team of computer scientists and engineers in short order to combat the threat. The team includes science lead Dan Ki-

"The framework was specifically tailored to understanding how to protect privacy in statistical analysis of large databases, which is exactly the problem the Census faces."

fer, a professor of computer science at Penn State University, and engineering lead Simson Garfinkel, previously a computer scientist at the National Institute of Standards and Technology (NIST). The team is currently working to apply differential privacy to the Census' upcoming efforts for 2020.

It is not an easy task.

"We have to do it fast, and we have to do it well," says Abowd. Though he readily admits the tight timeline and volume of work are heavy burdens, and these are not the only obstacles.

The community of researchers who use Census data will be dealing with data in 2020 that has a new system of protection applied to it, and not everyone is happy about that.

One outspoken critic is Steven Ruggles, Regents Professor of History and Population Studies at the University of Minnesota, and director of the Institute for Social Research and Data Innovation, which is focused on advancing "our knowledge of societies and populations across time and space, including economic and demographic behavior, health, well-being, and human-environment interactions." Ruggles regularly uses Census data in his work, and says the use of differential privacy could limit the ability of researchers to find useful insights in that data.

"The fundamental problem is loss of accuracy of the data," says Ruggles.

"In the case of tabular small-area data, noise injection will blur the results, potentially leading investigators and planners to miss patterns in the data. For example, the noise injection could lead to underestimation of residential segregation."

Ruggles also does not believe the implementation of differential privacy on U.S. Census data is even necessary. "There has never been a documented case of anyone's identity being revealed in a public-use data product, so it is a huge overreaction."

Ullman, on the other hand, sees differential privacy as the best solution available to prevent database reconstruction attacks, while still keeping the data of the Census usable.

Because the Census has an enormous dataset, Ullman says it is possible to release huge quantities of summary statistics with manageable amounts of noise. Differential privacy then quantifies how releasing additional summary statistics will increase privacy risks, making it possible to "weigh the harm to privacy against the public benefits in a sensible way."

"There is simply no competing framework right now that has the potential to offer all of these benefits," Ullman says. **C**

Further Reading

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006) Calibrating Noise to Sensitivity in Private Data Analysis. In: Halevi S., Rabin T. (eds) *Theory of Cryptography. TCC 2006. Lecture Notes in Computer Science*, vol 3876. Springer, Berlin, Heidelberg <http://bit.ly/2DbErFW>

Hansen, M.

To Reduce Privacy Risks, the Census Plans to Report Less Accurate Data, *The New York Times*, Dec. 5, 2018 <https://nyti.ms/2UITL4n>

Garfinkel, S., Abowd, J., and Martindale, C. Understanding Database Reconstruction Attacks on Public Data, *ACM Queue*, Nov. 28, 2018 <https://queue.acm.org/detail.cfm?id=3295691>

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



DOI:10.1145/3332805

Pamela Samuelson

Legally Speaking API Copyrights Revisited

Deliberating on the main arguments in recent sets of briefs filed in support of Google's U.S. Supreme Court petition.

ARE APPLICATION PROGRAM interfaces (APIs) “original expression” that copyright law protects from unauthorized implementations in computer program code? Or are they too functional to be within the scope of protection that copyright law provides to computer programs? Alternatively, should it be fair use for unauthorized persons to reimplement APIs in independently written code?

Between 1992 and 2014, federal appellate courts in the U.S. were generally in agreement that interfaces necessary to achieving compatibility among programs were unprotectable by copyright law. These rulings made it unnecessary for courts to address fair use defenses for reuses of APIs.

In a 2014 decision, the Court of Appeals for the Federal Circuit (CAFC) overturned a trial court ruling that the Java API declarations that Google used for its Android platform were not protectable by copyright law. The CAFC ruled that these declarations were indeed original expressions that copyright should and did protect because

they were creative and there was more than one way to name particular Java command terms (for example, `Arith.` larger instead of `Math.max`).

In its 2014 decision, the CAFC did not decide that Google's use of the Java API declarations in Android necessarily infringed. Because Google had raised a fair use defense to Oracle's claim of infringement, the CAFC sent the case back to the lower court for a trial on that defense.

After a two-week jury trial in 2016, Google's fair use defense prevailed. However, Oracle once again appealed to the CAFC, arguing that no reasonable jury could have found Google's use of the Java declarations to be fair and non-infringing. The CAFC agreed and remanded the case to the lower court to consider Oracle's damage claims.

To forestall a damages trial, Google has asked the U.S. Supreme Court to review both the CAFC's copyrightability and fair use rulings. This is not its first such request. Google previously sought Supreme Court review of the CAFC's copyrightability ruling, but the Court decided against hearing this appeal, perhaps because the fair use issue had yet to be considered.

Fifteen amicus curiae (friend of the court) briefs were filed in February 2019 in support of Google's Supreme Court petition. This column discusses the main arguments developed in three sets of these briefs: one filed on behalf of 78 computer scientists, two filed by software companies, and one filed by 65 intellectual property (IP) scholars.

Each brief sought to provide a unique perspective that would help the Supreme Court understand the negative implications for the software industry if the Court does not overturn the CAFC's copyrightability and/or fair use rulings. After Oracle files its opposition brief, the Supreme Court will decide whether to grant Google's petition.

The Computer Scientists' Amicus Brief

Among the 78 computer scientists who joined this brief were 12 ACM Turing Award recipients, 24 ACM Fellows, 11 IEEE Fellows, 14 American Academy of Arts and Sciences Fellows, six National Academy of Sciences Members, 24 National Academy of Engineering Members, and five National Medal of



Technology recipients, as well as professors from numerous universities.

The main goal of this brief was to explain in non-technical terms the significance of the computer science distinction between interfaces and implementations. The brief asserts there is a long-standing consensus among computing professionals that interfaces were and should be free for all to use as long as programmers reimplement the interfaces in independently written code. The CAFC failed to grasp this distinction in its copyrightability ruling. Instead, the CAFC characterized the Java API declarations as source code, which it thought Google had literally copied in Android.

The brief also explained various factors that constrain design choices about API command names. “While software interface designers have some choice for naming methods and inputs, the method’s function, word length, and clarity constrain their choice. Particularly for programming language interfaces, which define the most basic commands used across programs, there are few practical options for nam-

ing declarations that satisfy these constraints.” By contrast, there are many different ways to reimplement interfaces in computer program code to carry out the identified functions.

The scientists’ brief also criticized the CAFC for rebuffing Google’s argument that its reuse of the Java API declarations had fostered compatibility. In their view, Google’s use of the declarations had “empowered software developers to write Java programs that run equally well on both desktops and smartphones.” The brief concluded the Court should take the appeal to correct the erroneous analysis on which the CAFC based its copyrightability ruling.

Software Company and Related Amici Briefs

Red Hat, Mozilla, and Python Software were lead amici in three briefs filed by software companies in support of Google’s petition on the copyrightability issue. Red Hat, for instance, characterized the CAFC’s copyrightability ruling as having upset settled expectations in the software industry and as posing a deep threat to the open source

software industry, which depends on freedom to reimplement APIs to create interoperable programs.

The Red Hat brief pointed to prior legal decisions establishing a bright line rule that APIs necessary for compatibility are not protectable by copyright law. Such a rule is preferable to reliance on fair use defenses, which create legal uncertainty and thereby impede innovation.

Microsoft filed a brief that heavily criticized the CAFC’s fair use ruling. It viewed that court’s framework for engaging in fair use analysis to be rigid, unduly narrow in its focus, and misguided. The CAFC had failed to give due consideration to the highly functional nature of software in software fair use cases and to the transformative character of many software reuses.

Microsoft pointed to prior cases that had “accommodate[d] the practical need for third parties to access and reuse functional code—like the software interfaces at issue [in *Oracle*—to ensure the availability of programmers and to facilitate interoperability across myriad software platforms and hardware devices.” Microsoft also asserted the software industry

needs a flexible, robust, and balanced framework for fair use analysis, which the CAFC had failed to provide.

Several other organizations also filed amicus briefs in support of Google's petition. They included the Computer and Communications Industry Association, the Developers Alliance, the American Antitrust Institute, the Electronic Frontier Foundation, R Street Institute, Engine Advocacy, and a group of software innovators, startups, and investors. These briefs tended to emphasize the exceptional importance of the case to explain why the Court should hear Google's appeal and the risks to fair competition and ongoing innovation in the software industry if the CAFC is not overruled.

IP Scholar Amicus Briefs

Two of the three IP scholar briefs focused on the CAFC's copyrightability ruling, while a third focused on the CAFC's fair use ruling.

For the most part, these briefs do not directly address the merits of the CAFC's *Oracle* decisions or the analyses on which the court relied (although it doesn't take a genius to tell which litigant's position on the merits the briefs' signatories generally favored).

There is an important strategic reason for eliding the merits at this stage of Google's appeal. The Supreme Court mainly decides to review federal appellate court decisions when persuaded there is a split among federal circuit court interpretations of federal law. U.S. copyright law is supposed to be uniformly interpreted and applied throughout the nation.

So if the Third and Federal Circuits take one position on the protectability of program APIs and several other circuit courts take the opposite position, that constitutes a split among the circuits. Circuit splits make federal law into a disharmonious mess.

A second reason the Court sometimes takes appeals is that a lower court ruling is inconsistent with the Court's previous rulings on the same or similar issues. So the IP scholar briefs also focus on the *Oracle* decisions' misapplications of prior Supreme Court rulings.

These two considerations explain why an amicus brief on behalf of 65

IP scholars, of which I was a principal author, concentrates on respects in which the CAFC's copyrightability ruling is inconsistent with a key Supreme Court precedent as well as with rulings by several other circuit courts.

Here is an excerpt from our brief to give you a sense for this type of argumentation: "Beyond merger, the Federal Circuit's interpretation of the scope of copyright protection available to software innovations conflicts with the rulings of other circuits in four respects. First, the Federal Circuit's interpretation of the exclusion of methods and systems from copyright's scope under 17 U.S.C. § 102(b) is contrary to the First Circuit's interpretation in *Lotus Development Corp. v. Borland International, Inc.*, 49 F.3d 807 (1st Cir. 1995), *aff'd by an equally divided Court*, 516 U.S. 233 (1996).

"Second, several circuit courts have ruled in favor of compatibility defenses in software copyright cases. Only the Third and Federal Circuits have rejected them.

"Third, the Federal Circuit's conception of 'structure, sequence, and organization' (SSO) of programs as protectable expression as long as it embodies a modicum of creativity conflicts with the Second Circuit's landmark decision, *Computer Associates International, Inc. v. Altai, Inc.*, 982 F.2d 693 (2d Cir. 1992). *Altai* rejected the conception of SSO as determinative of protectable expression. *Id.* at 706.

"Fourth, the Federal Circuit's assertion that copyright and utility patents can provide overlapping protection to program SSO is in conflict with [the Supreme Court's] *Baker* [*v. Selden* decision] as well as Tenth and Eleventh Circuit decisions."

We IP scholars will have a lot to say on the merits if we get to that. But the main goal of our brief is to persuade the Court to grant Google's petition because of conflicts with Supreme Court precedents and splits among circuit court rulings on software copyright issues.

Conclusion

Google is fighting a steep uphill battle to persuade the Supreme Court to review the CAFC's *Oracle* copyrightability and fair use decisions. Every year the Court selects only about 80 cases to review out of the 7,000–8,000

petitions that ask the Court to take their appeals. Thus, the odds are heavily stacked against the granting of any petition.

Two things seem to tip in favor of the Court's grant of Google's petition. First, there genuinely are split decisions among federal circuit courts on important scope of software copyright issues. Indeed, the Court recognized such a split in 1995 when it granted *Lotus*' petition for review of the First Circuit's *Borland* decision. Because the Court itself split 4-4 in that case, the First Circuit's decision was affirmed, but without setting a precedent or resolving the split. Unsurprisingly, the circuit splits on software copyright protections have deepened since then.

Second, the specific software copyright issues presented in the *Oracle* case are of exceptional importance to the software industry, of which Oracle and Google are two giants. Other major software developers, including Red Hat and Microsoft, and industry associations filed supportive amicus briefs to stress the important consequences at stake in this case.

If the Court does decide to hear the *Oracle* case, it is, however, unlikely to review both the copyrightability and fair use rulings. Most amici, including me, who support Google's petition would prefer the Court took the copyrightability issue rather than the fair use issue. But the fair use decision is also deeply flawed. So a grant on either issue would be much better for the software industry than a denial of review.

One other interesting data point is that since 2007, the Court has reversed whatever circuit court ruling it reviewed 70% of the time. That does not mean Google will definitely prevail before the Court, but its odds of winning improve if the Court takes the *Oracle* case on the merits. One bit of possibly good news for Google is that the Court has asked the Solicitor General's opinion about whether it should take the case. That somewhat increases the chance that the Court will indeed hear Google's appeal. 

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

Copyright held by author.

Computing Ethics

Who Benefits?

Considering the case of smart cities.

I AM THE INCOMING editor of the *Communications Computing Ethics* column. I appreciate what previous column editors have done since this column's inception in 2008 and intend to follow their lead, creating a space where computing professionals can raise good questions about ethics emerging from our work. This does not guarantee good answers but should elicit good discussions, which are always encouraged in the pages of this magazine.

For my inaugural column, I begin with perhaps the oldest ethics-related question of all: *Cui bono*, which means “who benefits?” People are known to be self-interested, out to improve their own welfare. The larger society sets ethical boundaries on improving one's welfare. Forbidden are theft, fraud, nepotism, bribery, violence, and a host of other behaviors. Asking *cui bono* starts us down the path to ethical issues. This column uses the case of smart cities to illustrate the ethical dilemmas created by an otherwise innocuous-seeming issue.

Smart Cities For Whom?

Ethical behavior is a professional requirement. ACM says computing professionals should contribute to society and human well-being (ACM Code of Ethics and Professional Conduct General Ethical Principle 1.1). It is hoped that incentives do the job by giving computing professionals the means: by putting them on the cutting edge, by recognizing their innovations, by helping them to improve the lives of millions with technology.



Smart cities seem a good case in point. Smart cities use cutting-edge and innovative technology to improve livability for millions.^a They promise:

- ▶ Traffic cameras and sensors that automatically adjust traffic-light timing and toll collection to reduce congestion while conserving fuel;
- ▶ Smart buildings with cameras and sensors that determine occupancy and adjust HVAC (heating, ventilation, air conditioning) and lighting to conserve energy;
- ▶ Wi-Fi kiosks with local maps and

points of interest to improve way-finding;

- ▶ Compacting solar-powered, trash cans that signal when full to reduce collection, odors, and vermin; and
- ▶ Self-driving vehicles to reduce congestion, parking space shortages, and fuel consumption.

Cui bono? In principle, everyone. But a closer look at the smart cities rhetoric shows the benefits focused on a subset of the total. Computing professionals produce hardware that is smaller, more powerful, and more energy efficient, as well as new and improved algorithms for data collection, handling, manipulation, analysis, and presentation. Tech companies benefit from new technology and expanded markets. Municipal officials benefit from their reputation for innovation.

^a Recent examples of smart cities in *Communications* include: “Building a Smart City: Lessons from Barcelona”: <https://bit.ly/2vUVCbZ>; “The New Smart Cities”: <https://bit.ly/2LGqDv6>; and “Smart Cities: Concepts, Architectures, Research Opportunities”: <https://bit.ly/2Jimz20>.

Property owners benefit from higher property values. Benefits abound for computing professionals, the tech industry, some municipal officials, and property owners. Yet the concept of smart cities is decontextualized and abstracted. None of these objectives is unethical, *per se*, but they do not touch on the interests of people who live in cities who are not computing professionals, tech companies, municipal officials, or property owners. Nor do they touch on the interests of rural residents (20% of the U.S.). One might say that is the way of the world, but the ACM Code of Ethics and Professional Conduct General Ethical Principle 1.1 also states, “When the interests of multiple groups conflict, the needs of those less advantaged should be given increased attention and priority.”

West Baltimore includes poor neighborhoods. People in West Baltimore have expressed interest in smart cities. The interests of the less-advantaged conflicted with other interests. The less-advantaged do not believe that automatic traffic-light adjustments and smart buildings are a big win for them. Their neighborhoods have little congestion and do not need automatic traffic lights. Their aging housing stock needs much more basic attention than that provided by smart buildings. Their high-crime streets do not invite people to access Wi-Fi at a public kiosk. Compacting trashcans are outweighed by loss of trash-collector jobs, which they need. Self-driving cars are more attractive to people who *have* cars, and self-driving buses are ridiculous to people who hope to get a job as a bus driver and who depend on human drivers to keep order and deter crime aboard the bus.

Many who live in West Baltimore would not benefit from smart cities as currently conceived. In fact, they might lose. But that does not mean they cannot benefit from innovation. Many in West Baltimore want free Wi-Fi on buses where they spend hours each day riding to and from school and work.⁵ These residents would benefit from free or affordable Wi-Fi at home (and no fear of unexpected data charges) to do homework, apply for jobs, improve skills, pay bills, and otherwise participate in modern life. That way those who now allow neighbors to congregate outside of their apartments

to share their Wi-Fi service would not have to ask their neighbors to disconnect when the service gets too slow. Many would benefit from video feeds from cameras and microphones that detect gunfire, with emphasis on community empowerment rather than surveillance. Are these smart-city objectives? They are difficult to find in promotional materials for smart cities.

This mismatch is not new. There should be no shock at it, as though it first arose with smart cities. In fact, it is nearly as old as the *cui bono* question, and we have had more than half a century to consider it. In 1963, the author James Baldwin famously called urban renewal “negro removal.”^b Jane Jacobs pointed out that early freeways were optimized for cars, and destroyed vibrant, resilient neighborhoods while creating dangerous downtowns that were deserted after 6 P.M.⁶ Housing projects intended to help the poor often concentrated poverty and limited social capital and upward mobility. The inner city is not a machine to engineer for maximum efficiency. It has always been difficult to bring technology to cities¹² and to rural areas.¹⁰ Will smart cities repeat the failures of the past?

Doing Better

Urban areas are complicated and sociotechnical. They are inherently emergent, never pinned down. They remind us of the importance of place.^{4,9} The strategic management of place focuses overwhelmingly on economic competitiveness, with livability taking a back seat.¹ Urban areas bring together residents who have only their place in common. Urban boundaries are amorphous and often dynamic. They include parts

b See <https://bit.ly/2vXd8MC>

It can be difficult to evaluate costs and benefits of smart cities technologies.

or all of multiple cities, counties, and even states. Management is difficult in any place that porously encompasses residents, workers who commute in from other places, and temporary visitors.⁶ Yet urban planning is traditionally top down, with urban planners treated as apolitical technocrats who emphasize efficiency.³ New technologies in municipalities are often driven by municipal and tech company leaders without participation, oversight, or influence from residents.⁸ Urban informatics concerns data collection, manipulation, analysis, and presentation.⁷ Smart cities means collection of data to supply models that drive policymakers’ decisions. Most urban residents are not policymakers.

Civic engagement through town hall meetings and public comment periods is now augmented by social media, municipal websites, email, texting, Twitter, and so forth.² Sometimes these help, but urban and rural residents of all ethnicities and social classes are still disconcerted by unanticipated policy changes and they distrust both computer models and government officials enough to mount active campaigns of resistance.¹¹ Once again, *cui bono*? Policies that work for vendors may not work for residents. Policies that work for urban residents may not work for rural residents. Policies that work for one class of residents might not work for another. Civic engagement varies by race, culture, geography, and socioeconomic status. Poor residents who work multiple jobs, rely on buses, need childcare, and have poor connectivity might participate less. The diluted voices of far-flung rural residents are faint. Urban informatics that is successful only if people stay in touch with planners means that only those who stay in touch benefit.⁷

Smart cities projects need to continually and meaningfully connect with diverse residents. Computing professionals must understand the lived experiences of city and rural residents, asking hard questions. What decisions created the situation we are trying to improve? How did those decisions affect people’s lives and opportunities? What constraints and incentives influenced these decisions? Do the processes current decision makers follow repeat the mistakes of the past? Most

important, have we heard from all the relevant stakeholders about this? Do stakeholder interests align or conflict?

Mechanisms to coordinate across multiple cities, counties, and states are even more difficult than coordinating across silos in a single jurisdiction. Municipal budgets are tight and resource allocation fraught. Urban and rural officials and experts often lack technical skills. It can be difficult to evaluate costs and benefits of smart cities technologies. Computing professionals can help, but to understand who benefits they must look beyond the limited points of view of municipal officials and experts.

Ethical Dilemmas

Smart cities must be livable cities—or else what is the point? Enabling multiple stakeholders to participate in smart cities discussions is a challenge. The disadvantaged and rural residents are often excluded and difficult to bring in, but addressing their needs spurs innovation and magnifies impact. Bringing them in can create inefficiency and slow down the process, making practicality a key challenge that must be overcome. The ACM Code of Ethics and Professional Conduct obliges computer professionals to contribute to society and to human well-being, giving increased attention and priority to the less advantaged. The job is more difficult if the disadvantaged are excluded and the privileged included. And smart cities are just the beginning. The dream must be extended to rural places if technology is to improve quality of life and support the public good. Details may vary, but rural residents face many of the same challenges as those in the city: connectivity to support transport, work, study, play, economic development, and sustainability.

Getting to both smart and livable cities and rural areas requires participatory strategies that empower multiple stakeholders in the integration of technologies into their communities. Higher-quality data, more democratic decision making, more equitable provision of services, and vibrant livable neighborhoods and rural areas might result. Effective engagement strategy requires working with local institutions, building trust, and co-design through which stakeholders become

Computing professionals are expected to work toward justice.

partners with computing professionals and others to design and implement smart-city technologies.

Computing professionals are expected to work toward justice. Definitions of justice vary (which is why there are courts). The ACM Code of Ethics requires attention to the needs of the disadvantaged. Equity, inclusion, and sustainability require participatory processes for technologies that reinforce the interests of people living in shared places. Co-designing enhances civic engagement and improves community resilience. It might not answer all the questions to ask, *Cui bono*? But it opens the door to justice. **C**

References

1. Audretsch, D.B. *Everything in its Place: Entrepreneurship and the Strategic Management of Cities, Regions, and States*. Oxford University Press, NY, 2015.
2. Criado, J.I., Sandoval-Almazan, R., and Gil-Garcia, J.R. Government innovation through social media. *Government Information Quarterly* 30, 4 (Apr. 2013), 319–326.
3. Dalton, L.C. Why the rational paradigm persists—The resistance of professional education and practice to alternative forms of planning. *Journal of Planning Education and Research* 5, 3 (1986), 147–153.
4. Florida, R.L. *The Rise of the Creative Class: And How It's Transforming Work, Leisure, Community, and Everyday Life*. Basic Books, New York, 2002.
5. GovEx. *First things first: Laying the foundation for a smart city*, 2018; <https://bit.ly/2NdSv1W>
6. Jacobs, J. *The Death and Life of Great American Cities*. Random House, NY, 1961.
7. Kontokosta, C.E. Urban informatics in the science and practice of planning. *Journal of Planning Education and Research*, 1–14, 2018.
8. Mattern, S. Interfacing urban intelligence. *Code and the City*, 2016, 49–60.
9. Porter, M.E. Location, competition, and economic development: Local clusters in a global economy. *Economic Development Quarterly* 14, 1 (2000), 15–34.
10. Rogers, E.M. *Diffusion of Innovations*. Free Press, New York, 1962.
11. Scharfenberg, D. Computers can solve your problem. You may not like the answer. What happened when Boston Public Schools tried for equity with an algorithm. *The Boston Globe* (Sept. 21, 2018).
12. Urban Institute. *The Struggle to Bring Technology to Cities*. The Urban Institute, Washington, D.C., 1971.

Susan J. Winter (sjwinter@umd.edu) is Associate Dean for Research in the College of Information Studies at the University of Maryland, MD, USA.

Copyright held by author.

Calendar of Events

July 2–7

IVA '19: International Conference on Intelligent Virtual Agents, Paris, France,
Sponsored: ACM/SIG,
Contact: Catherine Pelachaud,
Email: catherine.pelachaud@isir.upmc.fr

July 2–5

Mobihoc '19: The 19th ACM International Symposium on Mobile Ad Hoc Networking and Computing, Catania, Italy,
Sponsored: ACM/SIG,
Contact: Chiara Petrioli,
Email: petrioli@di.uniroma1.it

July 3–5

COMPASS '19: ACM SIGCAS Conference on Computing and Sustainable Societies, Accra, Ghana,
Sponsored: ACM/SIG,
Contact: Richard Anderson,
Email: anderson@cs.washington.edu

July 8–12

ASIA CCS '19: ACM Asia Conference on Computer and Communications Security, Auckland, New Zealand,
Sponsored: ACM/SIG,
Contact: Giovanni Russello,
Email: g.russello@auckland.ac.nz

July 13–17

GECCO '19: Genetic and Evolutionary Computation Conference, Prague, Czech Republic,
Sponsored: ACM/SIG,
Contact: Thomas Stützle,
Email: stuetzle@ulb.ac.be

July 15–17

ITiCSE '19: Innovation and Technology in Computer Science Education, Aberdeen, Scotland, U.K.,
Sponsored: ACM/SIG,
Contact: Bruce Scharlau,
Email: b.scharlau@abdn.ac.uk

► Richard Ladner, Column Editor

Broadening Participation

A New Labor Market for People with ‘Coolabilities’

How the unique perspective and enhanced strengths accompanying disabilities can benefit the workforce.

MANY PEOPLE WITH disabilities have enhanced strengths, such as the special ability of blind people to interpret sounds and touch, or the outstanding knack of people on the autism spectrum to observe detail. Not so well known is the potential of this large and untapped resource of excellence for the economy and society. We call these enhanced strengths “coolabilities.”

Powerful technologies are today ready to open the door to a new paradigm of work: instead of squeezing people into existing job slots, companies can tailor work that fits individuals’ unique skills, talents, and passions, matching them with inspiring teams and offering them a choice of meaningful tasks. This has tremendous benefits for both the employee and employer by creating a “long-tail labor market” in which diversity brings competitive advantage.

People with coolabilities can spearhead the new market for tailored jobs because they are a particularly underutilized resource.

A “People-Centered Economy” (PCE) with Tailored Jobs for All

The authors are members of the Innovation for Jobs community (i4j),^a co-founded in 2012 by David Nordfors and Vinton Cerf, co-inventor of the Internet. We champion the “People-Centered

Economy” (PCE) as covered in our recent book.^b Like the Copernican revolution, the PCE shifts the point of reference of our economy: centering it on the value of people and placing what is central today—tasks, services, products—in orbit. This changes the driving economic force from minimizing the cost of tasks to maximizing the value of people. It is a recipe for sustainability: tasks will come and go; people will remain. The purpose of things is to make people valuable, not the other way around.

The value in an economy, according to PCE, boils down to the value that its people see in each other. The economy is defined as people embedded in an ecosystem of organization that competes to make them valuable for each other. Seen this way, it is the purpose of organizations to serve people; never the other way around. Organizations will offer people two basic categories of services: the first category offers people better ways to earn their livelihoods, and the second offers them better ways to spend it.

^b See D. Nordfors and V.G. Cerf. *The People Centered Economy—The New Ecosystem for Work*. 2018; See especially these chapters: “The unbeatable people-centered economy” by David Nordfors; “COOLABILITIES—A New Language for Strengths in Disabling Conditions” by Chally Grundwag; “The birth of the inclusion ecosystem. Precision employment for people with disabilities, coolabilities and the rest of us.” by V.R. Ferose, Lorien Pratt, Sudipto Dasgupta, and Ganapathy Subramanian; “The Autism Advantage Movement” by Thorkil Sonne.

Seen through the PCE lens, the conundrum of this day and age, whether AI will create or kill jobs, becomes crystal clear, it is simply the case of a balanced economy: “Is AI-innovation being applied more to earning or to spending?” The emphasis today is on “spending,” that is, increasing demand for products and services, so the conclusion is that we need to support innovation that helps people earn. We need as much innovation for earning a living as we have for helping people spend what they earn.

Such innovation must be based on an “ecosystem for innovating jobs,” which largely remains to be developed. As people gain power in the PCE, entrepreneurs seeking revenues look for demographics of undervalued people for whom they can innovate more valuable jobs. This is the emerging role that our i4j community serves today.

Coolabilities: An Untapped Resource of Excellence

Coolabilities are a centerpiece of the PCE. A labor market is emerging in which those with coolabilities are no longer viewed as disabled and therefore “unemployable,” but are even more valuable on the labor market because of their powerful abilities. As pointed out by Burgstahler and Ladner,¹ computing fields need more people with disabilities because their expertise and perspectives spark innovation. It is not just abilities (expertise), but it is also that

^a i4j Innovation for Jobs; <https://i4j.info/>

they have a unique perspective adding to the diversity of a workforce.

Neuroscientists have identified enhanced abilities and compensatory mechanisms for a number of disabling conditions. Cross-modal plasticity is one example: a blind person's visual cortex is not useless; rather it adapts to help other senses such as hearing. Advances in technology such as the fMRI help us to gain deeper understanding of such processes. The concept of coolabilities connects these initiatives and opens a wider lens for understanding both scientific and social aspects of strengths linked to disabilities.

The nature of coolabilities is shown in the table. It shows six conditions, together with accompanying disabilities and typical coolabilities. Research into all imaginable conditions studying when, how, and why coolabilities occur is essential to provide a wider lens for understanding and utilizing human potential.

We suggest attributes of a coolability, which are *not* mutually exclusive:

► **Singular:** These are innate enhanced abilities essential to a condition. They appear, for example, in congenital cases; the wiring of the brain and behavior are present from the beginning. Evidence suggests that many people on the autism spectrum are hyper-systemizers, due to differences in brain functionality, according to Baron Cohen. For people with disabilities where sensory signals are missing, the parts of the brain that process those signals may use other signals instead. The auditory cortex has functionality for connecting and interpreting sentences, which for many deaf people adds capacity to interpret visual and vibrotactile cues. In the same way, blind people will often have enhanced spatial understanding of touch and sound.

► **Compensatory:** These are acquired abilities that occur or strengthen after a loss. Examples include increased hand strength among people in wheelchairs.

► **Contextual:** These have to do with context, environment, and framing, and arise when a perceived weakness becomes a strength. For example, a hearing deficit can be an advantage for a person when working in a noisy environment. Having a “narrow field of interests”—one of the characteristics that is often used to describe people on the

autism spectrum—may also indicate they have deep expertise in one of them.

It is important to understand that these suggested categories are not like slots, but rather provide coordinates for a multidimensional description of coolabilities. The hyper-systemizing coolability of many people on the autism spectrum is both singular and contextual, if it is understood as the disabling hypersensitivity to the environment, reframed by context. The repurposing of the brain that sharpens other senses when one sense is missing is singular, but also compensatory, because people use their heightened senses to compensate. Many deaf people will “listen” with their eyes and many blind people will “see” with their ears and hands. They have a singular talent, and they train it by making use of it.

Jobs for People with Coolabilities

The labor market has traditionally included at least a few jobs for people with coolabilities. In Thailand, for example, “blind massage” is appreciated because blind people's sensitive hands are known to have a better feel for muscles. Today the best-known coolabilities examples are of some people on the Autism spectrum, who often show exceptional talent for attention to detail and complex pattern recognition. These talents are currently appreciated by the software industry, where jobs have been created for individuals with coolabilities on the autism spectrum.

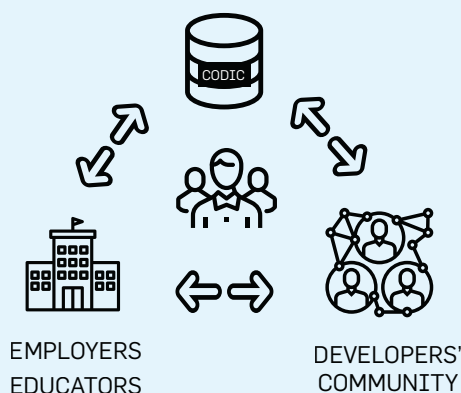
Until now, vocations for people with coolabilities have been limited, traditional, exotic, and piecemeal. In many countries of the world it is common

Coolabilities conditions.

Condition	Disabilities	Coolabilities
ASD	High sensitivity, sensory overload. Difficulties deciphering social cues.	Enhanced observation and systemizing abilities, attention to detail, focus, memory, honesty.
ADHD	Hyperactive; distractible; restless; impulsive; decreased inhibition.	Enhanced imagination; creativity, hyperfocus, flow, multi-tasking, lateral thinking, original solutions.
Dyslexia	Difficulties in reading, spelling, and decoding written language.	Enhanced visual-spatial system thinking ('flow-charts') innovative, perseverance, motivation.
Deaf	Complete or profound hearing loss.	Enhanced visual, tactile abilities, boosted by the auditory cortex: 'hearing' body language and vibrotactile abilities, and so forth.
Blind	Complete or profound visual impairment.	Enhanced auditory, tactile, olfactory abilities, boosted by the visual cortex: spatial hearing, echo-location, hands that “see,” and so forth.
Williams Syndrome	Developmental delays, physical health issues.	Enhanced empathy, friendly, helpful, verbal, musical, enhanced memory.

Coolabilities.ai overview.

COOLABILITIES — DISABILITIES CORRELATION DATABASE



to see people with certain disabilities placed, as stereotypes, into certain professions. But coolabilities do not limit the talent to any specific professions, rather point out to a general phenomenon, and a world of strength and talent that waits to be discovered. We are in the early days, yet worldwide coolabilities initiatives are beginning to coalesce and show synergies. The multidimensional description of coolabilities we suggest replaces labels and slots with a coordinate system spanning a limitless space of understanding, opportunity, and action.

Coolabilities at SAP Labs Latin America

Coolabilities are currently being implemented at SAP Labs Latin America. Brazil legally requires that between 2% and 5% of every company's permanent workforce to be people with disabilities (PWD).^c The government levies penalties against companies that do not comply. SAP Brazil has 1,950 permanent employees, and 97 are required to be PWD. To ensure compliance, the SAP Managing Director (MD) John Dennison has recently on-boarded 15 PWD as apprentices in a joint effort with the SENAC trade association. His office has conducted interviews designed to identify these employees' coolabilities. The next step will be to map coolabilities to open positions, ensuring the best fit. Says Dennison, "we don't want to just ensure compliance, we want every person to have a meaningful career at SAP."

Requirements like these are spreading worldwide: France regulates disabled hiring at 6%, Chile 1%, and Colombia provides benefits to companies hiring people with disabilities. Success in Brazil means the same can be replicated to other SAP locations with similar requirements.

But the strengths of people with disabilities must be proven beyond merely a regulatory compliance: their strengths need to be measured systematically to enhance the coolabilities database, which we describe next, and to ensure organizations make decisions effectively, providing the desired business outcomes at scale.

Companies like Specialisterne, founded by Thorkil Sonne (also found-

er of the "autism advantage" movement) have taken great steps in this direction.^d However, employment of people with disabilities in the commercial market today is largely dependent on donations from foundations; it needs more commercial investment from growth-oriented investors. Hundreds of open positions at large companies wait to be occupied by people with coolabilities. It is an existing demand waiting to be satisfied. What remains is to bootstrap the market.

Coolabilities.ai: Bootstrapping the Ecosystem for Innovating Jobs

According to the 2011 World Report on Disability,³ one billion people, or 15% of the world population, experience some form of disability, making them the world's largest "minority"—a demographic suffering an unemployment rate up to 80% in some countries. In the U.S., approximately 35% of working-age people with disabilities are working, less than half the number of people without disabilities. With nearly one-fifth of the U.S. population diagnosed with some form of disability (according to census), exploring ways to employ coolabilities has enormous potential for the U.S. alone.

Our i4j project "coolabilities.ai" has a plan for bootstrapping an ecosystem for innovating jobs for people with coolabilities and is designed to be applicable for all job specializations, not just those for the disabled. Our community includes engineers, experts, and enthusiasts who care about turning a perceived problem into an opportunity.

The key common resource of the "coolabilities.ai" platform is the "CODIC," the COolabilities-DISabilities Correlation Database (see the figure). It supports an API² that maps correlations between conditions and traits—both weaknesses (disabilities) and strengths (coolabilities). It can be crowdsourced in a manner similar to Wikipedia and personalized job matching can be provided by a deep learning/decision intelligence system. Such a peer community will be required because the task will involve making judgments and developing policies. It can use a multitude of sources, from peer reviewed research

papers to surveys and other narratives suggesting correlations.

CODIC enables developers to design applications. One example is a "coolabilities finder" where people can search for their enhanced strengths, helped by what they know about their conditions, strengths, history, needs, weaknesses, and more.

Imagine, for example, an entrepreneur with an interest in the potential of people with dyslexia. Affecting approximately one-tenth of the population, dyslexia creates a substantial drop-out risk for high-potential individuals. Many people with dyslexia have a number of strong abilities that are useful for design jobs, leadership roles, and other positions. Many do not have a college education, because such education generally requires strong reading and writing skills. This may be one reason why people with dyslexia often find themselves as independent entrepreneurs.

The entrepreneur can use the CODIC to research how many people are dyslexic, their correlating strengths and weaknesses, and to learn stories of successful dyslexia coolabilities advantages.

By targeting dyslexics using a "coolability finder," the developers' community can form an ecosystem for designing applications that link talent with opportunity. We already know of numerous positions for people with disabilities, such as the job openings at SAP for people from the ASD community. Members of coolabilities.ai can leverage this business opportunity by creating the solutions needed for their employment where corporations lack their own recruitment and training pipelines. **□**

References

1. Burgstahler, S.E. and Ladner, R.E. Increasing the participation of persons with disabilities in computing fields. *IEEE Computer* 40, 5 (May 2007); DOI 10.1109/MC.2007.175
2. Nordfors, D. et al. Coolabilities API. In *Proceedings of The IEEE International Conference on Data Science and Advanced Analytics (IEEE DSAA)*, 2018; <https://bit.ly/30lwtFq>
3. World Bank and WHO. World report on disability; <https://bit.ly/2JMWwRQ>.

David Nordfors (david@iij.org) is CEO and co-founder of i4j Innovation for Jobs, Palo Alto, CA, USA.

Chally Grundwag (chally@iij.org) spearheads the development of the coolabilities concept, Palo Alto, CA, USA.

V.R. Feroze (feroze.v.r@sap.com) leads i4j's coolabilities.ai project and is Senior Vice President and Head of Globalization Services at SAP, Palo Alto, CA, USA.

c <https://bit.ly/2Jo8qk2>

d <https://bit.ly/2FVhQyp>

Copyright held by authors.

Viewpoint

GOTO Rankings Considered Helpful*

Seeking to improve rankings by utilizing more objective data and meaningful metrics.

RANKINGS ARE A fact of life. Whether or not one likes them (a previous *Communications* editorial argued we should eschew rankings altogether⁴), they exist and are influential. Within academia, and in computer science in particular, rankings not only capture our attention but also widely influence people who have a limited understanding of computing science research, including prospective students, university administrators, and policy-makers. In short, rankings matter.

Today, academic departments are mostly ranked by for-profit enterprises. The people doing the ranking are not computer scientists, and typically have very little understanding of our field. For example, *U.S. News and World Report*, in ranking Ph.D. programs in sub-areas of computer science inaccurately describes the characteristics of research in the area of “Programming Language” [sic] (see Figure 1).

This lack of understanding of the field suggests it is highly questionable that *U.S. News and World Report* has the necessary expertise to rank the quality of Ph.D. programs across computer science. In fact, we know that many rankers often use the wrong data. For example, we have repeatedly seen problems with rankers who only consider journal publications, leaving out conferences, which capture the most influential



publications in most areas of computing. The consequences are rankings that are completely implausible. For example, while King Abdulaziz University may be a fine institution, it is unlikely that anyone with any familiarity with computing-related departments would rank the university number five in the world, as *U.S. News and World Report* does in its ranking of “Best Global Universities” (see Figure 2).

Another key limitation of a number of rankings, including those produced

by *U.S. News and World Report* and *The Times Higher Education* rankings, is that they depend in whole or in part on reputation surveys. One problem with reputation is that it is a lagging indicator. When an institution improves, it can take years for its reputation to catch up. Reputation surveys therefore are inherently “stale.” A more serious problem with reputation surveys is that opinions are often based on subjective assessments with very little basis in objective data.

* The title of this Viewpoint and (re-ordered) bullets herein are in homage to Edsger Dijkstra’s famous 1968 letter to *Communications*.²

Figure 1. U.S. News and World Report inaccurate research area description (<https://www.usnews.com/best-graduate-schools/top-science-schools/computer-programming-rankings>, May 2018).

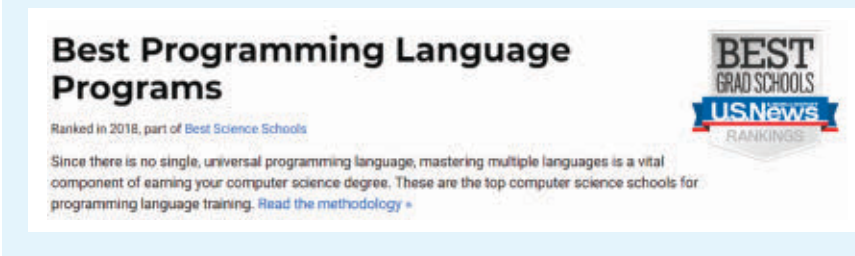
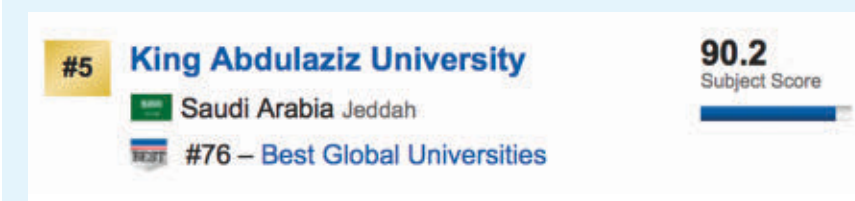


Figure 2. U.S. News and World Report implausible ranking (<https://www.usnews.com/education/best-global-universities/computer-science>, May 2018).



No one is sufficiently knowledgeable about all aspects of computer science and all departments to even make an informed *guess* about the broad range of work in an entire department. In fact, a “mid-rank” department is often the most difficult to assess by reputation because the department may be particularly strong in some sub-areas but weaker in others, that is, the subjective rating of the department may vary greatly depending on the sub-area of the assessor.

To summarize, rankings matter and will not go away, regardless of their shortcomings. Commercial rankers today do a poor job of ranking computer science departments. Since we understand our community and what matters, we should take control of the ranking process.

At the very least, we as a community should insist on rankings derived from objective data, whether it be based on publications, citations, honors, funding, or other criteria. We should ensure rankings are well-founded, based on meaningful metrics, even if we have diverging perspectives on how best to fold the data into a scalar score or rank. We may still arrive at very different rankings, but we will have a defensible basis for comparisons.

Toward this end, the Computing Research Association (CRA) has stated that a “methodology [which] makes inferences from the wrong data without transparency” ought to be ignored.¹ It has also adopted the following statement about best practices:

“CRA believes that evaluation methodologies must be data-driven and meet at least the following criteria:

- **Good data:** have been cleaned and curated
- **Open:** data is available, regarding attributes measured, at least for verification
- **Transparent:** process and methodologies are entirely transparent
- **Objective:** based on measurable attributes”

We call rankings that meet these criteria *GOTO Rankings*. Today, there are at least two GOTO rankings: <http://csrankings.org> and <http://csmetrics.org> (both are linked from the site <http://gotorankings.org>). CSrankings is faculty-centric and based on publications at top venues, providing links to faculty home pages, Google Scholar profiles, DBLP pages, and overall publication profiles. It ranks departments by aggregating the full-time tenure-track faculty at each institution. CSMetrics is institution-focused, without regard to department structure or job designations for paper authors. It includes industrial labs and takes citations into account. It derives its rankings from the Microsoft Academic Graph,³ an open and frequently updated dataset.

These are not the only two reasonable ways to rank departments.⁵ One may disagree with the rankings these sites produce, or with their choices of weighting schemes or venue inclusion.

But one can clearly understand the basis for each and inspect all or most of the included data. These GOTO rankings are a far cry from the products of most commercial rankers.

Call to Action

We call on all CS departments and colleges to boycott reputation-based and non-transparent ranking schemes, including but not limited to *U.S. News and World Report*:

- Do not fill out their surveys. Deprive these non-GOTO rankings of air, at least for computer science.
- Do not promote or publicize the results of such ranking schemes in departmental outlets.
- Discourage university administrators from using reputation-based and non-transparent rankings.
- Encourage the use of GOTO Rankings such as CSrankings and CSMetrics as better alternatives. 

References

1. Davidson, S. CRA statement on *U.S. News and World Report* rankings of computer science universities, 2017; <https://bit.ly/2W8hSOj>
2. Dijkstra, E. Go to statement considered harmful. *Commun. ACM* 11, 3 (Mar. 1968), 147–148.
3. Sinha, A. et al. An overview of Microsoft Academic Service (MAS) and applications. In *Proceedings of the 24th International Conference on World Wide Web*, 2015, 243–246.
4. Vardi, M.Y. Academic rankings considered harmful! *Commun. ACM* 59, 9 (Sept. 2016).
5. Wang, K. The knowledge Web meets big scholars. In *Proceedings of the 24th International Conference on World Wide Web*, 2015, 577–578.

Emery Berger (emery@cs.umass.edu) is a Professor in the College of Information and Computer Sciences at the University of Massachusetts Amherst, Amherst, MA, USA and a Visiting Researcher at Microsoft Research, Redmond, WA, USA.

Stephen M. Blackburn (steve.blackburn@anu.edu.au) is a Professor in the Research School of Computer Science, Australia National University, Canberra, ACT, Australia.

Carla Brodley (c.brodley@northeastern.edu) is Dean of the Khoury College of Computer Sciences at Northeastern University, Boston, MA, USA.

H.V. Jagadish (jag@umich.edu) is Bernard A. Galler Collegiate Professor of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, MI, USA.

Kathryn S. McKinley (ksmckinley@google.com) is a Senior Staff Research Scientist at Google, Seattle, WA, USA.

Mario A. Nascimento (mario.nascimento@ualberta.ca) is Chair of the Department of Computing Science at the University of Alberta, Edmonton, AB, Canada.

Minjeong Shin (minjeong.shin@anu.edu.au) is a Ph.D candidate in the Research School of Computer Science, Australia National University, Canberra, ACT, Australia.

Kuansan Wang (Kuansan.Wang@microsoft.com) is Managing Director of MSR Outreach Academic Services at Microsoft Research, Redmond, WA, USA.

Lexing Xie (Lexing.xie@anu.edu.au) is a Professor in the Research School of Computer Science, Australia National University, Canberra, ACT, Australia.

Copyright held by authors.

Call for Participation

Special Section on East Asia and Oceania

Communications of the ACM's regional special sections—designed to spotlight a region of the world with the goal of introducing readers to new voices, innovations, and technological research—is calling for interested authors to participate in the Special Section on East Asia and Oceania.

This region includes Japan, Korea, Taiwan, South East Asia (Singapore, Malaysia, Indonesia, Brunei, Vietnam, Thailand, Myanmar, Philippines, Laos, Cambodia), and Oceania (Australia, New Zealand, Papua New Guinea, Fiji, Melanesia, Polynesia, Micronesia).

Articles can cover any aspect or field of computing, including research, education, or innovation. Submissions are sought from academia, industry, and government agencies throughout the region.

Potential areas of interest include:

- **Technical and governance challenges that can benefit from data science and AI;**
- **Key advances in cybersecurity and privacy;**
- **Key technological advances to aid healthcare and the aging population;**
- **Technology to advance sustainable social and environmental development;**
- **Key technological advances to aid and enhance education;**
- **Regional advances in supercomputing; and**
- **Regional advances in consumer electronics, embedded systems, and robotics.**

Interested authors are requested to send a one-page, free-format PDF file with title, full list of author names, and description of the planned article to:

cacm20@comp.nus.edu.sg
by **15th July 2019**.

The types of articles chosen for inclusion will be designated as either Hot Topics (800–1,200 words max) or Big Trends (1,500–1,800 words max).

Selected authors will be invited to a workshop at the National University of Singapore on Aug. 30, 2019 to participate in discussions to shape the narrative for the Special Section on East Asia and Oceania.



Association for
Computing Machinery

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

Data doesn't always fit nicely into a new home.

BY PAT HELLAND

Extract, Shoehorn, and Load

A LOT OF data is moved from system to system in an important and increasing part of the computing landscape. This is traditionally known as ETL (extract, transform, and load). While many systems are extremely good at this process, the source for the extraction and the destination for the load frequently have different representations for their data. It is common for this transformation to squeeze, truncate, or pad the data to make it fit into the target. This is really like using a shoehorn to fit into a shoe that is too small. Sometimes it's a needed step. Frequently it's a real pain!

Two major parts of ETL are the extraction and the load. These processes are where the rubber meets the participating data stores.

Extraction pulls data out of a source system. This may be relational data kept in a database. If so, it may be converted to an object relational format where each object *transforms* the join of multiple relational

rows into a cohesive thing. Data is frequently organized as messages when it is sucked out. It's also common for data to be extracted from key-value stores where it is kept in a semi-structured representation.

Load happens when the data is placed into the target system. The target will have its own metadata describing the shape and form of the data in its belly. If the target is an analytics system, then its data will likely be loaded into a relational form.

While it may be counterintuitive, it is frequently useful to take relational data out of a system as objects; convert, massage, and shoehorn the data from one object representation to another; and load it into the target system in relational form.

The new, shoehorned data is then used for analytic queries.

To make this work, you need metadata¹—for both the source being extracted and the target being loaded.

The metadata for the extracted source is *descriptive*. The data exists. The metadata describes its shape, form, and meaning. It is always the case that extracted data is copied out and the metadata describes what its shape was and what its shape is as it's extracted.

The metadata for the loaded data is *prescriptive*. The ETL system makes it fit the output metadata's shape and form exactly as it is prescribed to do. Only when the system knows what the results should look like can it do the work.

ETL systems always need to know the current shape of the input, as well as the shape the data should become.

Frequently the output data to be loaded into the target system is relational in its shape. Many analytics systems expect relational data to analyze and report. Relational is great as the format for both planned and ad hoc queries.

What If the Shoe Doesn't Fit?

When incoming data and its descriptive metadata don't match the outgo-



ing data shape specified by the prescriptive data, you must do something about it.

When a shoe is too large, people will shove padding into it. Similarly, when the incoming data doesn't have all the information required for the outgoing shape and form, you add stuff. This may be a default value or a null value.

If a shoe is too small and the foot is too large, sometimes you use a shoe-horn to force the foot into the shoe, comfort be damned. This is a real pain! Similarly, when the incoming data has too much information, the system needs to discard data that doesn't fit the outgoing metadata.

The process of discarding or padding data is very common.

All too often, the descriptive metadata for the input is not a perfect match to the prescriptive metadata for the desired output!

Sometimes data is extracted from many sources with either the same or different input metadata describing the stuff being loaded. It's essential that the data from the various sources be modified to fit into the target metadata.

Note that normalizing the data to

relational form may be difficult with different input data from different systems. The needed information may be missing from some input source.

Conclusion

ETL takes disparate sources and destinations and moves data from one to the other. Frequently there is only a partially useful mapping of the metadata. Sometimes data must be discarded to traverse the path from source to destination. Other times the source data may need to be augmented with null values or default values. It's also possible that the mapping is complex and loses much of the meaning kept in the original translation as the data is reshaped and re-formed.

Metadata for the loaded source data is *descriptive*—it describes the data. Metadata for the data loaded into the target is *prescriptive*—it prescribes the required target shape and form. The challenge is that the *described output* may be ill fitting to the *prescribed input*.

It turns out the business value of ill-fitting data is extremely high. The process of taking the input data, discarding what doesn't fit, adding default or

null values for missing stuff, and generally shoe-horning it to the prescribed shape is important. The prescribed shape is usually one that is amenable to analysis for deeper meaning.

It is the shoe-horning that gives the data the shape it needs to be understood consistently. 

Related articles on queue.acm.org

Immutability Changes Everything

Pat Helland

<https://queue.acm.org/detail.cfm?id=2884038>

Data in Flight

Julian Hyde

<https://queue.acm.org/detail.cfm?id=1667562>

Other People's Data

Stephen Petschulat

<https://queue.acm.org/detail.cfm?id=1655240>

References

1. Helland, P. If you have too much data then 'good enough' is good enough. *acmqueue* 9, 5 (2001); <https://queue.acm.org/detail.cfm?id=1988603>.

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

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

Software acumen is the new norm.

BY THOMAS A. LIMONCELLI

The Top 10 Things Executives Should Know About Software

A FRIEND OF mine is an accountant at a large company. The CEO and other executives do not know what accounting is, and that is OK. Everyone works around it.

OK, that is a lie. No company like that exists.

I do have a friend, however, who is a software engineer at a large company where the CEO and other executives do not understand software. They don't understand what is reasonable to expect software to do, how it is made, how software projects are managed, or how a Web-based service is run.

That isn't something that employees can "work around."

Maybe that was OK years ago, but it isn't OK now. In fact, my advice to this friend was to start sending out her resume.

Many companies that don't think of themselves as software companies are finding that software is a key component of their operations. If executives and management do not understand how software is made, they will be ineffective compared to those who do. This will either limit their careers or negatively affect their company's performance. Either way, they are doomed. (You don't have to take my word for it: Gartner predicted that 50% of CIOs who have not transformed their organization's capabilities by 2020 will be displaced.⁹)

In this article, I list the things that "executives who get software" understand in an effort to help those executives and managers who have found themselves in this new world. The list is not exhaustive, as the full list could fill multiple books, but it is based on a very unscientific poll of my friends in the industry.

Software Ate the World

In 2011, Marc Andreessen¹ wrote an article predicting, "Software will eat the world." By that he meant two things: First, many traditional businesses are being replaced by software companies. Second, all other companies are finding the value they deliver is increasingly a result of software.

When Andreessen wrote his article none of the 10 biggest companies (by market value) were in software-driven businesses. Today, six of the 10 biggest companies are primarily driven by software. The others are ripe for a transformation.

The first category is easy to understand; online music stores such as Apple's iTunes have probably replaced your local music store. A physical location is eliminated in favor of one solely defined by software.

The second category is a subtler change. For example, while automo-

biles have not been replaced by websites, what makes auto companies succeed is increasingly a result of their software acumen. Their supply chain, manufacturing, marketing, and sales processes are controlled by software. The typical car has 10–100 million lines of code in it. The majority of what differentiates car models comes from software-powered features such as the dashboard and audio system. Everything I love about my new car is software; everything I dislike is software that could have been better.

I recently purchased a very low-tech refrigerator. As far as I can tell, it has zero software inside of it. Not long after the purchase, however, I received an email offering a water-filter-replacement subscription. That system is entirely software driven. Considering the high price of the subscription filters, I presume they are responsible for more profit than the original purchase.

If you don't control what makes your product valuable, then you are not much of a company. Therefore, executives and managers must now understand the software-delivery lifecycle.

Both Stack Overflow Talent and LinkedIn now list more software engineering job advertisements for nontechnical companies than the tech industry itself.¹¹ This is a major shift in the economy and indicates that companies are ramping up their software practices.

The List

Here are my top 10 things I believe all executives and managers must know about software:

1. Software is not magic.

Often it looks like magic, or is magical, but it isn't magic. Every element was designed by a human and has its basis in math or a process that can be explained in human words.

Unlike magic, software isn't conjured out of thin air. It needs to be designed, built, and operated. Just as a house has layers of systems that



work together (foundation, structure, plumbing, rooms, furniture, and so on), software has layers and subsystems that create the whole. It can be designed well or badly, and a fast design is rarely a lasting one.

If you cannot describe in words what it will do (both the desired outcome and how it can get there), then a computer cannot do it. The “how” is called an *algorithm* and is not magic.

A Web search for pictures of chairs doesn't actually show pictures of chairs. It shows images that frequently appear on Web pages that mention the word *chairs*. The difference is subtle, and it took years for someone to think of that trick and perfect the technique. Yet, it isn't magic.

Your email system's spam detection looks pretty magical, but it is not magic. Bayesian statistics and other mathematical models work under the hood to achieve the behavior you see.

Autocorrect feels magical (try turning it off for a day), but the best autocorrect systems process trillions of data points from the past to create a database of precomputed autocorrections to use in the future.

Machine learning (ML) and other AI techniques are not magic. ML is prediction based on data, instead of explicit rules or instructions. It is monkey-see-monkey-do using linear algebra. An ML system is trained by showing it millions of pictures with bananas, then million of pictures without bananas. Now, if you show it a picture it will tell you if it looks like the first or second group.

Not magic. ML is enormously useful across many domains, but sometimes it can act like *The Sorcerer's Apprentice*. For example, using ML to sort resumes based on past hiring decisions can amplify a racist hiring history, even without any intended bias.²

2. Software is never “done.”

Software is an iterative process with many revisions and updates shipping over its lifetime. Your job is to create an environment that recognizes this.

Likewise, we never expected marketing and customer acquisition to be “done.” They too are iterative processes. We learn and grow with each iteration as we continue to deliver value to the business. We don't ever plan to “stop” doing any of those things, even once we find something that is a successful launch.



It would be nice if software could be finished in one release, but that is not reality. Requirement documents are full of ambiguity. The first release of software is full of “oh, that's what I wrote but it wasn't what I meant” moments. The best software inspires new ideas and new features. Seeing that the new sales-management system is more efficient inspires even more efficiencies. If you don't plan on future releases that will incorporate the best ideas of your employees, you have built a system that just solves yesterday's problems. The world changes, your competitors offer new features, people have new ideas. There are always bugs to be fixed—maybe in your

code, or in the underlying software frameworks and systems that it is built upon. Your software may be perfect, but I assure you that over time people will find security holes in the platform it is built on.

It is your job to expect an organization that recognizes this.

The way we recognize this is to build an organization that confidently produces new software releases at regular intervals. When fully automated testing and other engineering disciplines are in place, we build confidence. This confidence creates the ability to eschew multiyear release cycles and instead ship high-quality software quarterly, monthly, or even weekly. The particular frequency isn't important; confidence is. Confidence leads to faster innovation.

This also means rejecting project plans that involve one perfect release, then no more. Or plans that do not involve sufficient testing, or eliminate the beta-test period, or allow developers to make changes to live production systems instead of having an approved and tested path to release. Features that make software more shippable should not be left until the end; ease of shipping has business value.

Lastly, let's stop with software projects that are allowed to run for multiple years before showing any progress. Release early and often. Require a minimum viable product to launch, followed by periodic releases that add features. The first release might be just the basic framework or support only a few edge cases. Each release provides an opportunity to get feedback and change course. Early releases might run only in a beta area, inaccessible to real users. At least you have started the feedback cycle. Beta testing saves lives—and careers.

Of equal importance, behind-the-scenes operations now have a chance to begin developing their processes and procedures, build and vet infrastructure, and test the invisible foundation that supports everything else. Imagine if the Obamacare website had first supported only Rhode Island, then added support for states one at a time. The experience from each iteration would have propelled it forward and made it a success from the start.

3. Software is a team effort; nobody can do it all.

The software developer is neither product manager nor UX (user experience) designer nor quality assurance analyst nor security guru nor technical writer nor operations engineer. You need them all.

No executive would propose that each salesperson do his or her own marketing, or that the sales force should be fired because marketing understands the product and can do sales, too. Marketing and sales are related but different. Therefore, a division of labor exists between the two.



Likewise, software teams need separate people for requirements gathering, quality assurance and test engineering, technical writing, and so on.

There is a myth of the developer who “does it all,” known as the “full stack developer” or “10x engineer.” This doesn't exist outside of the smallest company. Yes, a very small company may have a single person who does both marketing and sales, but you probably don't work for a company that tiny. Neither do your engineers.

Yes, your 12-year-old son made a website all by himself. Don't let that make you think that it cannot be that difficult, or that coding is “just typing.” I assure you Johnny's website isn't processing billions of financial transactions per hour. When I was 10 years old I built a “robot” out of cardboard boxes. My parents were smart enough to take that as an indication that I was interested in engineering, not to think I could skip calculus.

Which reminds me: Dear Parents, Just because your child is “good at Facebook” doesn't mean he or she will be the next Zuckerberg. Stop saying that at cocktail parties. It's embarrassing. (P.S. No teens use Facebook any more. Your kids post to Facebook only

as a decoy so that you don't try to find where they actually hang out. I'm sorry you had to learn that here.)

4. Design isn't how something looks; it is how it works.

Steve Jobs famously said, “Design is not just what it looks like and feels like. Design is how it works.” UX designers don't sit around trying to decide what color the menus will be, or if buttons will be round or square. They determine what the workflow and interactions will be.

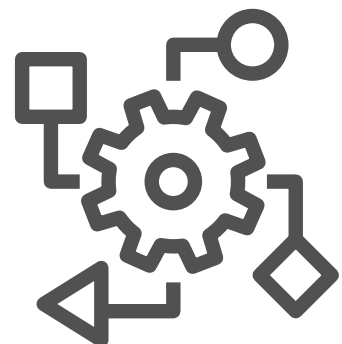
Will the user be presented with one screen with three choices, or will the choices be presented one screen at a time? The decision requires psychology, empathy for the user, and testing, testing, testing.

One of the biggest challenges of UX design is that once you know the system well, you lose the ability to predict what a new user will expect. The person who designed the system is automatically disqualified for predicting what a new user will need.

I remember the first time I had the opportunity to watch users through a one-way mirror as they used a product I was involved with. “It's the button on the left! Look to the left! Oh god, why aren't they clicking the button on the left!” Then reality sets in. Our brilliant placement of buttons was brilliant only because we knew the system so well.

Therefore, testing with real users is required. This could be as simple as recruiting a coworker who has yet to see the system, or as complex as using a double-blind study with one-way mirrors and eye-tracking systems.

I also remember watching someone test Google Maps. The user was asked to route from New York Penn Station to a particular hotel. After that task was completed, the UX designer asked, “What do you think you would want



to do now?” The person responded, “Once I’m checked in, I find a restaurant.” Soon after, Google Maps added a “find a nearby restaurant” feature. That’s a good UX designer!

A UX may be beautiful and elegant and comparable to a piece of art, but asking a UX designer to change the background to a picture of a sailboat is not helpful.

It is your job to trust testing data over opinions, to create an environment that plans for multiple revisions before the product ships and expects further refinement after.

Do not confuse a UX designer with a graphic designer. A graphic designer develops layouts to inspire and inform in a variety of media from brochures to websites. Asking a UX designer to design the company holiday card is as much of a faux pas as asking the technical writer to write the company newsletter. These are all different skills.

5. Security is everyone’s responsibility

You are in the security business whether you know it or not, and whether you want to be or not. All software has security requirements and potential security vulnerabilities. The systems involved in producing your software have security requirements and vulnerabilities, too. While security infrastructure components such as firewalls and intrusion detection are necessary, they are not sufficient: you must also design, implement, and operate your software platforms with built-in security controls. Security is as much about good process as it is good technology.

If you think you are not a target, then you are wrong. All computer systems are targets, as the prize is not just the information in them but the mere fact that it is a computer. For example, a system with no information of value is a cybersecurity target because it can be used to relay an attack on other computers, or mine bitcoin, or store someone’s pirated video library.

Security is not an on/off switch. There are many shades of gray. You don’t build a system, then press the “make it secure” button.

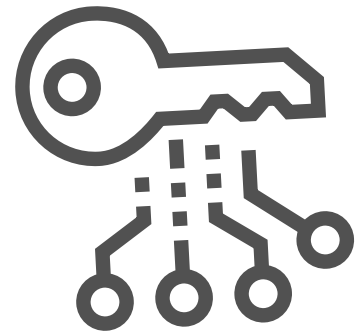
Security is about risk and your tolerance level for risk. Encrypting communication between two points doesn’t

Unlike magic, software isn’t conjured out of thin air. It needs to be designed, built, and operated ... It can be designed well or badly, and a fast design is rarely a lasting one.

make it secure, but it enhances the security such that only a superpower has the resources to crack the code. Mitigating risk in one area doesn’t help in other areas. Securing the network doesn’t prevent physical security issues. An employee propping a door open enables someone else to steal your backup tapes.

As Gene Spafford famously stated, “The only truly secure system is one that is powered off, cast in a block of concrete, and sealed in a lead-lined room with armed guards—and even then, I have my doubts.”³

Compliance with security standards such as NIST CSF (National Institute of Standards and Technology Cybersecurity Framework), PCI DSS (Payment Card Industry Data Security Standard), and SOC 2 (Service Organization Control report) quantifies risk and, when done right, reduces risk. These standards do not assure perfect security; such a thing does not exist. More importantly, they provide guidance on how to respond responsibly and report the inevitable security breach. Being honest, forthright, and public is my recommendation.



Security is best designed in from the start. Bolting it on after the fact is expensive and often ineffective. You would not build a boat and then “add in” a way for it to float.

Today the most common vector for security issues is not the sexy high-tech security hole some elite hacker discovered last night. It is the old, boring, everyone-else-fixed-it-years-ago issue that goes unnoticed. You would be stunned at how many systems are calcified and cannot be updated because updates are impossible, expensive, or unavailable. They may have been considered (relatively) secure when new, but now new vulnerabilities have been discovered. Software, left alone, grows stale like bread.

It is your job to balance security paranoia with reality, and budget time and resources appropriately.

6. Feature size doesn't predict developer time.

Feature size (as perceived by users) is entirely unrelated to how long it will take to create said feature. Small features can take days or years. Big features (as perceived by users) can take days or years.

It is your job to create and support a software development process that accepts this and does not second-guess engineering's work estimates. Producing the work estimate itself may take a surprisingly long time.



Negotiation is encouraged. The engineers may reply with a surprisingly long work estimate but offer changes to the requirements that will cut the time significantly. Remember to include time for testing, training, deployment, and unexpected family or medical leave.

Never promise a feature without consulting with engineering for a work estimate. It is not a sign of your corporate power to promise a feature by a certain deadline on the spot. I assure you that what people find more impressive is a professional process where their request is taken seriously, work estimates are produced, and the request is delivered on time (or rejected for honest reasons).

7. Greatness comes from thousands of small improvements.

Greatness comes from thousands, perhaps millions, of small improvements done over a long stretch of time. The effect of each change is measured, and the change is rolled back if the outcome is negative.

Google was not built in a day.

Google's search engine is the result of millions of individual improvements. Once a week the search-quality panel meets. Engineers step up to the podium and present their proposed changes. They show how much of an improvement would be made based on simulations. The committee debates and votes it up or down. Weeks later the measurements are reviewed, and the change is either kept or rolled back.

Google search is the triumph of iterative development over "big bang" thinking. You can't make a good search engine on your first attempt.

Only in Hollywood movies does a brilliant young mind come up with an amazing new idea that is implemented and works perfectly the first time. In the real world, it takes years to create an overnight success.

This is true whether the greatness you are trying to achieve is a system that provides better service to customers, is more efficient, has fewer errors, or just organizationally runs more smoothly.

It is your job to require systems to be designed to make it easy to try new things and to define pertinent KPIs (key performance indicators) that can easily be measured before and after changes. Most importantly, there must be a process by which the results are examined and a decision is made to keep or roll back the change. A rollback should not be considered a failure or be punished. What is learned from each rollback is as valuable as what is learned in each change that is retained.

Thomas Edison claimed to have tested 1,000 filaments on the way to creating his light bulb. When a reporter asked, "How did it feel to fail 1,000 times?" he replied, "I didn't fail 1,000 times. The light bulb was an invention with 1,000 steps."

This is another reason why software systems need to support rapid releases.



The biggest improvements come from working across silos and involving all stakeholders. If there is no collaboration across teams, then each team will optimize their area, often to the detriment of the efficiency of the other teams. By working across teams, you develop empathy and can create the most impactful changes.

I recently read about a U.S. company that stayed ahead of foreign competition through efficiency. It was able to achieve this advantage by constantly examining the end-to-end process. In one case large amounts of materials and manufacturing time were being spent on plastic covers. A major customer was removing and disposing of those covers because they got in the way. If the manufacturer had not visited the customer, it would never have realized it could improve efficiency by selling a model without that cover.

Likewise, both the process of building software and the process in which the software is used must be under constant revision brought about by end-to-end examination.

8. Technical debt is bad but unavoidable.

Technical debt is the work you will need to do in the future because you chose an easy solution now instead of using a better approach that would take longer. Any software project of reasonable size has technical debt.⁷ Technical debt makes all forward progress slower, and it snowballs the more you ignore it.

I fear that executives with a finance background hear "debt" and think it is an investment that will pay off in the future. Technical debt is the opposite. It is toxic and painful. It is a ticking time bomb. Caskey L. Dickson⁴ compares it with "naked call options," future obligations that could arise at any time, without advance notice, and having an unlimited downside.

In 1972 Fram ran a TV commercial for its oil filters in which an auto mechanic explained that a customer tried to save \$4 by not replacing a filter; later the customer had to pay \$200 for an expensive main bearing replacement. The mechanic concluded, "You can pay me now, or pay me later."⁸

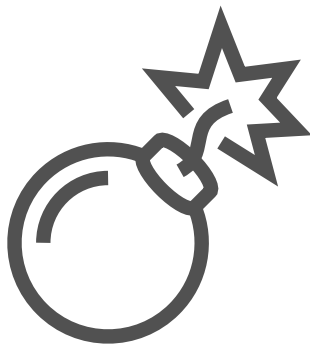
Once I was involved in a software project with a subsystem that com-

municated to a supplier. Initially the system talked to only one supplier, so that was pretty easy. Then a second was grafted on. Then another. Some features had to be implemented three times, once for each supplier. This was not sustainable.

When asked to support a fourth supplier, the developers revolted. Yes, they could graft it on in about a month, but the software was starting to creak like an old house in a hurricane. The quick fixes had accumulated technical debt.

Their proposal was to spend two months refactoring (reworking) the supplier architecture so that it was a plug-in system. New suppliers could then be added in a week, not a month.

The executives were not happy. Why would this next supplier take more than two months to add when previous suppliers were added in one month?



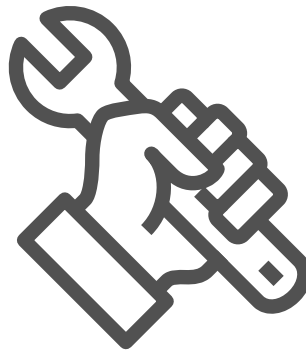
The two months invested in paying down technical debt would make future additions faster, stabilize the code base, and make it easier to add new features. It is difficult to measure the exact benefits.

You can pay me now, or pay me later.

It is your job to allocate time to pay down technical debt. Runaway technical debt slows down the ability to add other features, and it leads to unstable software. Paying down technical debt should be tied to business goals, similar to nonfunctional features.

9. Software doesn't run itself.

While vendors and developers may try to tell you otherwise, software doesn't just run itself. Any software-based system (websites and Web applications, in particular) requires operational staff and processes; otherwise, it just sits there like a closed book. Someone has to turn it on, care for it, and tend to its needs.



I assert that operations is more important than software development itself. Code is written once but runs millions of times. Therefore, operations are, by that rough measure, millions of times more important.

As a result, it is your job to expect operations to be part of any software-based system. It must be planned for, budgeted, managed, and run efficiently just like anything else.

Operational features (usually called nonfunctional features) are invisible to users except as second-order effects. Data backup is a good example of a nonfunctional feature. No user requests data to be backed up. Users do, however, ask for deleted data to be restored. Sadly, there can be no restore without a backup. A restore is a functional feature; a backup is an operational (nonfunctional) feature.

Features that make a software service easy or efficient to operate are never requested by users. They do, however, enjoy the benefits of a system that is cost effective and reliable. Customers leave unreliable websites and don't come back.

Software must be scaled, monitored, updated, and so on. Wikipedia has an excellent list of nonfunctional requirements that drive such features.¹³ Operations are in a constant battle to improve efficiency. This often requires new code.

The need for continuous improvement includes not just new features, but new nonfunctional features. Therefore, it is your job to allocate resources not only for the features that customers demand, but also for operational features. Striking a balance between the two competing needs is difficult.

A successful product is the negotiated union of business and operational requirements.

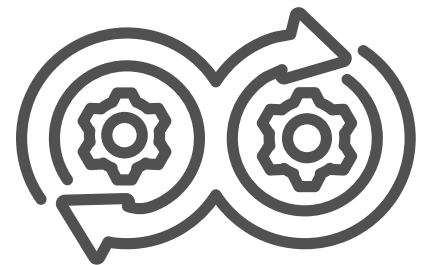
10. Complex systems need DevOps to run well.

A complex system is best improved through DevOps. This has many definitions, but I prefer to think of *DevOps* as accelerating the delivery of value (features, bug fixes, process improvements, and so on) by rapid iteration. To achieve this, everyone involved must participate. That is, they must work across silos. The name DevOps comes from the movement to remove the wall between developers and operations (IT), which is absolutely required to achieve rapid releases. Great DevOps environments, however, extend this to work across all silos end to end.

DevOps has been misinterpreted to mean developers perform operations. This "you build it, you run it" strategy is one way of working across silos (eliminating them), but it isn't the only way. More on that later.

The system that builds and continuously improves your software is a machine. Every time you turn the crank, a new (hopefully improved) release of software pops out and goes into production.

Delivering your product to customers is also a machine. Your marketing, sales, logistics, billing, and other systems all work together. Every time you turn the crank your product is delivered.



Either kind of machine is a complex system with many dependencies. To run well, a complex system needs three things: a good process, good communication by all the people involved, and the ability to try new things.

These are codified as the Three Ways of DevOps:

► *The first way is "system thinking" or "flow."* The focus here is on improving the end-to-end process, not specific silos, as described in item 7. The First Way is about driving improvements that move you from a

process that sucks to one that is awesome. In pathological cases the process is nonexistent—each silo improvising and guessing its way through the process each time the crank turns. The result of the First Way is improved velocity and reduced defects. Things work better.

► *The second way is “amplify feedback loops.”* The focus is on improving communication among the people and components within the system. Communication is a feedback loop and should be bidirectional, responsive, transparent, and blameless. A system cannot work well without the ability of the people involved to learn, share, and grow. The Second Way is about driving improvements that move you from communication that is lacking to communication that is comprehensive. In pathological cases communication is punished. The result of the Second Way is understanding, empathy, and responsiveness to customers both internal and external. Knowledge is where it is needed.

► *The third way is a “culture of continual experimentation and learning.”* This is where you focus on creating a culture where you try new things, evaluate the results, and decide whether to keep or revert the change. The Third Way is about going from a culture where change is resisted to one where change is constant. Risk is accepted. Rituals reward teams for taking risks and learning from failure. In pathological cases the organization is calcified: change isn't possible, suggestions for change are rejected or possibly punished. The result of the Third Way is evolutionary change over time, punctuated by major leaps and innovation.

Wait, there is more ...

Indeed, there are volumes more that an executive should know about software. Sadly, cultural pressure and David Letterman say I should stop at 10.

Here are some bonus items:

Bonus item 1.

Uptime is never perfect.

Asking for 100% uptime makes you look ignorant. Each order of magnitude of improvement costs ludicrously more than the level prior: 99.0% uptime is fine for plenty of systems; 99.999% is

more expensive than you can afford. Punishing people for downtime sends the wrong message. Instead, ask “What did we learn?” If your organization learned something, the downtime was a gift. Recommended reading: *Beyond Blame, Learning from Failure and Success*, by Dave Zwieback,¹⁴ and Wikipedia's page on High Availability.¹²

Bonus item 2. Spammers and abusers ruin everything.

Fighting spam and abuse is an arms race. If you can build an online app in a week, you will spend a year figuring out how to prevent spammers from ruining it. Google Sheets has anti-abuse detection because criminals make spreadsheets full of links to scams and then send the links to people who think any link that mentions Google is safe. The amount of anti-abuse work required to run online communities such as Twitter, Facebook, or other social networks would make you cry.

Bonus item 3.

Malleability is expensive.

Some changes to software require a new release, while other changes can happen while the system is running. The latter is expensive. It would be easy for Facebook profiles to store only your name, location, and a few other facts. The ability to store any field is an expensive engineering task. Be careful when asking for flexibility. It affects testing, security, usability, and a lot more.

Conclusion

Software is eating the world. To do their jobs well, executives and managers outside of technology will benefit from understanding some fundamentals of software and the software-delivery process.

Further resources. If you are an executive who wants software acumen, there are many resources. The first is your VP of engineering or CTO. Ask the person in one of these jobs what you should learn.

I also highly recommend reading *The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win*, by Gene Kim, Kevin Behr, and George Spafford.¹⁰ It provides an inside view of IT and a practical understanding of how to use DevOps techniques to manage it.

I also recommend *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High-Performing Technology Organizations*, by Nicole Forsgren, Jez Humble, and Gene Kim.⁶ It provides a CEO view of the science that makes DevOps work.

Acknowledgments.

Lynn Ballard, Nicole Forsgren, Tim Bell, Lance A. Brown, Jennifer Davis, Trent R. Hein, Mark Henderson, Steve VanDevender, Harald Wagener. 

Related articles on queue.acm.org

The Age of Corporate Open Source Enlightenment

Paul Ferris

<https://queue.acm.org/detail.cfm?id=945124>

Managing Technical Debt

Eric Allman

<https://queue.acm.org/detail.cfm?id=2168798>

Why Cloud Computing Will Never Be Free

Dave Durkee

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

References

1. Andreessen, M. Why software is eating the world. *The Wall Street J.* (Aug. 20, 2011); <https://on.wsj.com/2IDLhKk>.
2. Buranyi, S. Rise of the racist robots—how AI is learning all our worst impulses. *The Guardian* (Aug. 8, 2017); <https://bit.ly/2uBqVpk>.
3. Dewdney, A.K. Computer recreations: of worms, viruses and core war. *Scientific American* 260, 3 (1989), 110.
4. Dickson, C. L. Why your manager loves technical debt and what to do about it. In *Proceedings of the Usenix LISA Conference*, 2015; <https://www.usenix.org/conference/lisa15/conference-program/presentation/dickson>.
5. Fong-Jones, L. Twitter, 2018; <https://twitter.com/lizthegrey/status/1052636505712275458?lang=en>.
6. Forsgren, N., Humble, J., and Kim, G. *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations*. IT Revolution Press, 2018.
7. Fowler, M. Technical debt. *Martinfowler.com*, 2003; <https://martinfowler.com/bliki/TechnicalDebt.html>.
8. Fram Oil Filter commercial. 1972; <https://www.youtube.com/watch?v=OHug0AIhVoQ>.
9. Gartner. Gartner predicts, 2016; <https://gtnr.it/2YLtXaF>.
10. Kim, G., Behr, K. and Spafford, G. *The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win*. IT Revolution Press, 2013; <https://www.goodreads.com/book/show/17255186-the-phoenix-project>.
11. Snover, J. Digital transformation: thriving through the transition. *DevOps Enterprise Summit*, 2018; <https://www.youtube.com/watch?v=qHxkcndCQoI>.
12. Wikipedia. High availability; https://en.wikipedia.org/wiki/High_availability.
13. Wikipedia. Non-functional requirement; https://en.wikipedia.org/wiki/Non-functional_requirement.
14. Zwieback, D. *Beyond Blame: Learning from Failure and Success*. O'Reilly Media, 2015; <https://www.goodreads.com/book/show/23237459-beyond-blame>.

Thomas A. Limoncelli is the SRE manager at Stack Overflow Inc. in New York City. His books include *The Practice of System and Network Administration*, *The Practice of Cloud System Administration*, and *Time Management for System Administrators*. He blogs at EverythingSysadmin.com and tweets at @YesThatTom.

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



**A discussion with David Evans,
Richard McDonald, and Terry Coatta.**

Access Controls and Healthcare Records: Who Owns the Data?

YOU NEED NOT be an expert with years of healthcare data-management experience to conclude the field is a hot mess. One visit to a hospital, clinic, or pharmacy can convince you of that. Burdened by legacy and fragmented into silos so alien from one another they can scarcely communicate, healthcare recordkeeping

has for decades frustrated any and all efforts to unify it.

The underlying reason couldn't be more obvious: Each clinic, hospital, practice, and pharmacy operates its own isolated record-management system. The platforms and techniques vary from organization to organization, with almost no provisions having been made to share any of the information.

But what if these records were handled in more of a patient-centric manner, using systems and networks that

allow data to be readily shared by all the physicians, clinics, hospitals, and pharmacies a person might choose to share them with or have occasion to visit? And, more radically, what if it was the patients—rather than the providers—who were considered to actually own the data?

It was with thoughts like these that a Toronto-based startup called HealthChain set out to create a platform for managing a patient's medication profile on the basis of relationships estab-



RICHARD MCDONALD

Questions come up as to who actually owns those records, who looks after them, and who needs to have access to them.



lished between patients and their various providers.

That vision became the challenge that HealthChain CTO **David Evans** took on, drawing on 25 years of work in the financial industry on portfolio management and quantitative research systems. In the years that led up to his transition to healthcare, he found himself increasingly intrigued with the possibilities of applying emerging digital identities and blockchain technologies to the creation of more efficient government services. Now he has an opportunity to put some of those ideas to the test.

To provide some insight into how HealthChain is addressing the medication profile-management challenge, Evans is joined in discussion here by **Richard McDonald**, a recently retired IBM Distinguished Engineer, and **Terry Coatta**, the CTO of Marine Learning Systems, a Vancouver-based startup working to develop a learning platform.

RICHARD MCDONALD: As people who have had occasion to visit doctors' offices—or even hospitals—from time to time, we all know just how important recordkeeping is to those operations. Historically, that has taken the form of paper records kept in overstuffed filing cabinets. But increasingly, it now seems also to include electronic records as the medical profession is making a belated push to fully enter the 21st century. As that process continues to move forward, questions come up as to who actually owns those records, who looks after them, and who needs to have access to them. What do you see as some of the key issues with respect to the custodial responsibilities these various medical organizations now need to address?

DAVID EVANS: As you say, at this point there are doctors who use computer systems and others who don't. The ones who keep computer-based records generally use some localized instrumentation called an EMR (Electronic Medical Record) system (also known as an EHR (Electronic Health Record) system, depending on country and usage). In terms of who owns the records kept on these things, any doctor who enters details about a patient into an EMR system is accountable for the accuracy of

that information. With that said, when it comes to ownership of the full medical record for that patient, there's growing sentiment that this information properly belongs to the patient.

The truth on the ground right now is that any patient's records are actually scattered among any number of siloed databases since, as you go from one clinic to another or from one pharmacy to another, each is going to create a fresh record for you. When it comes to determining who owns those records, I'm inclined to distinguish the information that each practitioner is responsible for maintaining on his or her own system from the sum total of all the medical information collected about some particular patient over the course of that patient's lifetime. While many believe this is information that belongs to the patient, the current reality is that the patient doesn't actually even enjoy access to that data and has little to no control over how it's used.

TERRY COATTA: Just to be clear about those silos, are you talking about centralized databases or just a lot of small individual databases maintained in different doctors' offices?

EVANS: It's essentially a mix of the two. There are lots of independent doctors' offices out there, and to the degree they are using computer systems at all, it may amount to little more than a database running on a desktop or laptop.

By the same token, there are health-care teams that span multiple locations. One of the largest in Ontario has 37 different clinic locations that all share a single EMR implementation hosted in the cloud. But I believe even that implementation is organized such that doctors are able to see information for only those patients who visit their particular clinic location.

MCDONALD: What does this look like from the perspective of an admin working at one of these clinics?

EVANS: Even where EMR applications are in place, there's a lot of aging technology to contend with. Also, early on, we were cautioned not to walk into a doctor's office just expecting to be allowed to change the workflow. There's a reason these technologies haven't necessarily advanced all that much since the 1980s. The doctors are used to them. They reflexively know they need to hit the tab three times on this

one particular page, owing to some archaic design choice. And yes, it may be stupid, but it's a design they've long since grown accustomed to.

COATTA: Is there an integration story among these different sites? Obviously, if you've got all these data silos floating around in the world, you would think there would be some standards related to information interchange. Or is it basically just the Wild West out there?

EVANS: The most promising global standard for this is FHIR, which stands for fast healthcare interoperability resources. It's a standard that has gone through a number of iterations over the years, and it's focused on interoperability.

It isn't without its flaws, but FHIR is definitely the best resource for interoperability between different systems right now. Still, there's room for lots of data duplication, and it's essentially intended only for one-time transfers of data. That is, there isn't a FHIR network in use now where all these different databases are kept in sync in any way. Or at least there sure isn't anything along those lines that I've ever seen.

MCDONALD: Am I right to assume that patients have very little say over what happens with their information?

EVANS: That's absolutely correct. If you look at health-privacy standards, they all emphasize patient consent. But for all practical purposes, the only kind of consent that actually seems to exist is implied consent, since, as things currently stand, there's no practical mechanism patients can use either to provide or withdraw consent as to how their data is to be used. Which is to say the patient is almost completely out of the picture.

The other side of this is that information also isn't shared among providers in any sort of way patients might reasonably expect. If you always just go to the same clinic, they already know you and have your records readily at hand. But if, for some reason, you find you need to go to some other clinic or end up in an emergency room, chances are you're a blank slate for anyone who treats you there. They are not going to know what allergies you have. They are not going to know what prior conditions you have. They are not going to know what medications you're on. So, that means they are going to have to

scramble around to scrounge up all the information they can from either you or a family member.

COATTA: That sounds like a complete mess, so what part of that problem are you now trying to address?

EVANS: To borrow a term from the blockchain world, we're working to deliver a shared ledger to the medical community. Our goal is to provide a view of a patient's records that not only doctors and pharmacists are able to share, but that can also be available to the patient. This is something that's actually possible today.

One of our key objectives is to keep track of all this information from the patient's perspective: What pharmacies do they use? What clinics do they visit? Which doctors treat them at those clinics? And how is it that each of these participants in the patient's Circle of Care—as we call it—is authorized to access the patient's records? Moreover, can we provide transparency and some control for patients in terms of how their data is being used and accessed?

COATTA: That sounds like an admirable goal, but can you actually make this happen? It sounds like you might be trying to move the immovable object here.

EVANS: Actually, I wouldn't say that we're working to move or replace anything. In fact, by regulation, we're precluded from replacing the existing EMR systems. We're definitely not aiming to capture all the data an EMR needs to retain since each custodian organization—that is, each healthcare provider—remains legally responsible for maintaining its own records.

However, as they continue to update the medication profile in some amount of detail for each patient, what we're hoping to do is to integrate with all those EMRs so they can keep shared state about these patients' records up to date while also leveraging that information in such a way that everyone within a patient's Circle of Care can readily review it.

We're definitely not looking to change the world here, but only to make it possible for doctors, pharmacies, and patients to share a common view of a patient's prescription history across all the different providers the patient has used over time. That's a big

enough challenge in its own right and it's an important goal, but it's also a lot more practical than attempting to replace all the EMR systems out there.

MCDONALD: It would seem that one of the keys to this problem has to do with keeping track of who the patients are and having some way to identify them across all these different parties—while also managing access control, obviously. Can you elaborate a little on this, especially the issues around digital identity?

EVANS: The system requires unique identifiers, of course. So far, that means we're able to recognize patients by way of the healthcare card numbers issued to them. By the same token, we recognize doctors by their license numbers and pharmacies by their accreditation numbers. But we also need to be able to accept some of the other identifiers accepted out in the world today, so we're working to come up with a more robust registration process. What this really comes down to, though, is taking whatever steps are necessary to guard against having multiple profiles on the system for the same patient, doctor, or pharmacist.

Built on a foundation of proven, familiar open-source software—Hyperledger Fabric and Hyperledger Composer—HealthChain presents no obvious impediments to universal deployment.

Access to a network of patients and providers formed using HealthChain, however, is limited only to credentialed participants, who in turn are granted access only to certain information assets on which they're allowed to perform a specific set of functions.

This means that, ultimately, access-control lists may prove to be the key to resolving the longstanding healthcare data-management stalemate, since they're not only the means by which access to objects can be bound to each participant type, but also the means for defining the operations permitted on any given object.

MCDONALD: Now that you have told us what you set out to accomplish with your application, tell us what you actually did.

EVANS: Well, first off, we are using Hyperledger Fabric and Hyperledger Composer, both of which come out of a blockchain-related project backed by the Linux Foundation. Fabric basically provides a bunch of tools around a blockchain component, which effectively gives us an indelible record of all the idempotent transaction requests that come in. We think of these transactions as “smart contracts,” but in any event they’re transactions that update state. Which is to say Hyperledger Fabric gives us a way to maintain a global state database.

The underlying database technology is CouchDB, which gives us an object-store for handling JSON (JavaScript Object Notation) objects, with MapReduce being used to apply indexes and perform fast queries against all these JSON object structures. Another important technology for us that came along as part of Fabric is Kafka, which provides for high-performance message streaming. That provides for event notifications between peers as well as for the delivery of notifications to other processes.

Hyperledger Composer, meanwhile, is a framework built on top of Fabric to help you define what the Hyperledger world thinks of as a business network, where all the participants within a network, as well as all the various assets the network is expected to manage, are identified, along with the control rules that govern access between all the different participants and their respective assets. In our case, we leverage this business-model language to define rules for a business network expressly designed for the management of prescription records.

MCDONALD: What are some of the novel things your architecture does to accomplish this?

EVANS: Hyperledger Fabric is what’s known as a permissioned—or private—blockchain, which requires everyone to have an identity recognized by that blockchain. Once you’re issued an identity, you’re also issued PKI (public key infrastructure) certificates, which you can then use to identify yourself. I should add that blockchains are generally really good at handing out a bunch of keys. Unfortunately, they’re not all that good at managing those keys. So, if you are building a system that utilizes

blockchain—or PKI in general—you’re going to need to come up with your own way to manage those keys.

We have built a microservice architecture, which resides on top of this whole Hyperledger stack, that is focused on matching authenticated users with their respective keys or certificates. Once that’s accomplished, the idempotent transactions can then be executed and digitally signed using the key belonging to that individual.

Another dimension is that one person can have multiple credentials on the same network. This, too, has to be managed. Obviously, everyone can become a patient at some point, so all of us qualify for patient credentials. Only some people qualify as doctors, so they need to present appropriate credentials that the system recognizes. There also will be clinical administrative staff that needs to use the system. Even though they do not have the credentials to write prescriptions, they might be able to enter draft prescriptions for a doctor to approve later. What this means is that one user could potentially be associated with multiple keys—each of which defines a different set of things they are allowed to do within the business network.

This is where access controls come in. Each participant type associated with a person on the network determines who is allowed to perform particular functions. If you happen to be someone who has multiple keys on the platform, each key represents a different set of things you are credentialed to do. For example, if you are a patient and you query the system through HealthChain’s Portage APIs to “show me all patients,” the only record you will get back will be your own. A doctor, on the other hand, might say, “Show me all patients named Bob.” But that doesn’t mean the doctor should be shown the records for all the patients named Bob across the entire system. It just means the doctor should be shown the records of those patients named Bob who have listed that particular doctor as part of their Circle of Care.

COATTA: But if I have multiple keys, would you manage all those for me within the context of a single identity?

EVANS: This is a bit of a slippery

concept. In addition to defining participant types, we define the different application types that can integrate with the HealthChain network. This, for example, allows us to ensure that if you are connecting from an EMR application at a clinic, you will be allowed to connect only as a doctor or a clinic administrator. It just would not make sense for you to connect using your patient credentials.

As another example, it’s common for doctors to connect from multiple clinic applications, so the certificate associated with an application instance ensures that we know which clinic the doctor is connecting from, since the authorization token issued to a doctor at the time of authentication is unique to that application instance. Regardless of which clinic application Dr. Jones might happen to be connecting from, it will be Dr. Jones’s credentials that are used to sign transactions on the network.

MCDONALD: But if Dr. Jones has a relationship with some particular pharmacy or clinic, and yet also happens to work at some particular hospital, how are you going to be able to sort that out? It sounds like there might be an organization tag associated with each particular transaction.

EVANS: Exactly. We keep records on which clinic or hospital locations each prescriber is authorized to act on behalf of. And each application certificate is bound to a specific clinic location. In this way, we are able to guard against doctors spoofing the locations they are communicating from.

Although policy arguments have long been made for letting patients themselves determine who should be given access to their healthcare records, it turns out the most compelling reason may prove to be a very simple technical one. The traditional premise is that this information belongs to the providers, so determining who can receive updates requires either wading through data for a large number of patients or relying upon some number of data joins. By instead looking at this conundrum from the patient’s perspective, it quickly resolves into simply determining which providers the patient has had dealings with previously and just leaving it at that—a far more elegant approach.

It just so happens the patient-centric approach could also lead to all sorts of new possibilities as more metadata accumulates around the defined patient-provider relationships. At minimum, this could make it possible to tune access controls at a much more granular level than is currently possible.

MCDONALD: What was it about the Hyperledger technology that initially drew you to the platform?

EVANS: First off, the CouchDB database technology underlying the platform is something that has proved to be quite robust and very capable of scaling. Data is managed in JSON format, which FHIR and other data standards generally support.

It also was reassuring to find that Kafka was being used to handle the messaging. Over the years I worked in the finance space, Kafka proved to be a solid contender in high-speed trading systems. Which is all to say I became pretty enamored with Hyperledger Fabric's underlying technology stack.

And, of course, the Hyperledger Fabric stack contains a blockchain data structure that captures an indelible record of transaction requests. People associate a lot of different things with blockchain, but, at heart, it's just a data structure for transactions that is replicated, with the benefits being security and redundancy. So, here you have got Fabric, a technology focused on making sure a state database stays in sync with all the transactions it has processed, while also ensuring each peer gets a replicated copy of state. If, for some reason, someone should manage to get access to one of those peers directly and try to modify the data, that node will be dropped from the network. It's this tamper-evident property of blockchain that gives the underlying data yet another dimension of security. These security characteristics—combined with the robustness of the underlying stack—sold us on the Hyperledger Fabric technology.

MCDONALD: A little earlier you cited an example that suggested the ability to adjust access permissions for a user according to the particular requirements of a situation. That tells me there is either a fair amount of work yet to be done programmatically to reflect policies capable of covering those situations or that you have some provisions

that allow clinic administrators to deal with these sorts of outlier situations on a case-by-case basis. In general, how have you dealt with implementation issues like these as they have come up?

EVANS: One of the challenges we faced right off had to do with figuring out the right model for managing the relationships between patients and their respective prescribers and medical providers. We initially took a traditional relational data-modeling way of thinking about things from the provider's perspective. But what we discovered was that, by taking the provider's view, every pharmacist, clinic location, and doctor would end up having a reference to any patient they had ever treated or served. As a result, we would have had to deal with huge arrays of patients associated with each provider.

From there, we moved along to a different relational concept where we started to think in terms of joining tables or joining objects that would keep references to each of the parties—effectively creating central relationship objects. The challenge we ran into there, though, had to do with the constraints of the Hyperledger Composer access controls. Basically, any given access control defines a participant, an asset, the type of access that participant ought to have to the asset, and a constraint. The problem with this was that by introducing a third mapping object at the time the access controls for these assets are evaluated, we would have been unable to access all the data required by the mapping object to properly evaluate whether or not that access should be granted.

This led us to take another stab at the problem by looking just at who really owns these relationships. And that is when things started to fall into place more naturally. We now keep those relationships as part of the patient profile. One big advantage of taking a patient-centric approach is that the patient profile provides all the information needed to enforce access controls. Then, if somebody without a relationship to the patient wants to take a look at their records, we can just ask, "Does the patient have a relationship with this person who is trying to get access to the records?"

MCDONALD: This is quite a departure from the norm of developing something that looks at things primarily



DAVID EVANS

Our goal is to provide a view of a patient's records that not only doctors and pharmacists are able to share, but that can also be available to the patient. This is something that's actually possible today.





TERRY COATTA

If you've got all these data silos floating around in the world, you would think there would be some standards related to information interchange. Or is it basically just the Wild West out there?



from the perspective of the clinic or the pharmacy.

EVANS: Exactly, and that's what makes it really exciting. It's pretty clear this could lead to all sorts of new possibilities since we are only just starting to attach more metadata to these relationships. For example, think about a scenario where a patient shows up in the emergency room of a hospital the patient has never visited before. This means the doctors there cannot call up any records for this patient. For exactly this sort of time-sensitive situation, it would be good to include some sort of provision that allows for break-the-glass, time-limited access to the patient's records—maybe only for the following 24 hours or so.

Another possibility we are exploring would amount to more of a lockbox-based security mechanism. Basically, for each patient, we already have a record of all the organizations or individuals that the patient has some sort of connection to, so we could just combine that knowledge with the cryptographic keys associated with each of those relationships. Then, using those keys, we could go one step further so that the identifier presented to a particular provider looking up a patient's healthcare number would actually be unique to that particular patient relationship. This, in effect, would amount to a mechanism for anonymizing data access right at the source.

The point here is that, as we look at attaching more metadata to these relationships, we can begin to really fine-tune these access controls at a very granular level.

MCDONALD: What are some of the other challenges you have addressed along the way?

EVANS: We were frustrated for a while because we were having a hard time getting performance out of our CouchDB implementation. We ended up introducing an Elasticsearch capability that leveraged events from transactions to trigger updates that would allow for a faster search capability. That was a bit of a hack, though. So, I'm happy to say we have since figured out how to optimize and apply indexes to CouchDB more effectively, meaning we have managed to reduce the need to use that Elasticsearch enhancement. Still, along the way, we

came to a much better understanding about CouchDB and the possibilities that indexing offers there.

COATTA: Now that you know much more about Hyperledger Fabric and what might be done to optimize CouchDB performance, what advice would you offer other developers looking to follow a similar path?

EVANS: DevOps, DevOps, DevOps. And that's because extreme automation is really what is called for here. We have been very fortunate to have a team that is quite strong on DevOps. We were able to migrate our testnet environment from an IBM infrastructure to AWS (Amazon Web Services), while managing to preserve all our data and keys—along with the access they imply. Now that we are on the AWS infrastructure, we are going to be able to scale up even more. Similarly, as other people start to move in this direction, they are going to find almost all the infrastructure involves Docker containers. We also ensure that any given microservice will accept only TLS (Transport Layer Security)-secured communications from other trusted services.

And because we are now operating in a world like this, it's not enough just to be a developer. You also need to be focused on automating your development environment and understanding your release process since things are just evolving too quickly not to be on top of that. There are a lot of headaches to go through before you learn how to manage keys effectively. And you need to be thinking in terms of seeding your environment for testing purposes right from the start so you will later have multiple configuration options you can readily use to perform various scalability and performance tests.

As for all those developers who expect to find themselves in some sort of blockchain environment at some point, I would say it's important to remember that these environments need to be kept alive forever. You start with a genesis block and then go on from there, with keys being distributed to each and every user and organization that has ever been a part of the chain. There has to be some reasonable means for managing that... forever essentially. So, as I say: DevOps, DevOps, DevOps is key. 

Introducing ACM Digital Threats: Research and Practice

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

Now Accepting Submissions

ACM Digital Threats: Research and Practice (DTRAP) is a peer-reviewed journal that targets the prevention, identification, mitigation, and elimination of digital threats. DTRAP promotes the foundational development of scientific rigor in digital security by bridging the gap between academic research and industry practice.

The journal welcomes the submission of scientifically rigorous manuscripts that address extant digital threats, rather than laboratory models of potential threats. To be accepted for publication, manuscripts must demonstrate scientific rigor and present results that are reproducible.

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



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



Association for
Computing Machinery

Advancing Computing as a Science & Profession

DOI:10.1145/3274770

Used in the design of the Internet and Unix, the layered services of the hourglass model have enabled viral adoption and deployment scalability.

BY MICAH BECK

On The Hourglass Model

THE HOURGLASS MODEL of layered systems architecture is a visual and conceptual representation of an approach to design that seeks to support a great diversity of applications and allow implementation using a great diversity of supporting services. At the center of the hourglass model is a distinguished layer in a stack of abstractions that is chosen as the sole means of accessing the lower-level resources of the system. This distinguished layer can be implemented using services that are considered as lying below it in the stack as well as other services and applications that are considered as lying above it. However, the components that lie above the distinguished layer cannot directly access the services that lie below it.

David Clark called the distinguished layer the “spanning layer” because in the Internet architecture it bridges the multiple local area network implementations that lie below it in the stack (see Figure 1). Clark defined the function of the spanning

layer by its ability to “... hide the detailed differences among these various technologies, and present a uniform service interface to the applications above” and identified the Internet Protocol as the spanning layer of the Internet (see Figure 2).⁵ Arguably the spanning layer also includes other elements of the Internet Protocol Suite that access lower-layer services (such as ARP and DHCP).

The shape suggested by the hourglass model expresses the goal that the spanning layer should support various applications and be implementable using many possible supporting layers. Referring to the hourglass as a design tool also expresses the intuition that restricting the functionality of the spanning layer is instrumental in achieving these goals. The elements of the model are combined visually in the form of an hourglass shape, with the “thin waist” of the hourglass representing the restricted spanning layer, and its large upper and lower bells representing the multiplicity of applications and supporting layers, respectively.

The hourglass model is widely used in describing the design of the Internet, and can be found in many modern networking textbooks.⁸ A similar principle has also been implicitly applied to the design of other successful spanning layers, notably the Unix operating system kernel interface by which we mean

» key insights

- **Adoption of a common service interface is a key to interoperability, portability, and “future-proofing” in the face of rapid technological change.**
- **The design of both the Internet and Unix followed the hourglass principle, leading to dominance in two software markets while also enabling disruptive innovation.**
- **Many successful interface designers adhere to a discipline of simplicity, generality, and limitation of the common interface.**
- **This article introduces the Deployment Scalability Tradeoff, a principle that seeks to explain the reason for the success of this discipline.**

ILLUSTRATION BY PETER GROWTHER ASSOCIATES



Figure 1. The hourglass model.

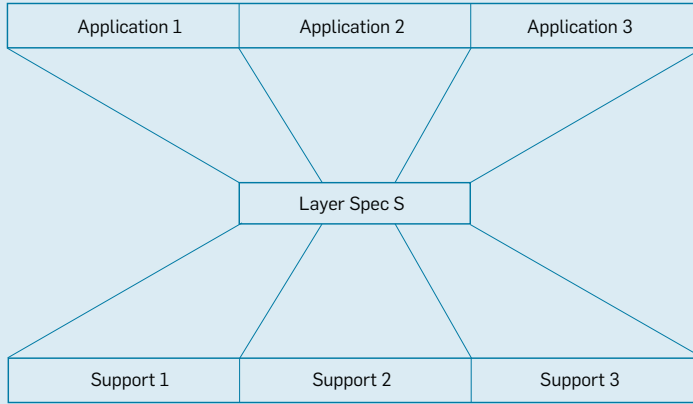
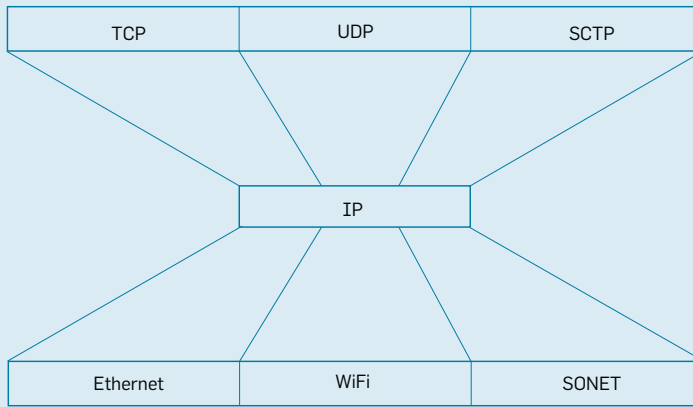


Figure 2. The Internet hourglass.



the primitive system calls and the interactions between user processes and the kernel prescribed by standard manual pages.⁹ The impressive success of the Internet has led to a wider interest in applying the hourglass model in other layered systems, with the goal of achieving similar results.^{1,7,11} However, application of the hourglass model has often led to controversy, perhaps in part because the language in which it has been expressed is informal.

The purpose of this article is to present a formal model of layering and to use this model to prove some relevant properties.^a I will then use this formal model to explain the application of the hourglass model in the design of the Internet and Unix, and to show how it relates to some less formal concepts in the design of layered systems.

Overview

Let's begin by presenting an abstract framework for reasoning about layered architectures and spanning layers in particular. We assume the existence of a certain relationship between layers, namely that one layer specification can support another. This article does not give definitions for layer specifications or the *supports* relation, as the complete formalization is somewhat complex and not, I believe, necessary to understand the argument. Given the existence of the supports relation, I share definitions of possible supports and possible applications of a layer in terms of it. The Hourglass Theorem consists of two simple properties that can be derived from these definitions (see the section on the Hourglass Theorem). These definitions and the properties that we derive from them create a framework for characterizing a spanning layer in terms of the multiplicity

of its possible applications and supports. The Hourglass Theorem, which expresses a trade-off between the multiplicity of possible applications and supports and the logical strength of the spanning layer, is proved. Moreover, the question of whether the Hourglass Theorem provides a formal justification for end-to-end arguments is explored. I then state and argue for the validity of a more general principle—the Deployment Scalability Trade-off—as there is an inherent correlation between deployment scalability of a system with a given spanning layer and the weakness, simplicity, generality and resource limitation of that layer's specification.

The DST is intended as a design principle relating the hourglass design in layered models to the scalability of systems that they describe. The DST combines logical weakness with design criteria that will not be formalized in order to put that principle into a context that also includes more familiar related concepts. The intention is that future work may lead to formalization of some of these other characteristics, the development of metrics, and even a characterization of the trade-offs precise enough to accurately model the implications of specific service design choices.

The Hourglass

Definition 1. A service specification is a formal description of the syntax and necessary properties of an operating system or application-programming interface (API).

A service specification S describes an interface: it specifies the behavior of certain program elements (functions or subprograms) through statements expressed in program logic. For instance, these might be such statements:

1. $\forall A, B \in \mathbb{Z} [(A + 1) + B = (A + B) + 1]$
2. $x, y : \mathbb{N} [\{x > 0\} y := x * x \{y > x\}]$

In formal terms, a service specification is a theory of the program logic. The set of all such specifications expressed in the language of the specific logic is denoted by Σ . In practical terms, a service specification describes the operations of a protocol suite or a programming interface, such as operating system calls.

Definition 2. A specification S_1 proves another specification S_2 (written

^a An undertaking that was suggested to me many years ago by Alan Demers.

$S_1 \vdash S_2$) iff S_2 can be derived from S_1 through application of the rules of the logic in which they are both expressed.

Definition 3. A specification S_1 is *weaker* than another specification S_2 iff $S_2 \vdash S_1$. S_1 is *strictly weaker* than S_2 iff $S_2 \vdash S_1$ but $S_1 \not\vdash S_2$.

Definition 4. A *supports* relation $S \prec_p T$ exists between two service specifications S and T and a program p iff in any model where S is correctly instantiated, the program p correctly implements T using the instantiation of S .

The supports relation is intended to be analogous to the “reduces to” relation of structural complexity theory.

The Hourglass Lemma. The intuition behind this lemma is that any API that can be supported by a given underlying layer can also be supported by any underlying layer that is stronger. Similarly, a layer that can support a given API can also support one that is weaker.

While detailed definitions of service specifications and the supports relation have been omitted here, I call upon the intuition of the reader to justify the following lemma presented here without proof. This lemma is the only place where the omitted basic definitions are used, and the remainder of this discussion is based upon the lemma.

LEMMA 1. If S_1 is *weaker* than S_2 , then 1) $S_1 \prec_p T \Rightarrow S_2 \prec_p T$, and 2) $T \prec_p S_2 \Rightarrow T \prec_p S_1$.

Proof omitted.

The two properties that comprise the Hourglass Lemma follow directly from fundamental definitions in program logic, and they also correspond very closely to covariance of return types and contravariance of argument types in object-oriented inheritance.^{3,b}

Pre- and post-images. Formal analogs to scalability are expressed in terms of how large the sets of service interfaces are that can possibly support or can be supported by a particular specification. To this end, the pre- and post-images of a specification are defined under *supports* (see Figure 3).

These definitions are given relative to the set Π of programs considered as possible implementations of one layer in terms of another. We do not specify Π because we know of no accepted characterization of all “acceptable im-

plementations” of one layer in terms of another. This is certainly a limited class, and is in fact finite since programs that are too large are considered unwieldy from a software engineering point of view. This class also changes over time, as hardware and software technology changes the set of available implementation tools.

Definition 5. $pre_{\Pi}(S) = \{T \mid \exists p \in \Pi [T \prec_p S]\}$

Definition 6. $post_{\Pi}(S) = \{T \mid \exists p \in \Pi [S \prec_p T]\}$

In representing the set Π in our model as an external parameter we are not accounting for software engineering aspects of these definitions.

Using the hourglass as an analytical tool. Reference to the hourglass model is sometimes conflated with the idea of the spanning layer as a standard enforced by some external means such as legal regulation or as a condition of membership in some community.

However, we can use the analysis of pre- and post-images of the supports relation as tools to analyze a layered system without necessarily relating it to any standards process.

By selecting any set of services at one level of a layered system, we can ask what the design consequences would be if it were adopted as the spanning layer of a hypothetical system. Adoption as a spanning layer means that no other services would be available at that layer. Any participant in the system would have to use it as the sole means of accessing the services and resources of lower layers. Viewed in this way, the pre-image of the supports relation denotes all possible implementations of the prospective spanning layer and the post image denotes all of its possible applications.

I use the term “denotes” because the pre- and post-image are not necessarily useful in actually enumerating these

Figure 3. Pre- and postimages.

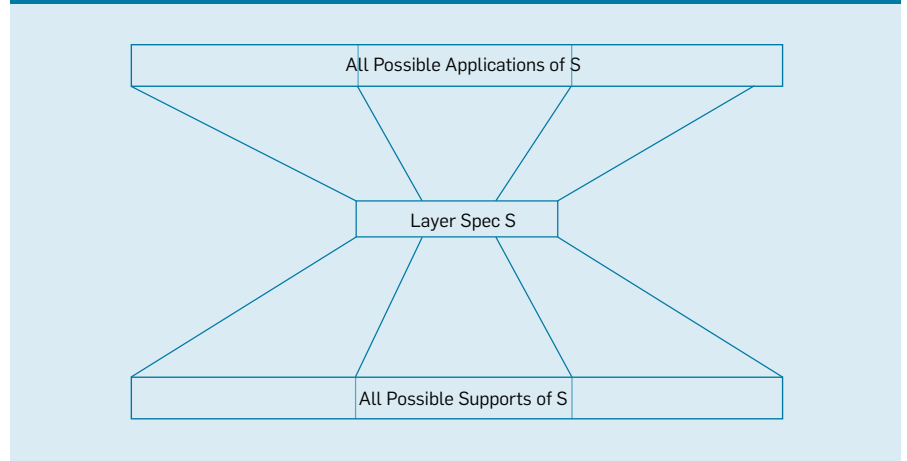
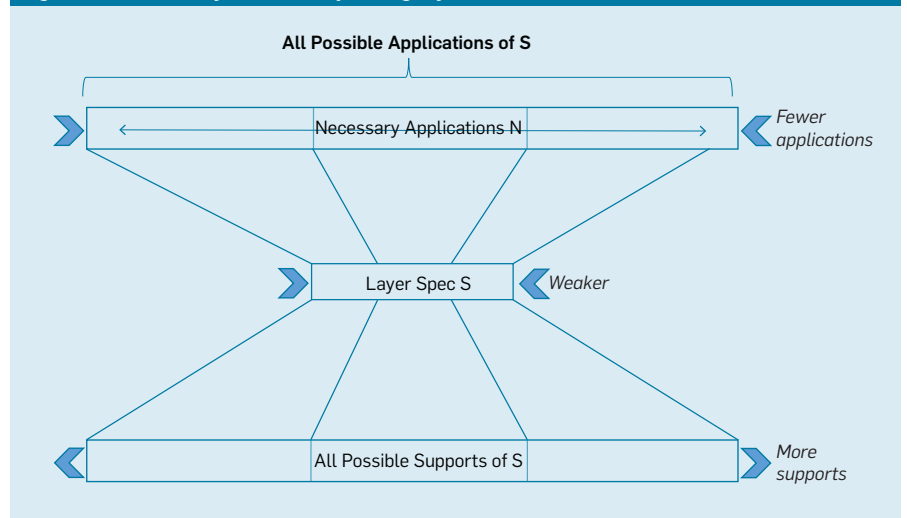


Figure 4. A minimally sufficient spanning layer.



^b This observation courtesy of Tim Griffin.

sets of specifications, since there is no formal specification for the value of Π , nor a way of determining whether a particular program p is in Π . Even when there is community agreement that certain programs are either in Π or in its complement, there may still be contention regarding some boundary cases.

Taking a descriptive view of the hourglass allows us to use it as an analytical or predictive tool to understand the impact of a community's adopting a particular interface as a standard, be it de jure or de facto. Making the distinction between the use of the hourglass as a descriptive tool or as a means of justifying a standard also explains how different hourglasses can be examined and compared within the discussion of the same layered system. Every prospective spanning layer has an associated pre- and post-image, regardless of whether it is considered for any kind of standardization.

The Hourglass Theorem

This theorem is central to our understanding of the hourglass model.

THEOREM 1. *If a specification S_1 is weaker than another specification S_2 , then 1) $\text{post}_\Pi(S_1) \subseteq \text{post}_\Pi(S_2)$, and 2) $\text{pre}_\Pi(S_1) \supseteq \text{pre}_\Pi(S_2)$. Proof:*

1. *By definition, $T \in \text{post}_\Pi(S_1)$ iff $\exists_p \in \Pi [S_1 \prec_p T]$ so by Lemma 1 $S_2 \prec_p T$, thus $T \in \text{post}_\Pi(S_2)$*

2. *The proof is symmetric to Part 1.*

□

The Hourglass Theorem conveys (approximately) that a weaker layer specification has fewer possible applications but more possible supporting layers than a stronger layer specification.

Minimal Sufficiency

In terms of the hourglass shape, the thin waist (weak spanning layer, as noted earlier) naturally tends to give rise to the large lower bell of the hourglass (many supports). However, a weaker spanning layer also tends to give rise to a smaller upper bell (fewer applications). Thus, some countervailing element must be introduced into the model to ensure it is in fact possible to implement all necessary applications (see Figure 4).

As a design goal, we model the necessity of implementing certain appli-



The thin waist of the hourglass is a narrow straw through which applications can draw upon the resources that are available in the less restricted lower layers of the stack.



cations by introducing the set of necessary applications as another external parameter $N \subseteq \Sigma$.

Definition 7. A specification S is sufficient to support a set of specifications N iff $N \subseteq \text{post}_\Pi(S)$.

A spanning layer must be strong enough to support all necessary applications, but the stronger it is the fewer possible supports it has. The notion of minimal sufficiency serves as a means to balance these two design requirements:

Definition 8. A specification is *minimally sufficient* for N iff it is sufficient for N but there is no strictly weaker specification sufficient for N .

The balance between more applications and more supports is achieved by first choosing the set of necessary applications N and then seeking a spanning layer sufficient for N that is as weak as possible. This scenario makes the choice of necessary applications N the most directly consequential element in the process of defining a spanning layer that meets the goals of the hourglass model.

Note the implication that the trade-off between the weakness of the spanning layer and its sufficiency for a particular set of applications N is unavoidable. This suggests the design of a spanning layer may have a tendency to fail if it attempts to both achieve a high degree of weakness and also be sufficient to support a large set of necessary applications.

End-to-End Arguments

End-to-end arguments have influenced the design of many layered systems, most famously the Internet. Historically, end-to-end arguments have often been invoked in discussions of whether it is appropriate to add functionality to a layer of the network, and, in particular, when discussing the Internet's spanning layer.

Claims have often been made that an end-to-end argument implies that adding functionality to the spanning layer will result in a diminution of the scalability of the Internet, although this term does not have a generally agreed-upon definition. Here, the hourglass model is used as a reference, to hypothesize that scalability is enabled by a spanning layer that has implementations using as many different supports as possible, given the necessary applica-

tions that it in turn supports. The analysis is that end-to-end arguments do not necessarily lead to a spanning layer that maximizes possible supports.^c

The introduction of the classic paper “End-to-End Arguments in System Design”¹⁰ gives a general statement of the argument that applies to all kinds of layered systems:

“The argument appeals to application requirements, and provides a rationale for moving function upward in a layered system, closer to the application that uses the function.”

However, the discussion then focuses on the more specific context of layered communication systems, and the argument is described again:

“The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)”

Much of the paper is devoted to the context of layered communication systems. Examples of issues implementing functions in lower layers fall into two major categories: When the lower layer lacks knowledge of application requirements or status, and when local communication functions are combined to create a global service, so the characteristics of the global service can only be detected by its clients.

Moving function upward in a layered system can have the effect of removing responsibility for particular functionality required by applications from lower layers. This leaves higher layers free to implement their true requirements without imposing costs or other artifacts due to inappropriate functionality being implemented by lower layer services. However, when applied to the spanning layer, end-to-end arguments do not necessarily lead to a design that is logically weaker, and thus has more possible supports.

Examples can be found in which moving function upward in a layered

system indeed leads to a weaker spanning layer. However, other examples can be given in which it leads to a stronger one. These examples have been omitted due to lack of space, but their existence suggests this question: Why have end-to-end arguments been so commonly invoked in discussions of scalability?

This analysis of the relationship between the hourglass model and end-to-end arguments is included because those arguments are often cited as a founding principle of the Internet, and credited as a major reason for its remarkable success. In fact, it was interest in understanding the power that is attributed to end-to-end arguments that led us to formalize scalability in layered systems and hypothesize logical weakness as an underlying cause. Finding no necessary causal connection between end-to-end arguments and logical weakness was unexpected, and the result is indeed noteworthy.

In light of this result, we offer a hypothesis for the apparent impact of end-to-end arguments on the scalability of the Internet: In the cases where application of an end-to-end argument results in a weakening of the spanning layer while still supporting all necessary applications, the result may be an increase in possible supports due to that weakening. If the result is an increase in scalability that increase may be attributed to the end-to-end argument, even if the effect is more specifically due to increased weakness. If this hypothesis is true, it would explain why end-to-end arguments are a relevant but inexact tool in the design of layered systems for maximum scalability.

Saltzer et al.¹⁰ present a very general design approach for the placement of specific functionality in layered systems: keep the lower layers general in order to allow the specific requirements of higher layers to be most effectively addressed. This approach is rooted in the methodology of formal reasoning in logic, mathematics and the sciences, but its application was informed by experience in the design and implementation of complex systems, with Multics being cited most often. As practitioners of a field grounded in principle as well as practice, computer scientists are drawn to ask why this approach has sometimes

seemed so powerful in conferring scalability and how it could be used to predict effects on system scalability of different design alternatives.

We understand Salter et al.’s classic paper as finding a justification for the more general argument in the particulars of layered communication systems. To the extent the Hourglass Theorem is a causal element in system scalability, it is not necessarily applicable to explain the effectiveness of end-to-end arguments. We now turn to other aspects of the hourglass model that have traditionally been associated with scalability and attempt to relate them to the formal model of layered systems and to end-to-end arguments, although we do not have formal results analogous to the Hourglass Theorem to justify claims of causal linkage. To fit these other aspects of the hourglass model into our framework is a step toward developing a more complete formalization.

Spanning Layer Characteristics

Consider a design space of spanning layers that can support a particular set of necessary applications. Each point in this space is characterized in a number of ways, according to its logical or engineering attributes. One important job of a system architect is to find a point in this design space in which certain goal attributes fall into target ranges by adjusting the values of those attributes that are under their control. From this perspective, the subspace of feasible designs has some shape that the system architect must understand and navigate in order to reach their design goals.

The hourglass model can be understood as describing the general shape of the subspace that we navigate in designing layered systems. If one goal is maximizing possible supports, then the Hourglass Theorem tells us that the slope of the subspace of feasible solutions when considering this goal as a function of the logical weakness of the spanning layer is non-negative. We have no metrics for logical strength or for the size of the space of possible solutions, only for the notions of one service description being weaker than another and one set of service descriptions being included in another.

These definitions allow system architects to reason about the sign of

^c Jerry Saltzer illuminated the point that the hourglass model is distinct from end-to-end.

the slope, but not its steepness nor what value is necessary in order to achieve a particular design goal. Because only the relationship between one independent attribute and one dependent goal attribute has been formalized, there are no results about interactions between the various dimensions. The ability to obtain such abstract results is one strength of mathematical logic; the fact that such results are not more specific and perhaps more satisfying to readers unfamiliar with logic may seem to some as a weakness. The purpose here is to create a structure for the definition of metrics and the proof of further properties by researchers in the field.

With this goal in mind we now consider a number of other spanning layer attributes (simplicity, generality, and resource limitation) that have been viewed as important within the design community, and present a hypothesis for how they act together to impact the overall goal of system scalability. These choices and the explanations offered reflect the author's study of and experience as a researcher in the fields of operating systems and wide area network services.

Simplicity. The attribute of simplicity is one aspect of the thin waist of the hourglass. (Note, simplicity is not correlated with logical weakness, as the strongest possible predicate is the primitive assertion "False," which is also the simplest.) Simplicity is an important aspect of the acceptability of the spanning layer as a tool used in the implementation of higher layer services.

A key aspect of simplicity is orthogonality. In a service interface, orthogonality means there is only one way of gaining access to any fundamental underlying service or resource. Redundant features increase the complexity of an interface without making it logically stronger. System architects understand the value of orthogonality in the design of interfaces and are more likely to accept as a community standard a design that has this form of simplicity.

An example of orthogonality in the Unix system call interface is the decomposition of file movement between directories into the creation of a physical link (using `link()`), creating a copy of a pointer in a destination location, followed by deletion of the original (using

`unlink()`). The composite file movement operation is implemented in a user level command (`mv`). This allows the user level file movement operation to be easily generalized to include movement between physical volumes (which requires copying of contents) and for efficient file sharing within a volume to be implemented using `link()`.

Generality. It is often observed the diversity of applications supported by the Internet far outstrips those foreseen by its original designers. Rather than crafting a spanning layer to support the functionality of only the initial target applications, the designers created a set of general primitives such that those target applications lay within the space of applications supported by them. That space also contains many other applications, including many they may not have originally foreseen.

In terms of our analysis of the hourglass model, the design of the spanning layer S yielded a very rich set of possible applications $post_H(S)$. The design challenge was to do so without increasing the logical strength of the spanning layer. Our analysis of end-to-end arguments leads us to associate their application with the design of a general spanning layer. This may help to explain why, even in cases where applying an end-to-end argument does not result in a weakened spanning layer, there may still be an increase in the class of supported application due to greater generality.

Resource limitation. The spanning layer provides an abstraction of the resources used in its implementation, preventing them from being accessed directly by applications.⁵ As such, it also defines the mechanisms that allow those resources to be shared by applications and among users. In some communities, the modes of sharing are open, with few restrictions in place to ensure fairness among users (such as resource quotas). Such openness is one way of enabling the spanning layer to be logically weak (such as by not implementing authorization, metering and billing of resource utilization.) One way of managing more open modes of resource sharing is to limit the resources used by any individual service request, requiring large allocations of resources to be fragmented, as in statistical multiplexing.⁶

Such fragmentation allows for more fluidity in the allocation of resources, with competition between users occurring on a finer scale. This point is perhaps clearest when comparing extremes, such as the provisioning of a virtual circuit of unbounded duration compared to the forwarding of a single datagram with a bounded Maximum Transmission Unit. A similar extreme comparison can be made between the allocation of a disk partition for an unbounded period of time and obtaining a time-limited lease on a single storage object with a maximum size. Resource limitation, along with the definition of acceptable algorithms for aggregating individual allocations (for example, "TCP friendly" flow control in Internet applications) means that use of the specification will not result in overtaxing the resources of the platform on which it is implemented.

In other words, the thin waist of the hourglass is also a narrow straw through which applications can draw upon the resources that are available in the less restricted lower layers of the stack. Resource limitation can affect the ability of the system to function in environments where the demand for resources locally or transiently exceeds the capacity of the system. A countervailing consideration is performance, as extremely fine-grain contention for resources can impose unacceptable overheads.

Deployment Scalability

End-to-end arguments have sometimes been cited as a reason that failure to keep complex functionality out of the spanning layer (maintaining a thin waist) will limit the scalability of a layered system. We have constructed a formal framework for analyzing layered systems and sought to use it understand the effect of end-to-end arguments. We have tried to characterize aspects of thinness in both formal and informal ways, and now seek to use this model to account for the design principle that motivates maintaining the thin waist of the hourglass.

The Hourglass Theorem has shown a structural link between the logical weakness of the spanning layer and an expansion in the set of possible supports. I have argued informally that sim-

licity, generality, and resource limitation can also affect possible supports and applications.

The design principle sought to examine is that thinness of the spanning layer is correlated with greater success in its adoption and longevity. This is sometimes expressed as the system exhibiting scalability. However, scalability means little if we do not specify the attribute in which we desire the system to exhibit an ability to grow.

Deployment scalability is used to characterize a spanning layer being adapted to finding success in the form of widespread adoption. Deployment scalability is intended to imply the kind of “viral” adoption that the Internet and Unix spanning layers have exhibited. This definition is proposed as a (admittedly imprecise) characterization of success in global infrastructure service interface design.

Definition 9. Deployment scalability is defined as widespread acceptance, implementation, and use of a service specification.

The notion of deployment scalability is introduced in order to have a vocabulary for expressing the goal that is implicit in the design of a spanning layer for global infrastructure.

For example, in describing the role of the Internet’s thin waist, Peterson and Davie⁸ state “The hourglass’s narrow waist represents a minimal and carefully chosen set of global capabilities that allows both higher-level applications and lower-level communication technologies to coexist, share capabilities and evolve rapidly.” It is the meaning of “minimal” and “carefully chosen” that we are trying to characterize.

In terms of the formal model of layered systems, we suggest that having many possible supports and many possible implementations is correlated with the goal of deployment scalability. The Hourglass Theorem would then extend this to a correlation between minimal sufficiency and deployment scalability. We have given some informal arguments to support similar relationships between other aspects of a thin spanning layer and deployment scalability.

Each of these aspects can be evaluated in isolation, but in the service of our original motivation to link the thin waist to a general notion of scalability,



The Hourglass Theorem has shown a structural link between the logical weakness of the spanning layer and an expansion in the set of possible supports.



we offer the Deployment Scalability Trade-off (DST): There is an inherent correlation between deployment scalability of a system with a given spanning layer and the weakness, simplicity, generality, and resource limitation of that layer’s specification.

The original motivation for creating a formal model of layered systems was to better understand end-to-end arguments in the context of the hourglass model. We sought to explain and guide efforts to generalize shared network resources while addressing the intuitive design principle that the requirements of scalability had to be the primary and overriding constraint. The DST is a candidate as a more general design principle that situates end-to-end arguments in a complex space of design criteria. The Hourglass Theorem is a first step in an explanation of the role of logical weakness in the DST.

Examples and Applications

Giving an account of an application of the Hourglass Theorem can be tedious. The antecedents of the theorem require the definition of the specification language, a program logic, all acceptable programs Π and the set of necessary applications N . This presentation will be restricted to giving a less formal account of the implications of the DST.

Tree building in IP multicast. Global Internet routing is made possible through the use of interoperable approaches to internal and external routing within and between local networks. The metrics assigned to individual links by network administrators are somewhat arbitrary, but when used as inputs to a combination of shortest path algorithms (interior gateway protocols) and a policy driven peering protocol based on commercial agreements (the Border Gateway Protocol), the result is often acceptably similar to some intuitive notion of efficiency.

Multicast routing is much more complex. An IP multicast group is based on tree-structured forwarding, with the tree being built dynamically as clients join and leave the group. The notion of efficiency in multicast must not only account for the path taken from the source to each receiver, but also the amount of control communication required during discovery of the paths that actually reach receivers

(such as flood and prune), maintaining the tree and responding to changes in topology. Algorithms that maintain accurate trees require persistent state at intermediate nodes, which results in the spanning layer being strengthened.

Historically, a number of protocols have been proposed that perform well in different environments, with particular bifurcation between groups that are sparse in the subnets they reach (with a low degree of branching toward the leaves of the tree) and those that are dense (with a higher degree of branching toward the leaves). Because different candidate protocols perform better in different scenarios, multiple implementation approaches have been maintained by network providers and selected by applications.

The resulting “fat” multicast spanning layer has limited simplicity and generality in not offering a single universal solution. The best choice for a particular situation may be unclear, or may change over time. This has arguably contributed to the lack of continuous, universally available deployment of IP multicast throughout the Internet. Application builders have used overlay multicast and repeated unicast as workarounds at the cost of redundant traffic.

Internet address translation. Network Address Translation (NAT) is a technique for allowing sharing of an IP address by multiple endpoints within a subnetwork. NAT uses DHCP to assign local addresses to endpoints within a “NATed” subnetwork that cannot in general be reached by datagrams sent from outside. The NAT-aware router then translates local addresses to use a single externally reachable source IP address on TCP connections initiated by clients within the NATed subnet. UDP protocols can also be supported.

The ability of a router to interpose itself between end points in a NATed subnetwork and external servers allows the semantics of TCP connections initiated from within the subnetwork to match the specification of the non-NATed network. The most common cases are connections between a Web browser or other client within the network and external servers. However, connections from outside the NATed subnet to endpoints within it are not possible without additional administrative intervention.

Viewing the Internet in terms of the hourglass model, adding NATed subnetworks to the implementation is a weakening of the IP spanning layer. The global reachability condition that datagrams can be sent from any sending endpoint to any receiver’s IP address does not hold in the NATed Internet. This breaking of symmetry in reachability is often viewed as a weakness of NAT.

In spite of arguments against it, NAT has become ubiquitous in the consumer Internet. While NAT does solve a problem with scarcity of IPv4 addresses, there are other ways to allow sharing of a single IP address by many nodes, some of which maintain symmetric reachability. Our analysis suggests the logical weakness of the NATed Internet’s design may in fact help to explain its greater deployment scalability.

By abandoning symmetric reachability, the NATed Internet trades-off a logically weaker spanning layer against an expanded class of possible supports. This comes at the expense of excluding some possible applications that require global reachability (such as pure peer-to-peer systems). The exclusion of some applications has been generally acceptable to the community of commercial Internet users, sometimes using workarounds created by the providers of commercial peer-to-peer services that require general reachability.

Users of applications that require symmetric reachability have responded by working within a separate community of interoperability, sometimes connecting to non-NATed networks such as those at many universities and research laboratories using Virtual Private Networks. This bifurcation is made more acceptable by the fact that most home and business users do not require global reachability. In this analysis, the broader support possible for NAT has overcome resistance due to violations of layering and lack of symmetric reachability.

Process creation in Unix. In early operating systems it was common for the creation of a new process to be a privileged operation that could be invoked only from code running with supervisory privileges. There were multiple reasons for such caution, but one was that the power to allocate operating system resources that comprise a new process was seen as too great to be delegated to the application level.

Another reason was the power of process creation (for example, determining the identity under which the newly created process would run) was seen as too dangerous. This led to a design approach in which command line interpretation was a near-immutable function of the operating system that could only be changed by the installation of new supervisory code modules, often a privilege available only to the vendor or system administrator.

In Unix, process creation was implemented by the `fork()` system call, a logically weaker operation that does not allow any of the attributes of the child process to be determined by the parent, but instead requires that the child inherit such attributes from the parent.⁹ Operations that changed sensitive properties of a process were factored out into orthogonal calls such as `chown()` and `nice()`. These were fully or partially restricted to operating in supervisory mode or integrated with `exec()` (which is not so restricted) using `chmod()` and the set-user-ID bit. The decision was made to allow the allocation of kernel resources by applications, which allows the possibility of “fork-bomb” denial of service attacks.

The result of this design was not only the ability to implement a variety of different command line interpreters as nonprivileged user processes (leading to innovations and the introduction of powerful new language features) but also the flexible use of `fork()` as a tool in the design of multiprocess applications. This design approach has allowed the adaptation of kernels that implement the Unix-based POSIX standard to run on mobile and embedded devices that could not have been anticipated by the original designers.

Caching metadata in HTTP. The World Wide Web established HTTP as a near-universal protocol for accessing persistent data objects using a global namespace (commonly referred to as the REST interface). This general use of HTTP has created a community of interoperation that has adopted it as a spanning layer.

The original specification of the HTTP protocol did not include any requirement of consistency in the objects returned in response to independent but identical HTTP requests. However, in the common case where

HTTP responses are based on a collection of stored objects they exhibit stability over time and consistency between clients. Temporal stability is the basis of caching implemented in Web clients and additional consistency between different clients enabled shared Web caching.⁴ However, this stability is not perfect and in particular does not hold for dynamic HTTP responses that are the result of arbitrary server side computation. This can result in the return of stale cache responses.

By using the HTTP Cache-Control header directives in an HTTP response, the server can declare the extent of temporal stability, stability across clients, or the complete lack of stability in that response. If servers respect the stability guarantees declared in Cache-Control directives, Web caches can use them to ensure correctness of their responses.

Viewed as a service specification, HTTP with a requirement for accuracy in Cache-Control directives is logically stronger because it enables accurate assertions to be made regarding the correspondence between such metadata and server responses. In terms of the Hourglass Theorem, the weakness of the less constrained interpretation of HTTP without accurate caching metadata allows for looser implementations. This is traded off against the ability to support applications that require consistency in HTTP responses.

In practice, the ease and cost savings of ignoring consistency of lifetime metadata in server content management has generally won out over the ability to support applications requiring consistency. While Web browsers do take advantage of temporal consistency, they also sometimes return stale responses and require end users to intervene manually. The popularity of shared HTTP caches has been hampered by their inability to ensure consistency. The inefficiency of uncached HTTP in delivering stable responses has largely been countervailed by the trend toward increasing bandwidth in the Internet, although it is a significant factor inhibiting the deployment scalability of the Internet in parts of the world where network bandwidth is highly constrained.

Designing a spanning layer for node services. Network architects have long sought to define an interface to enable interoperation in the creation of new

services using the generalized local transfer, storage and processing services of network intermediate nodes. Examples of such efforts include active networking, middleboxes, the computational grid, PlanetLab, and GENI, as well as current efforts at defining containers for computational workloads. A full survey is beyond the scope of this article.

Nodes that comprise such general networks are variously characterized as virtual machines or programmable routers. A standard interface to local node services would act as a spanning layer defining a community of interoperability in service creation. Many current proposals for such a standard define spanning layers that are logically strong, for instance, allowing for the guaranteed reservation of resources.

The Hourglass Theorem can be the basis for an argument that such a spanning layer should be chosen so as to be minimally sufficient for a set of necessary applications in order to maximize the number of possible supports.² If we accept the DST as a more general design rule, then simplicity, generality, and resource limitation should also be maximized.

A review of current proposals may reveal an acceptance of strong assumptions, complexity, specialization, and unbounded resource allocation as “necessary.” If so, the DST suggests such designs may suffer diminished deployment scalability, which can be detrimental in any standard so vital to the future of global information infrastructure.

Conclusion

This article is intended as a first step in a research program to devise a common language for analyzing the design of spanning layers in layered systems of all kinds and predicting the outcomes of such designs. The primary technical contribution is the formulation of a layered system of service interfaces in terms of program logic. This yields a definition of “deployment scalability” that seeks to capture the intent of the hourglass model.

The further discussion of other aspects of the thin waist is intended to capture some of the informal arguments that have been made about the design of the spanning layer. The Deployment Scalability Tradeoff is a general design principle intended to fulfill a role in ar-

guing for thinness. All aspects of this characterization seem ripe for further formalization and refinement.

Acknowledgments

Thanks to my long-time collaborator Terry Moore for his encouragement and philosophical dialog, to my comrade Tim Griffin for his critical support, and to Jerry Saltzer, Joe Touch, Rick McGeer, Bob Harper, Glenn Riccart, Elaine Wenderholm and the reviewers for their insightful comments.

This work was performed under financial assistance award 70NAN-B17H174 from the U.S. Department of Commerce, National Institute of Standards and Technology. 

References

1. Akhshabi, S. and Dovrolis, C. The evolution of layered protocol stacks leads to an hourglass-shaped architecture. In *Proceedings of the ACM SIGCOMM 2011 Conference*. ACM, New York, NY, 206–217.
2. Beck, M., Moore, T., Luszczek, P. and Danalis, A. Interoperable convergence of storage, networking, and computation. *Advances in Information and Communication. Lecture Notes in Networks and Systems 70*. Springer, 2020, 667–690.
3. Cardelli, L. A semantics of multiple inheritance. *Information and Computation*. Springer-Verlag, 1988, 51–67.
4. Chankhunthod, A., Danzig, P.B., Needaels, C., Schwartz, M.F. and Worrell, K.J. A hierarchical Internet object cache. In *Proceedings of the 1996 USENIX Technical Conference*, 153–163.
5. Clark, D.D. Interoperation, open interfaces and protocol architecture. *The Unpredictable Certainty: White Paper*. The National Academies Press, Washington, DC, 1997, 133–144.
6. Fagg, G., Moore, T., Beck, M., Wolski, R., Bassi, A., Plank, J.S., and Swamy, M. The Internet backplane protocol: A study in resource sharing. In *Proceedings of the 2nd IEEE/ACM Intern. Symp. Cluster Computing and the Grid*.
7. Foster, I., Kesselman, C. and Tuecke, S. The anatomy of the grid: Enabling scalable virtual organizations. *The Intern. J. High Performance Computing Applications 15*, 3 (2001), 200–222.
8. Peterson, L.L. and Davie, B.S. *Computer Networks, A Systems Approach, 5th Edition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011.
9. Ritchie, D.M. and Thompson, K. The Unix time-sharing system. *Commun. ACM 17*, 7 (July 1974), 365–375.
10. Saltzer, J.H., Reed, D.P. and Clark, D.D. End-to-end arguments in system design. *ACM Trans. Comput. Syst.* 2, 4 (Nov. 1984), 277–288.
11. Shilton, K., Burke, J., Zhang, L. and Claffy, K. Anticipating policy and social implications of named data networking. *Commun. ACM 59*, 12 (Dec. 2016), 92–101.

Micah Beck (mbeck@utk.edu) is an associate professor in the Department of Electrical Engineering and Computer Science at University of Tennessee, Knoxville, and is currently with the Office of Advanced Cyberinfrastructure in the National Science Foundation. The work discussed here was completed prior to his government service and does not reflect the views, conclusions, or opinions of the National Science Foundation or of the U.S. Government.

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



Watch the author discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/on-the-hourglass-model>

DOI:10.1145/3329674

An innovative, entry-level informatics course enables students to ponder CS problems in different ways, from different perspectives.

BY CHRISTOPHER FRAUENBERGER AND PETER PURGATHOFER

Ways of Thinking in Informatics

WITH THE INTIMATE entanglement of digital technology with humans and their social way of being, computer science has changed. While some of the problems we deal with are (still) well defined and mostly computationally solvable, many problems are now found to be wicked and ill defined. People and technologies are now part of an interwoven socio-material web in which humans are not the only actors anymore. This pervasive complexity raises challenges for computer scientists and technologists that go well beyond of what could be addressed by a traditional understanding of engineering the most efficient computational tools. It requires us to rethink what must be the core competencies of future computer scientists. New skills stemming from the social sciences or philosophy need to complement engineering skills to create digital technologies within lived experiences. With it comes a major shift in responsibility. In a *New York Times* article,

Farhad Manjoo argued why 2017 could be seen as a turning point for big technology companies as they “began to grudgingly accept that they have some responsibility to the offline world.”^a Technologists, whether working in dominating corporations, small start-ups, or within academia, can no longer pretend they only solve tech problems. They are required to engage in a moral discourse as most of their products or results are essentially social interventions.

So how can we facilitate such a shift in the thinking as well as in the culture? In a recent workshop we co-organized on “Values in Computing,” a group of leading experts in the field of human-computer interaction discussed this question. The outcome was distilled into the Denver Manifesto,^b which calls for a shift in the education of future researchers and practitioners to ensure they can not only write software, but are also critically reflecting on their moral positions and their contribution to society.

With this article, we want to report on our efforts to respond to this call and discuss a new entry-level course for students of informatics at TU Wien, Austria.^c Its

a <https://nyti.ms/2Ar47w9>

b <http://www.valuesincomputing.org/the-denver-manifesto/>

c <http://informatik.tuwien.ac.at/English>

» key insights

- **Computer science has changed. Technology is now intimately entangled with people and society, which requires us to rethink how we equip students to shape our technological futures.**
- **To counter narrow conceptions of problem solving, we present a CS course that offers students the diverse lenses through which we can engage with technology. Ways of thinking include scientific, computational, designerly, critical, creative, responsible, economical and criminal thinking.**
- **As a first-year course of a computer science program, we reach 700+ students each year. Despite the logistical challenges, we argue that it is a great opportunity to empower so many of our students to see everything they hear subsequently from more than one side.**



core premise is to enable students to think about problems in computer science (CS) in different ways and from different perspectives. The aim is to plant the seed of critical reflection and equip students with the intellectual tools to see everything they hear subsequently as part of something bigger. Consequently, we have called this new course “Ways of Thinking in Informatics.”

We have developed the syllabus over the course of one year and introduced it to over 960 bachelor students in the fall semester 2017. Here, we describe the context in which this course took place, discuss related literature, the rationale for its structure and fundamental concept, report on some of the practical challenges of teaching such a high number of students, and reflect on our experience from this first year.

Context and Inspiration

TU Wien is the largest technical univer-

sity in Austria with close to 30K active students. The Faculty of Informatics, with over 5K students, is the second largest in the university with an annual intake of 580 students. We offer five Bachelor programs with foci on visual and human-centered computing, medical informatics, software and information engineering, computer engineering, and business informatics. There is substantial overlap between these in foundational courses, particularly within the first semesters, covering programming, mathematics, logic, algorithms and software engineering. As the background of our students varies, depending on the type of school they attended, these foundational courses establish a common ground and basic knowledge to be built on. Ways of Thinking in Informatics became one of the compulsory courses for first-year students of all programs. This means an annual cohort of 700–1000

students taking this course every fall as additional students from other studies register for our course and some are repeating the course.

The faculty decided to introduce this course as an orientation and to highlight the diversity and potential of CS in a rapidly changing world. We were inspired by UC Berkeley’s introductory course “The Beauty and Joy of Computing,”^d which was one of a series of pilot courses that aimed to teach CS’s big ideas and the most important computational thinking practices.²⁰ However, we also felt the focus on computational thinking maybe is only part of a bigger picture, which is how we set out to build a curriculum that emphasized the diversity of thinking styles. Importantly, we also wanted to follow our university’s leitmotiv “Technik für Menschen” (Technology

^d <http://guide.berkeley.edu/search/?P=COMPSCI%2010>

for people) and situate these thinking styles within the context of society.

The need for including aspects of societal impacts and ethics into CS curricula was laced into the ACM/IEEE CS Curriculum 27 years ago.²² However, as Goldweber et al. note, the uptake and implementation of such themes in the CS curriculum has been slow. In their survey, they point out that while most institutions teach relevant topics, most narrowly focus on ethics or computing history and teach them relatively late and detached from other topics.⁴ Considering the recent surge in relevant courses being taught at many universities,^e this may be slowly changing in response to the public discourse around societal impacts of digital technology.

However, Ways of Thinking does not aim to be a course about societal aspects of CS. Rather, we argue CS is inherently social and no social aspects can be meaningfully separated from CS. Consequently, we teach very fundamental concepts such as abstraction, cryptography or complexity theory alongside and interwoven with algorithmic bias, the historical role of women in programming, or questions of privacy. This resonates with Skirpan et al., who recently argued that ethics education in CS must be continuous, in-situ, and perspectival.¹⁹ In other words, talking about ethics must be part of talking about CS, which in turn requires the ability to take multiple perspectives on computing problems.

Ways of Thinking

The key premise of the course is to enable students to apply different lenses on problems of CS, situated in the world. Each such lens, or way of thinking, frames these problems differently, provides different theoretical foundations, and brings with it a unique orientation for questions and methods through which they can be explored. The sequence of Ways of Thinking is designed to be semi-historical and somewhat follows a logic increasing complexity. However, we emphasize that these ways of thinking are neither qualitatively ranked, nor mutually exclusive. In fact, the final assignment to students



The aim [of this course] is to plant the seed of critical reflection and equip students with the intellectual tools to see everything they hear subsequently as part of something bigger.



asks them to analyze one given context through multiple lenses with the aim to reveal the benefits of applying different ways of thinking at the same time.

We also acknowledge that the range of thinking we cover in this course is neither complete nor the only possible way of categorizing the perspectives one could take. In fact, we would encourage educators in the field to develop other ways of organizing such a course that teaches perspective-taking as a prerequisite for becoming a reflective practitioner or researcher. Based on initial feedback and on our experience teaching the course, we believe that our choices have been effective in this respect.

Here, we walk through 10 different ways of thinking and have included three cross-sectional topics that are interwoven with them. We also provide brief accounts of all chapters of the course.^f

► **Pre-scientific thinking.** One of the initial ideas was to make a course that can be understood as an “applied” philosophy of science lecture, suitable for first semester students. In order to understand the enormous changes brought about by the scientific revolution, we start with a brief introduction to alchemy as an example for pre-scientific thinking. We show that one of the key differences between alchemy and science was openness, or more specifically, the lack thereof. Alchemy was based on secrets, hermeticism, and isolation. One of the key insights enabling the scientific revolution was the realization that open exchange and critical discourse should replace the secrecy and isolation of alchemy. We also discuss the interesting parallel that academic patenting and privately funded research, especially in informatics, has recently increased the need for secrecy, with unknown side effects, for example, in terms of accountability and knowledge production.

► **Scientific thinking.** The scientific revolution was a fundamental shift that empowered people to know for themselves, independent of the doctrine sanctioned by the ruling elite or the church. At least in principle, the

^e See, for example, the collection of ethics related courses started by Casey Fiesler; <https://bit.ly/2IZ2L3O>

^f A full English version of the syllabus with additional resources is available at <https://wot.pubpub.org>

early modern period postulated that knowledge could be reproduced and validated by anyone following the scientific method. We discuss early experiments, such as Galileo dropping balls from the inclined tower of Pisa as well as classic experiments in CS, such as Fitts' law, to show how these produced widely accepted knowledge of the world. Historically, the "Age of Reason" or Enlightenment led to major shifts in Western society that in parallel became the source for the modern, technological optimism still with us today (Whatever the problem, *there is an app for that*). We discuss early utopias and dystopias, such as the 1920s film *Metropolis* and Orwell's *1984*, connecting them to current narratives around social robots, big data surveillance, or how gender roles are inscribed in visions.

Subsequently, we introduce the work by Kuhn⁸ on scientific revolutions. As an example for the notion of paradigm shifts, we discuss the three "waves" in human-computer interaction (HCI), starting with the classic human-factors approach, leading to the cognitive science perspective, and to situated and embodied HCI.⁶ This leads us to the main ontological and epistemological positions of (post-) positivism, critical theory and constructivism as philosophical orientations towards science.⁵ Importantly, we develop a view on what doing "good science" means within these paradigms and highlight the link between one's paradigm stance and the possible questions that could be asked about a problem.

► **Mathematical thinking.** Undergraduate informatics curricula often incorporate a sizeable portion of mathematics. In most of our curricula, 25% of all first-year credit points come from math courses. Compared to high school, mathematics at the university level is less concerned with fluency in arithmetic. Instead, it requires students to understand the value of mathematics in problem solving, and the mathematical way of looking at the world. Some of the core concepts are abstraction, induction and deduction, recursion and, above all, the idea of provability.

In the typical mathematics course, the value of these concepts gets bur-

ied under an overwhelming avalanche of practical exercises and theoretical test taking. We show students the ideas behind those concepts and why they make sense in the specific perspective of mathematical thinking. For example, by understanding the difference between a proof and mere evidence, students can see the unique benefit they get from approaching problems with the toolset of mathematics. At the same time, they can see the price they have to pay when abstracting complex real-world problems full of interdependencies and inherent contradictions to come to mathematical expressions.

► **Computational thinking.** In recent years, the use of computers has enabled impressive advancements in many scientific disciplines. An example often cited is the sequencing of the human genome. The specific method that enabled this breakthrough was unthinkable before computers became commonplace. Put differently, using computers to scale up mathematics creates new possibilities for solving problems in ways that were inconceivable without computers. The idea behind this perspective became known as "computational thinking."²⁴

In the course, we discuss examples like the one mentioned here. We stage a session of live coding where the difficult to understand solution to the Monty Hall problem¹⁸ becomes accessible not only through running a computer simulation, but more importantly by carefully reading the code itself. Following this line of argument, we explore aspects of code as knowledge representation. Starting with the cognitive developmental stages of code understanding by Lister,¹¹ we show that code can be much more than instruction to a machine; it represents knowledge and can even be used to make an argument. Again, this offers an additional layer of meaning for students to think about code and coding, which they will be doing a lot during their studies.

► **Design thinking.** The term design thinking has come to carry a very specific meaning in recent years, mostly associated with Stanford's Hasso Plattner Institute of Design. Our interpretation differs as it relies more on contemporary design theory literature such as Bryan Lawson's *How Designers*

Think,⁹ Henrik Gedenryd's *How Designers Work*³ or Bill Buxton's *Sketching User Experiences*.² In particular, we see design as an open process, representing a stark contrast to the rationalistic and deterministic process models of traditional engineering.

For many students, this way of thinking bears a particular challenge to accept and understand. On one hand, they can appreciate the value of good design from successful products on the market, with Apple being the obvious model example, for better or for worse. This creates an interest in design, insofar as they see it as a crucial element of their future work. On the other hand, to understand the consequences of the "wicked" nature of design problems¹⁶ is more difficult to embrace, as it questions the traditional problem solving strategies of informatics as well as of engineering in general. Our goal is to foster an understanding that most problems are not "given" or even defined apriori, but rather emerge from an interwoven process of problem solving and problem framing that requires a special way of thinking.

► **Lateral/creative thinking.** Both the rational traditions of mathematical and computational thinking, as well as the open and sometimes subjective approaches in design thinking, need lateral or creative thinking. Creative problem solving has been seen as a central skill of engineers throughout history, even if there is no consensus on what constitutes creative thinking. As school historically leaves little room for creativity in places other than art, the expectations of our students are shaped accordingly. We try to break this association with the context of artistic practice by exploring unifying characteristics of creativity across all fields. Based on this discussion we show factors that stimulate or hinder creativity, as well as the areas of informatics where creativity needs to be given room to emerge. For example, many engineers pride themselves of a quick grasp of feasibility when presented with an idea. While this is certainly a useful skill in some situations, it also has the power to quickly shut down creative, outside-the-box thinking that allows new and

novel ideas to emerge. This leads us to discuss the concept of reflective practice by Schön¹⁷ and how reflecting on and in action can help to continuously evolve the practice of students, be it in terms of their learning, their research, or within their chosen professional path.

► **Critical thinking.** The value of critical thinking is undisputed, but there is a lot of discussion about how critical thinking can be taught (for example, see Willingham²³). By focusing on cognitive bias and logical fallacies, we provoke an analytical approach to the matter. We challenge students to identify biases and incoherent argumentation in themselves and in their colleagues and to support each other in this process. Approaching critical thinking from this rather abstract perspective creates a segue into another phenomenon that has seen a great deal of discussion recently: algorithmic bias.¹³ We discuss the problems we create when we attribute objectivity to algorithms and data, implicitly suggesting that computers can come to objective decisions; but data as well as algorithms usually embed and exacerbate bias in one form or another as many popular examples demonstrate.

► **Economic thinking.** Startups have become the de-facto role model for the post-academic career of many students. This chapter includes a critical discussion of the societal and economic influences of startup culture, as well as a short introduction into the vocabulary of the startup economy. For logistical reasons during the first year, this chapter was confined to this narrow aspect of economic thinking. The intention is, however, to expand this chapter to discuss business/value models and cost-benefit analysis and their relation to what software gets built and how this mirrors societal visions.

► **Responsible thinking.** The computing field increasingly recognizes its responsibility within society, as technology becomes a constituent part of today's world. To deflect this responsibility by stating "I was just the scientist" will not be good enough.¹⁰ From this central argument, we first introduce the three main positions in moral philosophy: virtue ethics, deontology, and consequentialism, and we demonstrate how ethical judg-

ments shift when these are applied to a controversial social media study.⁷ We discuss the notion of values and their sources and how they relate to beliefs and data-driven psychological profiling that receives so much attention currently.

We then explore two main areas of moral import for future computer scientists: ethical conduct in science, and responsible research and innovation (RRI). Going through classic examples such as the Milgram experiment and the Tuskegee study, and linking back to the Nuremberg trials, we derive some fundamental principles such as informed consent, respect, fairness, or judging the balance between knowledge gain and risk to participants. Within a broader picture, we pick up the RRI framework implemented by the European Union²¹ to discuss central pillars for a reflective and responsible practice.

► **Criminal thinking.** One of the central problems of a society steeped with technology is that every innocent little software bug potentially opens a malicious backdoor into software. During recent years, this has enabled a new class of criminal behavior. In this chapter, we try to show the many aspects of criminal creativity when looking at technology. The goal is to provoke an understanding of the importance of security and privacy by design in the sense of Bruce Schneier's perspective. He describes security as a complex affair best characterized through an interplay of technological and psychological factors. Students learn to see that creating secure technology for real-life situations necessitates looking beyond the technology itself and consider context, people, social dynamics, etc.

Cross-sectoral topics. In addition to the main ways of thinking chapters, three cross-sectional topics are included: History of computing, computers and society, and gender and diversity in informatics. We see the history of computing as a necessary foundation in order to understand the discipline, helping students to make sense of the discourse around current trends and issues. In our presentation, we emphasize the history of thought over the history of technologies or the history personalities as, for example, discussed in the works of informatics historian Jörg Pflüger.^{14,15}

The contents for computers and society are selected from current developments and press coverage of science, practice and politics concerning informatics. We use a weekly recurring format of the best and worst of informatics where we present the most interesting and encouraging stories, as well as scary news from scientific literature and news. The selected stories are discussed using the perspectives and concepts of the different ways of thinking covered so far. Questions of privacy, surveillance, copyright, and security are popular themes.

Additionally, areas of tension from society and technology as well as aspects of gender and diversity are interwoven whenever there is opportunity to do so. For example, in economical thinking we discuss the toxicity of the "bro" culture in startups that became evident in scandals surrounding Uber;⁸ in critical thinking we broach the issue of the manipulation of Google's search result sorting in the wake of the U.S. election;^h in criminal thinking we discuss the (ab)use of data from social networks to target vulnerable teenagers with ads.ⁱ

As with the core chapters in this course, we aim to relate fundamental issues, principles, or theories to recent points of discussion in the general public, which in turn creates a meaningful link to the lived experiences of students.

Exercises and Evaluation

For most of the chapters mentioned here we provide an assignment that students work on individually or in groups and hand in online. Each assignment typically comprises a series of tasks that include formulating responses to discussion items on the basis of online research, reading scientific articles, conducting interviews, and/or doing some practical work such as designing an infographic, conceptualizing a user interface, or play a learning game. Each assignment ends with a task in which students are asked to reflect on their learning outcomes. To facilitate the assignments, we use a format we developed over the last few years (for example, see Luck-

g <https://bit.ly/2GXnWAG>

h <https://bit.ly/2gUyN5>

i <https://bit.ly/2pBaiv6>

ner and Purgathofer¹²) that allows students to choose from multiple alternative exercises across each chapter, staged sequences of tasks, and double-blind peer reviewing among students as a way to learn how to offer and appreciate criticism. The evaluation of the work handed in by students individually as well as the quality of reviews they write makes up for 65% of their final grade.

The double-blind peer-reviewing aspect of their evaluation can also be seen as a constructive alignment¹ with the Ways of Thinking in Informatics. One of the learning outcomes is the critical reflection of students' own practice; they write and receive reviews that critically reflect on their own work as well as the work of other students; additionally we practice critical reflection during the lectures based on their input within the best and worst of informatics format.

The remaining 35% of their grade comes from the evaluation of a group work project where students analyze and discuss a speculative video about technology^j from the different perspectives offered by the course. Each member of a team of three or four students selects one of the main chapters of the course and discusses the content of the video through the chosen way of thinking. We support this by offering a number of lead questions for each chapter. The group then meets, debates commonalities and conflicts between the different perspectives, and documents the individual perspectives as well as the outcome of the discussion in a common paper. This paper is handed in, graded, and discussed with a tutor in a brief meeting.

Slightly more than three quarters of the students successfully completed the course. Of those 23% who failed (193 of 845), all dropped out during the semester and did not complete all challenges. While the formal course evaluation survey has not drawn many responses,^k the final challenge included a task that probed for students' overall experience and reflections. Alongside with infor-



Computer science is inherently social and no social aspects can be meaningfully separated from computer science.



mal feedback in the classroom, these painted a rather positive picture of the course's reception.

Discussion and Reflection

One of the goals we pursue with this course is to offer students tools and structures to help them make sense of the rest of their studies. Based on a largely constructivist learning theory, we believe that what you learn is to a great extent determined by the diverse and holistic ways you are enabled to think about a subject matter. Quotes such as these suggest this was on offer:

It gave me a good overview and served as a reminder to critically engage with future content in my studies. (Final reflections, translated)

By seeing larger ordering principles, students are invited to build cognitive equivalents of shelves or drawers for future knowledge to be organized by. For example, by exposing the inherent meaning of some mathematical terminology, we offer a new layer of meaning for students to organize what they learn in their mathematics courses. If they can appreciate the special nature of the mathematical proof, they can understand its value in the implementation of dependent systems.

We offer students different ways of thinking in informatics that can become ways to look at problems, ways to ask questions, ways to see deficits in the narrow and one-dimensional approaches we often find in overspecialized subject areas. In effect, we want to enable students to develop a reflective practice that suggests taking a step back from focused learning goals in an attempt to see the bigger picture. This was also perceived as a skill being taught in this course:

Importantly, constantly being asked to reflect on content will be valuable skill for my studies and future career. (Final reflections, translated)

Such a practice affords connecting knowledge at different levels and from different perspectives that will ultimately be the key skill for future technologists, whether in research or industry, to tackle the unknown challenges of the future. This quote from one of our students speaks to this:

Initially, I could not figure out what I have learnt from this course ... Later,

j "Uninvited Guests" by Superflux Lab; <http://superflux.in/index.php/work/uninvited-guests/>

k Low numbers of responses in these surveys are the norm at our university.

with more reflection, the value became more apparent and with the final challenge I really realized what I can take away from it—much more than I thought. (Final reflections, translated)

Another goal of the course was to help students in understanding the rationale for our curriculum. Ways of Thinking in Informatics is part of an introductory/orientation phase of the program. By offering apriori meaning for many of the courses they visit later, we supply an opportunity to see purpose in the curriculum.

We ran this course in its entirety for the first time during winter semester 2017. With over 800 registered students, it has been a tremendous challenge, not only to design the content and the pedagogical approach, but also to find innovative solutions to the logistics of teaching such large numbers of students. We can confidently say this experiment has been a success as evidenced by the largely positive feedback we received from students. What the longer-term impact is—that is, the ways in which we enabled students to think differently about what they will learn in the remainder of their studies—remains to be seen, but we are inspired by the students' engagement with this course, and are hopeful that it has created a new quality of foundations to their studies.

As computer scientists and educators, we also are humbled to be given the opportunity to plant this seed in so many students at a crucial juncture of their development—at the start of their studies. While the task to teach a new introductory course in informatics to hundreds of students hardly ever draws many volunteers among a faculty, our experience was that it results in no small gratification to have made a, maybe small, but significant difference in shaping what so many future technologists see as their role in the world. 

We invite everyone to leave inline comments on any part of the full syllabus at <http://wot.pubpub.org>

References

1. Biggs, J. Enhancing teaching through constructive alignment. *Higher Education* 32, 3 (1996), 347–364.
2. Buxton, B. *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann, 2007, ISBN 978-0123740373.



We believe that what you learn is to a great extent determined by the diverse and holistic ways you are enabled to think about a subject matter.



3. Gedenryd, H. How designers work—Making sense of authentic cognitive activities. Ph.D. thesis. Lund University, 1998.
4. Goldweber, M. et al. Enhancing the social issues components in our computing curriculum: Computing for the social good. In *Proceedings of the 2010 ITICSE Working Group Reports*. ACM, New York, NY, 2010, 117–133, ISBN 978-1-4503-0677-5.
5. Guba, E.G. and Lincoln, Y.S. Competing paradigms in qualitative research. *Handbook of Qualitative Research*. N.K. Denzin and Y.S. Lincoln, Eds. Sage Publications, London, U.K., 1994, 105–117.
6. Harrison, A., Tatar, D. and Sengers, P. The three paradigms of HCI. In *Proceedings of alt.chi*. ACM SIGCHI, 2007.
7. Kramer, A.D.I., Guillory, J.E. and Hancock, J.T. Experimental evidence of massive-scale emotional contagion through social networks. In *Proceedings of the National Academy of Sciences* 111, 24 (June 17, 2014), 8788–8790, ISSN 0027-8424.
8. Kuhn, T.S. *The Structure of Scientific Revolutions*. 2nd Edition. University of Chicago Press, 1970, ISBN 0-226-45803-2.
9. Lawson, B. *How Designers Think*. Routledge, 2005. ISBN 978-0750660778.
10. Light, A. et al. Special topic: Taking action in a changing world. *Interactions* 25, 1 (Dec. 2017), 34–45, DOI 10.1145/3169128.
11. Lister, R. Toward a developmental epistemology of computer programming. In *Proceedings of the 11th Workshop in Primary and Secondary Computing Education* (Münster, Germany, 2016), DOI: 10.1145/2978249.2978251.
12. Luckner, N. and Purgathofer, P. Exploring the use of peer review in large university courses. *IXD&A* 25 (2015), 21–38.
13. Nissenbaum, H. How computer systems embody values. *Computer* 34, 3 (2001), 120–119, DOI 10.1109/2.910905.
14. Pflüger, J. Konversation, manipulation, delegation: Zur ideengeschichte der interaktivität. *Geschichten der Informatik*. H.D. Hellige, Ed. Springer, Berlin, Heidelberg, 2004; https://doi.org/10.1007/978-3-642-18631-8_15.
15. Pflüger, J. Writing, building, growing: Leitvorstellungen der Programmiergeschichte. *Geschichten der Informatik*. H.D. Hellige, Ed. Springer, Berlin, Heidelberg, 2004; https://doi.org/10.1007/978-3-642-18631-8_12.
16. Rittel, H.W.J. and Webber, M.M. Dilemmas in a general theory of planning. *Policy Sciences* 4, 2 (1973), 155–169, DOI: 10.1007/BF01405730.
17. Schön, D.A. *The Reflective Practitioner: How Professionals Think in Action*. Basic Books, New York, 1983.
18. Selvin, S. A problem in probability (letter to the editor). *American Statistician* 29, 1 (1975), 67–71.
19. Skirpan, M. et al. Ethics education in context: A case study of novel ethics activities for the CS classroom. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. ACM, New York, NY, 2018, 940–945, DOI: 10.1145/3159450.3159573.
20. Snyder, L. et al. The first five computer science principles pilots: Summary and comparisons. *ACM Inroads* 3, 2 (2012).
21. Stilgoe, J., Owen, R. and Macnaghten, P. Developing a framework for responsible innovation. *Research Policy* 42, 9 (Nov. 2013), 1568–1580, DOI 10.1016/j.respol.2013.05.008.
22. Tucker, A.B. Ed. Computing curricula 1991. *Commun. ACM* 34, 6 (June 1991), 68–84; DOI 10.1145/103701.103710.
23. Willingham, D. Critical thinking: Why is it so hard to teach. *American Federation of Teachers* 31 (2007), 8–19; <https://doi.org/10.3200/AEPR.109.4.21-32>.
24. Wing, J.M. Computational thinking. *Commun. ACM* 49, 3 (Mar. 2006), 33–35, DOI 10.1145/1118178.1118215.

Christopher Frauenberger (christopher.frauenberger@tuwien.ac.at) is a senior researcher and principle investigator of Social Plays Technologies at TU Wien, Vienna, Austria.

Peter Purgathofer (peter.purgathofer@tuwien.ac.at) is an associate professor in the Institute of Visual Computing and Human Centered Technology at TU Wien, Vienna, Austria.

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

A Practitioner's Guide to the Natural Conversation Framework Conversational UX Design

This book adapts formal knowledge from the field of Conversation Analysis (CA) to the design of natural language interfaces. It outlines the Natural Conversation Framework (NCF), developed at IBM Research, a systematic framework for designing interfaces that work like natural conversation. The NCF consists of four main components: 1) an interaction model of "expandable sequences," 2) a corresponding content format, 3) a pattern language with 100 generic UX patterns and 4) a navigation method of six basic user actions. The authors introduce UX designers to a new way of thinking about user experience design in the context of conversational interfaces, including a new vocabulary, new principles and new interaction patterns. User experience designers and graduate students in the HCI field as well as developers and conversation analysis students should find this book of interest.

With recent advances in natural language understanding techniques and far-field microphone arrays, natural language interfaces, such as voice assistants and chatbots, are emerging as a popular new way to interact with computers. Today's platforms provide sophisticated tools for analyzing language and retrieving knowledge, but they fail to provide adequate support for modeling interaction. The user experience (UX) designer or software developer must figure out how a human conversation is organized, usually relying on commonsense rather than on formal knowledge. Fortunately, practitioners can rely on conversation science.

Robert J. Moore

Raphael Arar

ISBN: 978-1-4503-6302-0

DOI: 10.1145/3304087

<http://books.acm.org>



ACM BOOKS

Tracing the complicated yet still relatively unripe area of the Internet of Things search engine—from concepts, to classification, and open issues.

BY NGUYEN KHOI TRAN, QUAN Z. SHENG, M. ALI BABAR, LINA YAO, WEI EMMA ZHANG, AND SCHAHRAM DUSTDAR

Internet of Things Search Engine

ADVANCEMENTS UNDER THE moniker of the Internet of Things (IoT) allow things to network and become the primary producers of data in the Internet.¹⁴ IoT makes the state and interactions of real-world available to Web applications and information systems with minimal latency and complexity.²⁵ By enabling massive telemetry and individual addressing of “things,” the IoT offers three prominent benefits: spatial and temporal traceability of individual real-world objects for thief prevention, counterfeit product detection and food safety via accessing their pedigree; enabling ambient data collection and analytics for optimizing crop planning, enabling telemedicine and assisted living; and supporting real-time reactive systems such as smart building, automatic logistics and self-driving, networked cars.¹¹ Realizing these benefits requires the ability to discover and resolve queries for contents in the IoT. Offering these abilities is the responsibility of a class of software

system called the *Internet of Things search engine* (IoTSE).

IoTSE is a complicated and relatively immature research topic. The diversity of its solution space is, arguably, a primary challenge hindering its advance. Such diversity manifests itself in terms of the type of operations within an IoTSE instance (for example, discover content, index, and resolve queries), and the types of IoT content on which those operations are applied. Each combination of operation and content type represents a research area within the IoTSE literature with its own set of technical, social, and political issues. For instance, the IoTSE instances that discover and resolve queries on real-time sensing data from IoT-enabled sensors face the challenge of ensuring the “freshness” of data used for processing queries while minimizing the costly operation of pulling the data from sensors. IoTSE instances working with the actuating functionalities of IoT-enabled things, on the other hand, concern more with understanding the semantics of these functionalities. Due to the diversity of the IoTSE solution space and the lack of a shared vision of what IoTSE is and what it does, it is challenging to communicate the problems and the solutions related to this system.

>> key insights

- Any collection of information is only as useful as its information retrieval mechanism. Yet a comprehensive search engine is precisely the missing component of the Internet of Things (IoT). While some crawlers for IoT devices exist, an advanced IoT Search Engine (IoTSE) that can resolve queries for IoT content and based on IoT content is still beyond the horizon.
- Following an extensive review of the academic research efforts and industrial projects, this article proposes a meta-path model to describe IoTSE and discusses various IoTSE open issues that have emerged in the review.
- The conceptual model presented lays a foundation for the future integration of identified IoTSE visions.



The lack of such models and constructs for the communication of IoTSE inhibits more extensive research and development efforts that span research communities over an extended time, which are necessary for the advance of the IoTSE. As the existing studies on IoTSE have primarily focused only on technical issues related to a particular “IoTSE operation – IoT content type” combination, and as the existing reviews and surveys on IoTSE have primarily focused on a particular type of IoTSE, the lack of models and constructs to communicate and classify IoTSE, which are applicable to its diverse solution space, has not been addressed in the existing literature.

In this article, we introduce the fundamental concepts related to the functionality of an IoTSE instance and

a model called meta-path to provide a comprehensive yet succinct description of IoTSE instances by their functionality. We report a classification of IoTSE instances based on their meta-path description and present the representative IoTSE prototypes in each class. Finally, we discuss several open issues in the IoTSE research and development.

Methodology

The concepts and models presented in this article were generated from a structured and comprehensive study of the existing research works and industrial projects falling under the moniker of IoTSE. Our methodology was inspired by the systematic literature review method.⁸ It comprised four phases: detection, selection, extraction, and synthesis (Figure 1). The *Detection phase* involves identifying potentially relevant

articles from various academic sources. The *Selection phase* involves selecting a subset of articles that were high quality and relevant to the study. The *Extraction phase* involves extracting raw data relevant to the questions of the study. The *Synthesis phase* involves synthesizing raw data into knowledge to answer questions of the study.

Our method deviated from the systematic review method by using software tools for automation. Particularly, the detection and selection phase employed an in-house developed tool that queried academic data sources (that is, “primary search”) and retrieved articles that had been referenced by the articles detected in the primary search (that is, “snowballing”). We performed the primary search on various academic data sources, including the XML dataset of

DBLP, with the Boolean query “search OR discover *and* Internet of Things OR Web of things”. We assessed articles that emerged from the primary and snowballing search against the following selection criteria:

- ▶ Excluding papers that focus exclusively on physical and network layer.
- ▶ Excluding papers that focus on utilizing the sensing data from the IoT to extend the Web search
- ▶ Excluding the information retrieval papers that do not address IoT, *unless they are highly referenced by other relevant works*.

In the extraction phase, we extracted from papers the conceptualization, functionality, and internal operations of the reported IoTSE prototypes. Finally, we synthesized the extracted data into the concepts and models reported in this article.

By applying the reported method, we identified over 200 relevant works on IoTSE that span over a decade. Figure 2a compares the changes in the number of IoTSE-related works published and referenced between 2001 and 2016. The number of IoTSE works published each year has been increasing steadily since 2001. From 2009—the birth year of the IoT, this number has risen sharply and peaked at 38 works a year in 2014 and 2015. There was a drop in the number of published works in 2016, which we contribute to the fact that our prima-

ry study selection concluded by the end of that year and therefore missed the accepted-yet-unpublished works.

The changes in the number of referenced IoTSE works, however, have not assumed a similar pattern with the number of published works. Between 2001 and 2010, other works referenced most of the published works at least once. However, over the following six years, this number dropped gradually, and the gap between the number of published and referenced works widened. By 2015, only 13% of published IoTSE works received in-field citations.

Figure 2b compares the number of referenced IoTSE works and the number of in-field citations between 2001 and 2016 to provide more insights into the distribution of *attention* among the IoTSE literature. Despite fluctuations, the number of in-field citations rose steadily from 2001. After peaking at 67 citations in 2010, this number began to drop sharply. From these figures, we can see that a group of 29 works published between 2010 and 2012 received over 45% of the total number of in-field citation. This result might indicate that the perception of what an IoTSE instance is and what it should do have been driven by a subset of IoTSE works. A comprehensive analysis on the IoTSE literature and the internal operations of representative IoTSE prototypes is available elsewhere.²⁰

IoTSE Concepts

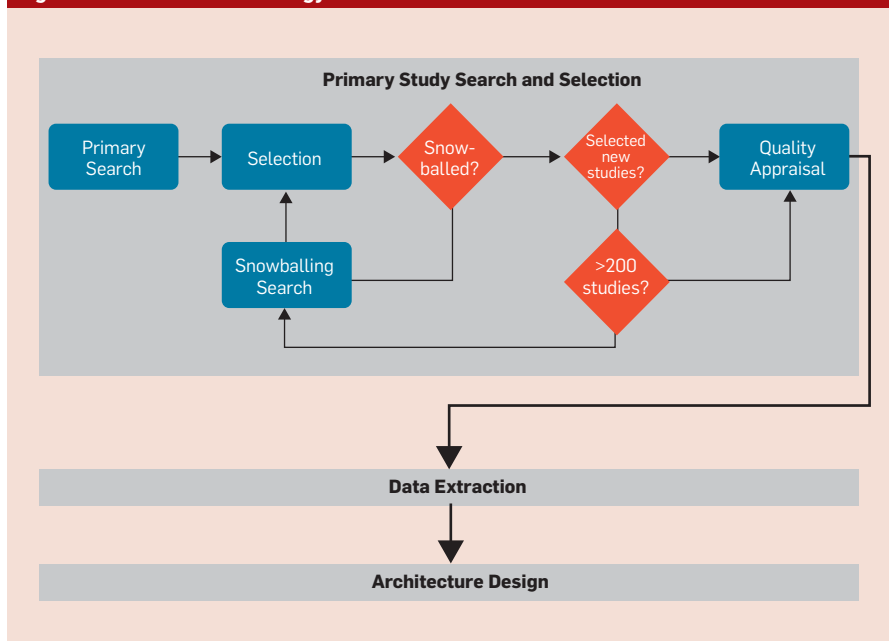
The Internet of Things comprises IoT *things*—physical objects enhanced with computing and networking capabilities and are potentially accessible via the Internet. For instance, a light bulb equipped with microcontrollers and wireless communication capability is an IoT thing that is commonly found in home automation applications. IoT things offer IoT *content*, such as the digital representation, data records, real-time sensor readings, and functionality that are offered by or related to things.

The IoT content appearing in the IoTSE literature can be organized into four types: representation, static information, dynamic information, and functionality. Figure 3 depicts four IoT content types of an IoT-enabled lightbulb. The representative content of the lightbulb comprises an HTML document that acts as a homepage of the light bulb for interacting with human users, and a JSON document that described the light bulb to machine agents. The dynamic information content of the light bulb denotes either the whole stream of energy consumption readings of the light bulb or the latest value in that stream. Due to the constant update of the lightbulb, these contents are “dynamic”. The static information content comprises the archived sensing data, the Web articles related to the lightbulb, and the records of its journey across supply chains. Finally, the functionality content includes *actuating services* that the lightbulb offers to alter its operation (toggling its power, changing its light color).

Discovery activity. An IoTSE instance processes queries on various collections of IoT content. When these collections are not available, an IoTSE instance must carry out the *discovery activity* to detect IoT content in a local or global scope, and optionally collect the content into its internal storage. More than 90% of the assessed IoTSE prototypes include discovery activities.

On a global scale, the content discovery problem can be framed as a Web crawling problem to identify a subset of Websites that serve IoT content (that is, IoT data sources) and retrieve the URI of the IoT content from those sites. In the existing literature, this crawling either relies on human’s guidance¹⁶ or the standard compliance of data source-

Figure 1. Research methodology.



es.¹² On a local scale, the content discovery can be addressed as a wireless discovery problem, in which an IoTSE instance either broadcasts beacon signals for things to register themselves, or detects and queries things directly to retrieve their content.^{22,26} Local discovery can also be addressed as a service discovery problem in local area networks, using technologies such as multicast Domain Name System (mDNS) or Bonjour.^a Semantic discovery is an alternative perspective on the content discovery problem. It concerns with detecting the semantics of IoT content and can be addressed by translation content to known data models.⁹

Search activity. The *Search activity* denotes the process of identifying a subset of discovered IoT content as search results of a given query. All assessed IoTSE prototypes covered this activity.

Formally, let c be an item of IoT content, and C be the collection of all content discovered by an IoTSE instance. For each query q , a set of contents c_q that are relevant to the query exists. The task of an IoTSE instance is to construct the result set $\sim c_q$ that approximates the unknown c_q by evaluating the relevance of each IoT resource against the given query with a relevance function $f(c, q)$. If the relevance function produces binary result, the process is considered *selection* or *lookup*:

Selection: $c_q = \{c \in C | f(c, q) = 1\}$,
where $f(c, q): C \times Q \rightarrow \{0, 1\}$

If the relevance function produces a real value, the result set contains resources whose scores are higher than a predefined threshold α . This process is called *resource scoring*.

Scoring: $r_q = \{c \in C | f(c, q) > \alpha\}$,
where $f(c, q): C \times Q \rightarrow \mathbb{R}$

The selection process cannot determine the degree of relevance of IoT content, and therefore can be considered less advanced compared to the scoring process. However, we discovered that nearly half of the analyzed IoTSE prototypes utilized selection. Most of the remaining prototypes scored IoT content based on its distance from a given query in a multi-dimensional space.

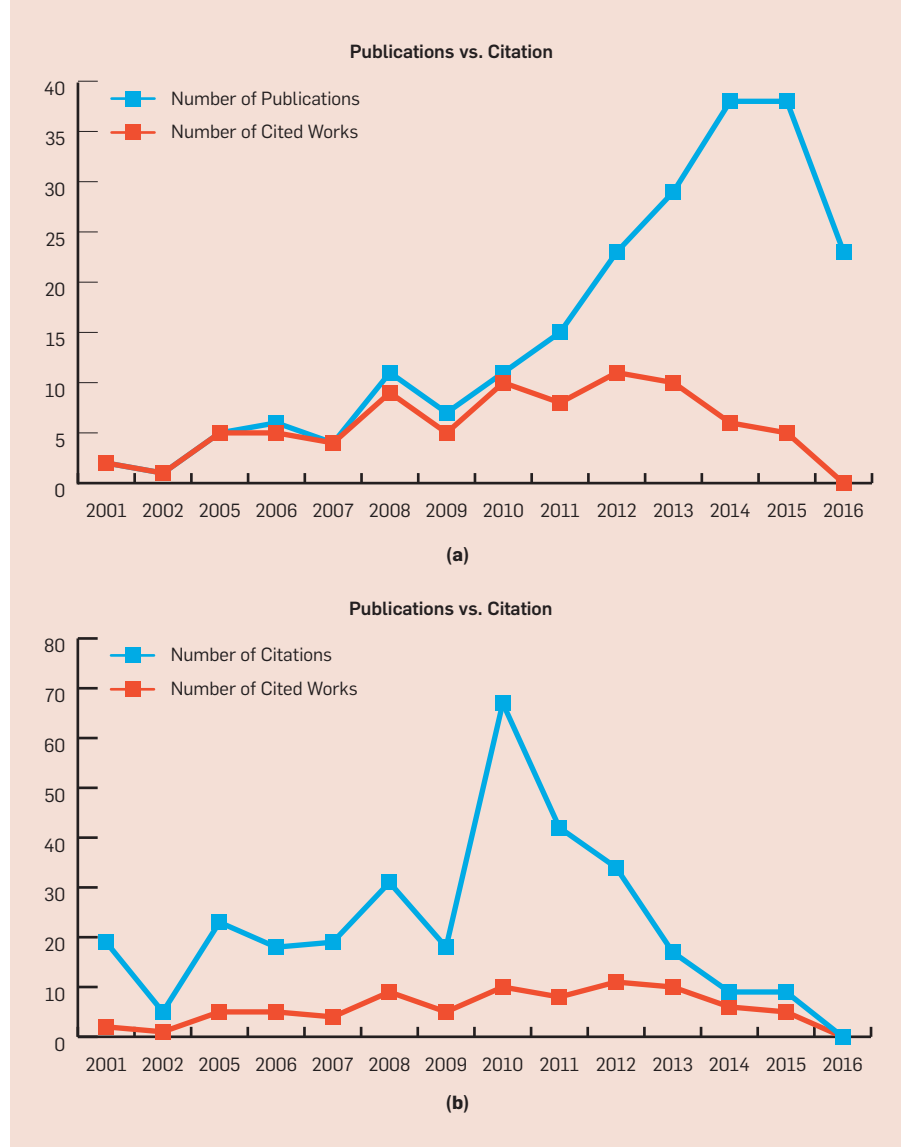
The storage and indexing of the discovered IoT content link the discovery activity with the search activity. Most of the existing IoTSE prototypes address the heterogeneity of IoT content by limiting the type and format of content and handle each type independently. For example, IoT-SVK² utilizes two B+ trees and an R tree index to address textual description, numeric sensing data, and location of things separately. Some IoTSE prototypes, such as DiscoWoT,⁹ address the heterogeneity problem by mapping various formats of IoT content onto a common format for processing.

Meta-path. The lack of a descriptive and comprehensive model to communicate and classify the functionality of IoTSE instances was a major problem

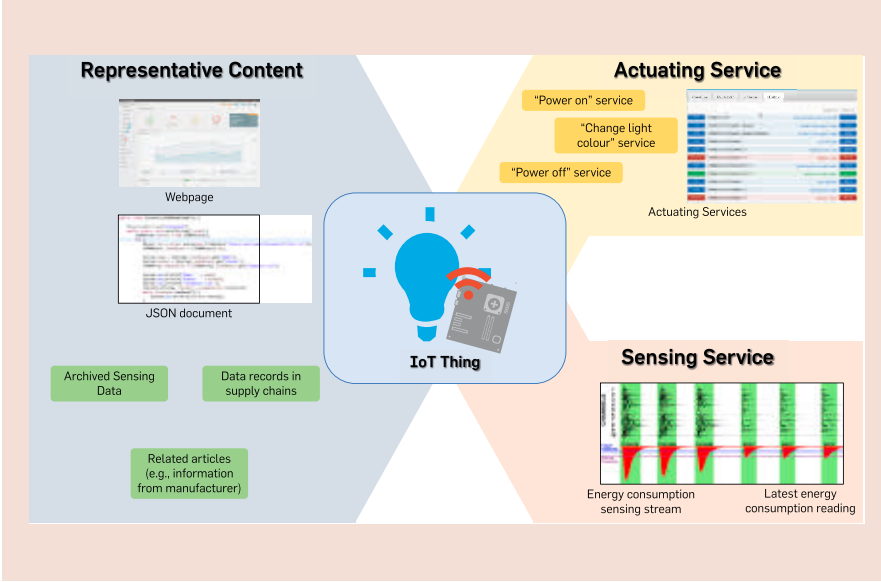
identified from our analysis. For instance, the term “object search” has been used to describe various types of IoTSE instances, which process queries on various types of content—real-time state, description, functionality of things—and return various types of IoT content including sensing data, location, data records, and actuating services of relevant IoT things.

The existing models describe an IoTSE instance either by the type of IoT content that is utilized for processing queries or returned as search results, without considering the relationship between them. Different from the previous types of search engine systems, such as Web search engines, IoTSE can utilize a combination of different IoT

Figure 2. Statistics regarding publication count and number of in-field citations of collected articles.



a <https://developer.apple.com/bonjour/>

Figure 3. Four types of IoT content of an IoT-enabled lightbulb.

finding all employees who had been in the meeting rooms that reported an abnormal energy consumption. However, we have not discovered IoTSE prototypes taking advantage of these correlations. Therefore, for the sake of clarity, we will not depict detailed thing-thing correlations in the following discussions.

Figure 4 depicts the IoT infrastructure in a smart building as a heterogeneous graph (upper) and the derived meta-graph. IoT things in this illustration consist of a smart light bulb, a meeting room, and a staff member who uses these facilities. The light bulb has seven IoT content items of four classes. The meta-graph captures relationships between IoT content types and things, as well as among things. For instance, two representative content items of the lightbulb are captured in the meta-graph as a single link between the representative content type and a thing.

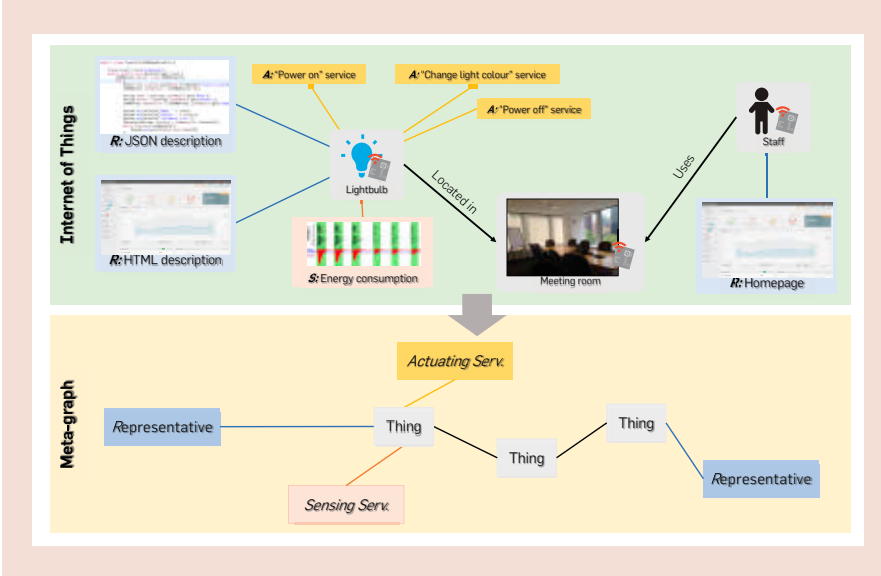
A meta-path is a sequence of edges on the meta-graph from one type of IoT content, through various IoT things, to another type of IoT content. In the IoTSE context, each meta-path can model the relationship between a type of IoT content used for assessing query and a type of IoT content used for deriving search results. By aggregating multiple meta-paths, we can model an IoTSE instances that utilize multiple types of IoT content.

A meta-path can be represented as follows:

$$(Query\ content\ type) \rightarrow Things^* \rightarrow (Result\ content\ type).$$

To demonstrate the meta-path model, we will model an IoTSE instance that queries for “homepages of IoT-enabled light bulbs which are reporting an abnormal energy consumption” as an example. This query can be decomposed into two subqueries: “finding the virtual representative of things, which are lightbulbs” and “finding the virtual representative of things, which are reporting an abnormal energy consumption.” The first subquery involves assessing each discovered representative (Representative) to determine whether it belongs to a lightbulb (Thing) and returning the representative of that lightbulb (Representative) as the search result. This subquery can be modeled with the meta-path $R \rightarrow T \rightarrow R$.

The second subquery involves as-

Figure 4. (Upper) Meta-graph from a concrete IoT network, and meta-path $R+D \rightarrow T \rightarrow D$.

content types to assess a query and to derive search results. Moreover, the types of IoT content appearing in a query influence the internal operations of an IoTSE instance.²⁰ As a result, an IoTSE model must capture succinctly both the types of involving IoT content and the relationships among those types. Terms such as “object search” are inadequate. To address this issue, we propose a model called *meta-path*.

Before defining meta-path, it would be helpful to introduce the idea of modeling the Internet of Things as a heterogeneous graph, which was inspired by PathSim.¹⁷ The nodes in this graph consist of IoT contents and IoT things that own these contents. The edges

that link things and content denote a possessive relationship between them. The edges that link things denote their possible correlations, such as sharing owners or operation environments.^{23,24}

From a concrete graph, we can derive a meta-graph that presents relationships between types of nodes. Each node in a meta-graph is either a type of IoT content or a thing. An edge between a content type and a thing represents the content type is offered by the thing. An edge between two things represents a correlation between them. Different types of thing-thing edges represent different forms of correlation between things. These thing-thing relationships can enable interesting queries such as

sessing each discovered sensing data stream (Dynamic IoT content) to detect an abnormality, finding the Thing that offers such data stream, and returning the representative of that thing (Representative) as the search result. This subquery can be modeled with the meta-path $D \rightarrow T \rightarrow R$.

By aggregating the two subqueries, we can model the IoTSE instance with the aggregated meta-path $R+D \rightarrow T \rightarrow R$. The IoTSE class addressing this meta-path is the second most common class in the IoTSE literature.

A Meta-path-based Classification System for IoTSE

IoTSE instances can be classified in various dimensions, from implementation technologies²⁷ to query processing behavior^{6,15} and the maturity of their development.³ Alternatively, we can classify IoTSE instances by their meta-paths, which provide succinct and comprehensive description of their functionality via the type of queries that they support. As mentioned previously, the form of a query that an IoTSE instance addresses influences its internal operations and, therefore, determines its solution space. A meta-path-based classification system will provide insights on what an IoTSE instance is and what it should do, according to the IoTSE literature.

We modeled the IoTSE prototypes selected earlier with the meta-path model and identified eight types of meta-path (Figure 5). Each meta-path represents a class of IoTSE.

$R \rightarrow R$ Class IoTSE

This IoTSE class is the most popular in the literature. Instances of this class resolve queries on ID, metadata, or content of representative IoT content, and return matching representatives as search results. The popularity of this class is a surprising finding, as it does not utilize distinctive content types of IoT, such as sensing data and actuating services, nor the relationship between IoT contents and things.

ForwardDS-IoT⁵ is an IoTSE prototype processing queries on semantic description of things, stored in a federation of repositories. Queries in ForwardDS-IoT specify conditions on metadata of IoT things and are translated into SPARQL queries. This prototype supports both synchronous and asynchronous query process-

ing. DiscoWoT⁹ is another prevalent prototype belonging to the $R \rightarrow R$ class. This system accepts identities of IoT contents as “queries” and returns representation of the given contents in a common format as “search results”. Its operation is based on crowd-sourced strategies for translating different types of resource description into the common format. Coverage of strategies represents “discovered IoT content” of DiscoWoT.

$D+R \rightarrow T \rightarrow R$ Class IoTSE

This class of IoTSE resolves queries on both sensing data streams (that is, dynamic IoT content) and representatives of IoT things, and returns the representatives of things that satisfy both query criteria. Efficient processing of dynamic IoT content is a primary challenge of this IoTSE class. The following two works represent two prevalent approaches to address this challenge.

IoT-SVK² searches for IoT things based on their textual description and real-time sensing values, with respect to spatial and temporal constraints. This search engine collects sensor readings continuously in parallel to the query assessment. To handle the constant influx of sensing data, IoT-SVK utilizes two B+ trees and R tree indexes, which are distributed across a hierarchy of indexing servers. Dyser search engine¹² also searches for IoT things based on their description and real-world states, which are derived from their real-time sensing values. Different from IoT-SVK, Dyser does not collect IoT content. Instead, it contacts things to validate their states for every query. To improve the efficiency of this operation, it predicts sensing

values by assuming the existence of repeating periods in sensing data and ranks things according to this prediction to minimize the number of things to validate.

$D \rightarrow D$ Class IoTSE

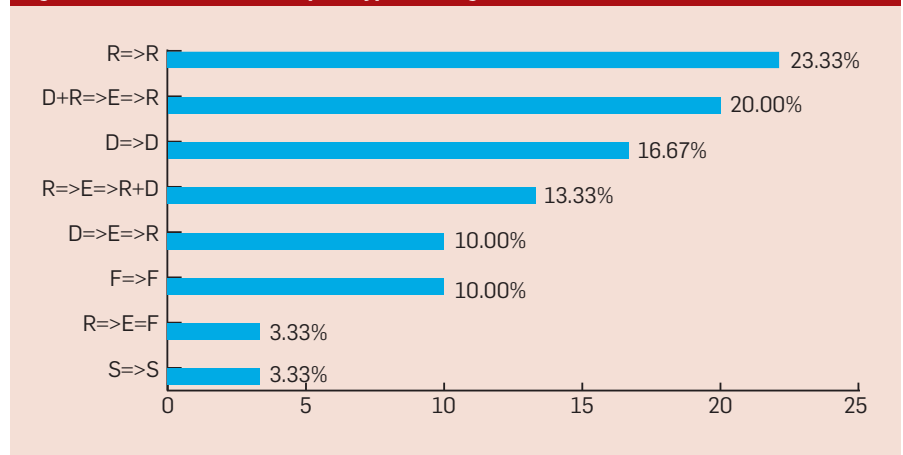
Search engine instances of this class process queries on metadata and content of sensing data streams and return relevant streams as search results. Key distinction of search engines in this class from the previous one is their focus on low-level sensor readings, instead of high-level states derived from readings.

CASSARAM¹³ queries sensing data streams on their contextual information, such as availability, accuracy, reliability and response time. It is motivated by the lack of the search functionality for an increasing number of sensors with overlapping capabilities deployed around the world. CASSARAM utilizes an extension of the Semantic Sensor Network Ontology (SSNO)¹ to describe the contextual information. A user would query this ontology with a SPARQL query generated by the graphical user interface of CASSARAM. This interface also captures the references of the search user. The Euclidian distance between matched sensors and the user reference in a multidimensional space built from different types of sensor contextual information is used for ranking purpose. Top ranked sensing streams are returned as search results.

$R \rightarrow T \rightarrow R+D$ Class IoTSE

This class of IoTSE can be considered as an extension of the class ($R \rightarrow R$). Instances of this class resolve queries

Figure 5. Distribution of meta-path types among reviewed IoTSE works.



on ID, metadata, or content of representative IoT content, and return both representatives and sensing streams of matching things as search results.

Snoogle²² is a representative prototype of this IoTSE class. It resolves queries on the textual data stored in IoT things to identify and locate the relevant things. Essentially, Snoogle is a text retrieval system operating on distributed, low-powered repositories. It utilizes a distributed top- k query algorithm with pruning, based on the characteristic of flash memory and Bloom filter, to increase the efficiency of the operation. The representative of matching things, along with their location at the query time (that is, dynamic information IoT content), is returned as search results.

$D \rightarrow T \rightarrow R$ Class IoTSE

This class of IoTSE resolves queries on various aspects of sensing data streams and returns the digital representative of things possessing the matching sensing data streams. Different from class $D+R \rightarrow T \rightarrow R$, search engines in this class do not consider other features of things.

Content-based Sensor Search (CSS)²¹ is a representative prototype of this class. It searches for IoT-enabled sensors that produce measurements within a certain range for a certain time prior. CSS contacts sensors to validate their values during query processing instead of collecting IoT content a priori. It utilizes time-independent prediction models (TIPM) to rank sensors based on their probability of having the queried state. These models assume that a sensor reading, which a sensor frequently and continuously reports in the past, has a higher probability to be its current reading. The details of the sensor nodes providing the matching streams are returned as search results.

$F \rightarrow F$ Class IoTSE

This class of IoTSE resolves queries on functionality of IoT things. Considering the popularity of functionality content in real world usage scenarios of IoT (for example, smart home), the limited support for this IoTSE class is a surprising finding. The lack of public datasets and standards for functionality content might have contributed to this limitation.

Mrisa et al.¹⁰ present a search and discovery mechanism for functionalities of physical entities. It aims to discover and expose high-level functionalities of a physical entity that can be realized by a combination of its low-level physical capabilities and functionalities exposed by other entities in the immediate area. These functionalities and capabilities are described in a shared ontology. Each physical entity queries this ontology with a set of SPARQL queries encapsulated in Java functions. This work is part of the avatar architecture from the ASAWoO project.^b

$R \rightarrow T \rightarrow F$ Class IoTSE

This class of IoTSE resolves queries on the representatives to find relevant things and returns functionality of those things as search results. Kamilaris et al.⁷ propose an IoTSE instance that utilizes a DNS-like mechanism to search for IoT things and return their functionalities. These functionalities are presented as RESTful Web services and capable of self-describing with specifications written in the Web Application Description Language (WADL).

$S \rightarrow S$ Class IoTSE

This class of IoTSE resolves queries on static information IoT content. Matching content is returned as search results. Microsearch¹⁸ is an instance of this IoTSE class. It is essentially a downscaled information retrieval system operating on sensor nodes with very limited computing and storage resources. It indexes small textual documents stored in the sensor node and returns the top- k documents that are most relevant to the query terms given by a search user.

Open Issues

As IoTSE research and engineering is a complex and relatively new area, researchers and practitioners face several types of technical challenges. Here, we discuss four open issues, derived from the existing literature, that affect most classes of IoTSE.

Building datasets for IoTSE research. Large-scale, open datasets that contain IoT content, sample queries, and ground truth, are critical to IoT

TSE research. They negate the need for difficult-to-replicate experiments and simplify the experimentation and evaluation of the research works on IoTSE. Research works on IoTSE have utilized some sensing datasets, such as Intel Lab,^c NOAA,^d bicycle rental,^e taxi GPS.^f Actuating functionality datasets, on the other hand, have not been found in the existing literature. Availability of the sample queries has also been limited, as they tend to be private property of industrial IoTSE instances.¹⁶ Providing access to IoT datasets is a challenge due to their massive size, reaching 21 terabytes a day,¹⁶ and their potential threats to privacy. Given these opportunities and challenges, building open IoT datasets is essential in the IoTSE research.

Ranking IoT contents by their natural order. Natural order ranking denotes the ordering of content by their intrinsic characteristics instead of their relevance to a given query. In large data collections where a massive number of data items can be relevant to a query, a search engine must rely on natural order ranking mechanisms to order and deliver the most relevant search results to its query clients. For example, the ranking of Web pages based on their importance by using link analysis algorithms such as PageRank is a form of natural order ranking.

As the anticipated size of the IoT is even more extensive than the Web, we anticipate that the natural order ranking mechanisms for the IoT content will be an exciting and challenging research topic that will play a crucial role in IoTSE. The first problem in this topic would be defining a natural order that is applicable across different IoT content. For the Web, the level of authority is the natural order of Web pages. For the IoT, what would be the natural order of the heterogeneous IoT content? When this natural order has been defined, the next problem would be developing mechanisms to calculate it on the IoT-scale.

A potential solution to natural order ranking could rely on the quality-

c <http://db.csail.mit.edu/labdata/labdata.html>

d <https://data.noaa.gov/datasetsearch/>

e <https://www.bicing.cat/>

f <https://github.com/roryhr/taxi-trajectories>

b <https://liris.cnrs.fr/asawoo/doku.php>

of-service metrics of IoT services. Another potential approach could reuse the solution of the Web by constructing a network of hidden links between IoT things^{23,24} and applying link analysis algorithms such as Page Rank and its variants to devise a natural ordering of content in the IoT.

Security, privacy, trust. IoTSE instances have the potential to detect and retrieve anything in the IoT, at any place and any time. They bring a wide range of benefits to human users and software agents but also present significant security and privacy risks. IoTSE instances can track a person, monitor an area without consent,⁴ and spy into warehouses of competing businesses.³ Perpetrators can also take advantage of IoTSE to propagate malicious sensing information and actuating services. As future IoT applications might rely solely on IoTSE to acquire IoT content for their operation, misleading information propagated by IoTSE can have severe impacts. For example, by planting sensors that imply a restaurant is full, competitors can drive it out of business. Addressing security, privacy, and trust issues, therefore, is arguably more critical to the success and adoption of IoTSE compared to perfecting its discovery and search algorithms.

Facilitating composition and reuse of IoTSE solution. Across different classes of IoTSE, we have observed shared internal operations such as content discovery, indexing, and searching, albeit with different implementation to serve different types of IoT content. We have also observed the overlaps between various meta-paths, such as between $[D + R \rightarrow T \rightarrow R]$ and $[D \rightarrow T \rightarrow R]$. These observations suggest that prior IoTSE instances can be reused to improve other instances or compose new instances, which might utilize a different meta-path.

Realizing composition and reuse of IoTSE solutions require a common IoTSE architecture and a supporting software infrastructure to support the development, accumulation of IoTSE components, and the engineering of IoTSE instances from those components. Tran et al.¹⁹ propose to utilize a shared software library to facilitate the development of reusable, composable IoTSE components and support

the composition of these components into operational IoTSE instances. Reducing the constraints of the shared library on component developers and simplifying the distribution of components in an IoTSE instance could improve the approach. The Service-oriented Architecture (SOA) is a potential solution to this problem, due to its enforced separation of concern between services and its native support for composition.

Conclusion

Internet of Things Search Engine denotes a software system responsible for discovering and resolving queries on contents of the Internet of Things. Due to the diversity of IoT contents, developing IoTSE is a complex and diverse problem that is still relatively immature. This article introduces concepts, models, and a classification system for IoTSE, which have been generated from a structured and comprehensive study of the literature on IoTSE. We have categorized the latest works into eight classes of IoTSE and presented four major open issues that impact all classes of IoTSE. C

References

- Compton, M. et al. The SSN ontology of the W3C semantic sensor network incubator group. *Web Semantics: Science, Services and Agents on the World Wide Web* 17 (2012), 25–2.
- Ding, Z., Chen, Z. and Yang, Q. IoT-SVKSearch: A real-time multimodal search engine mechanism for the Internet of Things. *Intern. J. Communication Systems* 27, 6 (2014), 871–897.
- Evdokimov, S. et al. Comparison of discovery service architectures for the Internet of Things. In *Proceedings of the IEEE 2010 Intern. Conf. on Sensor Networks, Ubiquitous, and Trustworthy Computing*.
- Frank, C. et al. The sensor internet at work: Locating everyday items using mobile phones. *Pervasive and Mobile Computing* 4, 3 (2008), 421–447.
- Gomes, P. et al. A federated discovery service for the Internet of Things. In *Proceedings of the 2nd ACM Workshop on Middleware for Context-Aware Applications in the IoT*. Vancouver, Canada, 2015.
- Jin, X. et al. Where searching will go in Internet of Things? In *Proceedings of Wireless Days*. IFIP, 2011. IEEE.
- Kamilaris, A., Papakonstantinou, K. and Pitsillides, A. Exploring the use of DNS as a search engine for the Web of Things. *IEEE World Forum on Internet of Things (WF-IoT)*. 2014. IEEE.
- Kitchenham, B.A. and Charters, S.M. Guidelines for performing systematic literature reviews in software engineering. EBSE Technical Report, Ver. 2.3, 2007.
- Mayer, S. and Guinard, D. An extensible discovery service for smart things. In *Proceedings of the 2nd ACM Intern. Workshop on Web of Things*. San Francisco, CA, 2011.
- Mrissa, M., Médini, L. and Jamont, J.-P. Semantic discovery and invocation of functionalities for the Web of Things. In *Proceedings of the 23rd IEEE Intern. Conf. on Enabling Technologies: Infrastructure for Collaborative Enterprises*. Parma, Italy, 2014.
- Ngu, A.H.H. et al. IoT middleware: A survey on issues and enabling technologies. *IEEE Internet of Things Journal* 4, 1 (2017), 1–20.
- Ostermaier, B. et al. A real-time search engine for the Web of Things. In *Proceedings of the 1st IEEE Intern. Conf. on the Internet of Things*. Tokyo, Japan, 2010.
- Perera, C. et al. Context-aware sensor search, selection and ranking model for Internet of Things middleware. In *Proceedings of the 14th IEEE Intern. Conf. on Mobile Data Management*. Milan, Italy, 2013.
- Qin, Y. et al. When things matter: A survey on data-centric Internet of Things. *J. Network and Computer Applications* 64 (2016), 137–153.
- Romer, K. et al. Real-time search for real-world entities: A survey. In *Proceedings of the IEEE* 98, 11 (2010), 1887–1902.
- Shemshadi, A., Sheng, Q.Z. and Qin, Y. ThingSeek: A crawler and search engine for the Internet of Things. In *Proceedings of the 39th ACM SIGIR Intern. Conf. on Research and Development in Information Retrieval*. Pisa, Italy, 2016.
- Sun, Y. et al. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. In *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.
- Tan, C.C. et al. Microsearch: When search engines meet small devices. In *Proceedings of the 6th Intern. Conf. on Pervasive Computing*. Sydney, Australia, 2008. Springer-Verlag Berlin.
- Tran, N.K. et al. A kernel-based approach to developing adaptable and reusable sensor retrieval systems for the Web of Things. In *Proceedings of the 18th Intern. Conf. on Web Information Systems Engineering*. Moscow, Russia, 2017.
- Tran, N.K. et al. Searching the Web of Things: State of the art, challenges and solutions. *ACM Computing Surveys* 50, 4 (2017), 1–34.
- Truong, C. and K. Römer. Content-based sensor search for the Web of Things. In *Proceedings of the IEEE Global Communications Conference*. Atlanta, GA, 2013.
- Wang, H., C.C. Tan and Q. Li. Snoogle: A search engine for pervasive environments. *IEEE Trans. Parallel and Distributed Systems* 21, 8 (2010), 1188–1202.
- Yao, L. et al. Things of interest recommendation by leveraging heterogeneous relations in the Internet of Things. *ACM Trans. Internet Technology* 16, 2 (2016), 9.
- Yao, L. et al. Unveiling underlying connections via mining human-thing interactions in the Web of Things. *ACM Trans. Intelligent Systems and Technology* 8, 5 (2017), 62.
- Yao, L., Sheng, Q.Z. and Dustdar, S. Web-based management of the Internet of Things. *IEEE Internet Computing* 19, 4 (2015), 60–67.
- Yap, K.-K., Srinivasan, V. and Motani, M. MAX: Human-centric search of the physical world. In *Proceedings of the 3rd Intern. Conf. on Embedded Networked Sensor Systems*. San Diego, CA, 2005.
- Zhou, Y. et al. Search techniques for the Web of Things: A taxonomy and survey. *Sensors* 16, 5 (2016), 600.

Nguyen Khoi Tran is a researcher at the University of Adelaide, South Australia.

Quan Z. Sheng is a professor at Macquarie University, Sydney, Australia.

M. Ali Babar is a professor at the University of Adelaide, South Australia.

Lina Yao is a senior lecturer at the University of New South Wales, Sydney, Australia.

Wei Emma Zhang is a lecturer at the University of Adelaide, South Australia.

Schahram Dustdar is a professor at TU Wien, Austria.

© 2019 ACM 0001-0782/19/7 \$15.00.



Watch the authors discuss this work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/iot-search-engine>

Markov logic can be used as a general framework for joining logical and statistical AI.

BY PEDRO DOMINGOS AND DANIEL LOWD

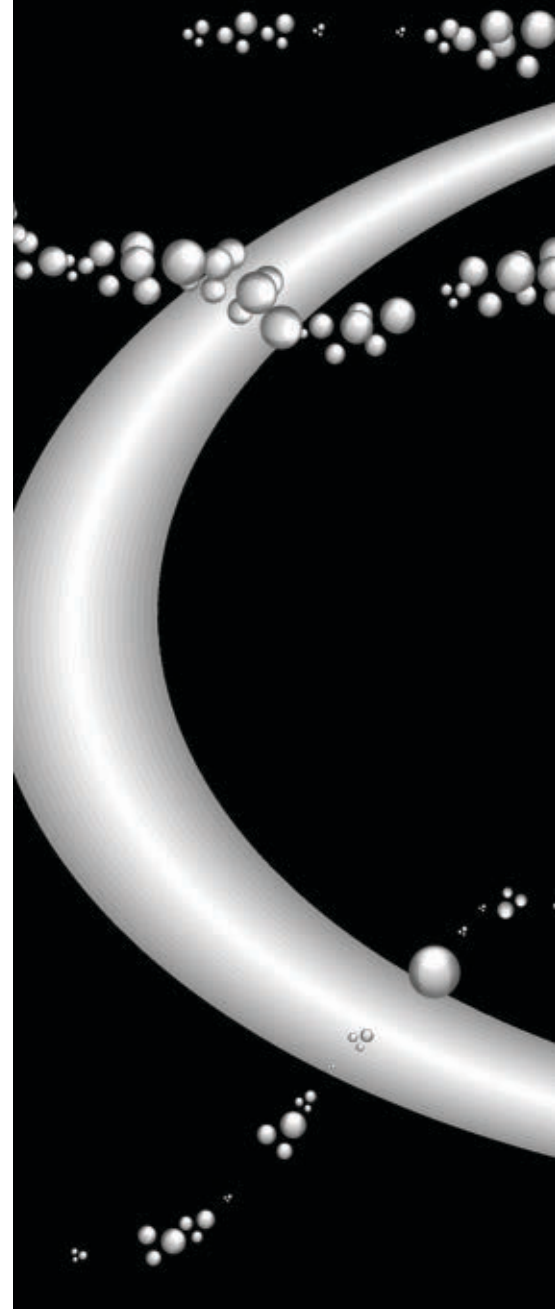
Unifying Logical and Statistical AI with Markov Logic

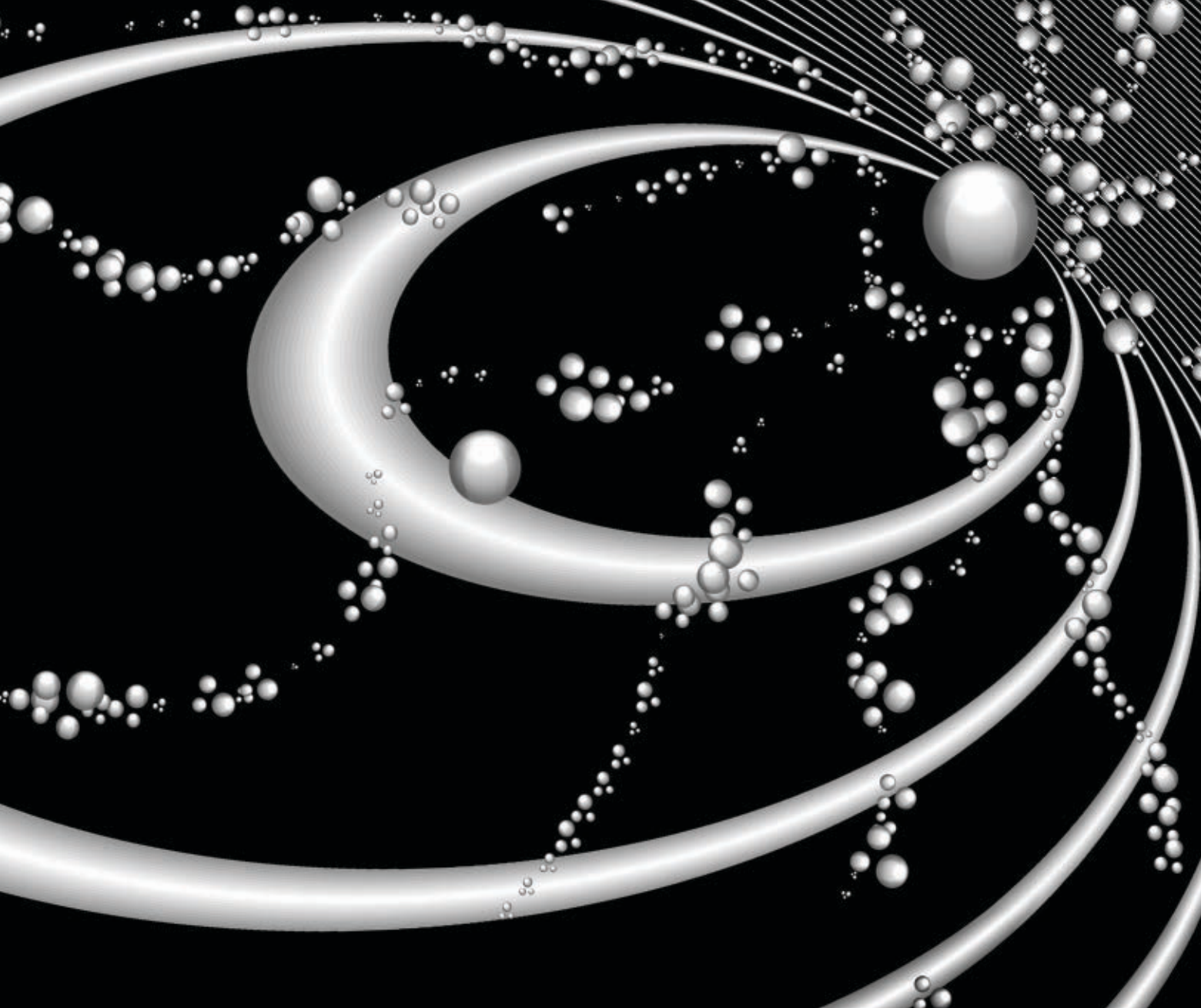
FOR MANY YEARS, the two dominant paradigms in artificial intelligence (AI) have been logical AI and statistical AI. Logical AI uses first-order logic and related representations to capture complex relationships and knowledge about the world. However, logic-based approaches are often too brittle to handle the uncertainty and noise present in many applications. Statistical AI uses probabilistic representations such as probabilistic graphical models to capture uncertainty. However, graphical models only represent distributions over propositional universes and must be customized to handle relational domains. As a result, expressing complex concepts and relationships in graphical models is often difficult and labor-intensive.

To handle the complexity and uncertainty present in most real-world problems, we need AI that is both logical and statistical, integrating first-order logic and graphical models. One or the other

» key insights

- Intelligent systems must be able to handle the complexity and uncertainty of the real world. Markov logic enables this by unifying first-order logic and probabilistic graphical models into a single representation. Many deep architectures are instances of Markov logic.
- A extensive suite of learning and inference algorithms for Markov logic has been developed, along with open source implementations like Alchemy.
- Markov logic has been applied to natural language understanding, information extraction and integration, robotics, social network analysis, computational biology, and many other areas.





by itself cannot provide the minimum functionality needed to support the full range of AI applications. Further, the two need to be fully integrated, and are not simply provided alongside each other. Most applications require simultaneously the expressiveness of first-order logic and the robustness of probability, not just one or the other. Unfortunately, the split between logical and statistical AI runs very deep. It dates to the earliest days of the field, and continues to be highly visible today. It takes a different form in each subfield of AI, but it is omnipresent. Table 1 shows examples of this. In each case, both the logical and the statistical approaches contribute something important. This justifies the abundant research on each of them, but also implies that ultimately a combination of the two is required.

Markov logic⁷ is a simple yet powerful generalization of first-order logic and probabilistic graphical models, which allows it to build on and integrate the best approaches from both logical and statistical AI. A *Markov logic network* (MLN) is a set of weighted first-order formulas, viewed as templates for constructing Markov networks. This yields a well-defined probability distribution in which worlds are more likely when they satisfy a higher-weight set of ground formulas. Intuitively, the magnitude of the weight corresponds to the relative strength of its formula; in the infinite-weight limit, Markov logic reduces to first-order logic. Weights can be set by hand or learned automatically from data. Algorithms for learning or revising formulas from data have also been developed. Inference algorithms for

Markov logic combine ideas from probabilistic and logical inference, such as Markov chain Monte Carlo, belief propagation, satisfiability, and resolution.

Markov logic has already been used to efficiently develop state-of-the-art models for many AI problems, such as collective classification, link prediction, ontology mapping, knowledge base refinement, and semantic parsing in application areas such as the Web, social networks, molecular biology, information extraction, and others. Markov logic makes solving new problems easier by offering a simple framework for representing well-defined probability distributions over uncertain, relational data. Many existing approaches can be described by a few weighted formulas, and multiple approaches can be combined by

Table 1. Examples of logical and statistical AI.

Field	Logical approach	Statistical approach
Knowledge representation	First-order logic	Graphical models
Automated reasoning	Satisfiability testing	Markov chain Monte Carlo
Machine learning	Inductive logic programming	Neural networks
Planning	Classical planning	Markov decision processes
Natural language processing	Definite clause grammars	Probabilistic context-free grammars

Table 2. Example of a first-order knowledge base and MLN.

English	First-order logic	Weight
"Friends of friends are friends."	$\forall x \forall y \forall z \text{ Fr}(x, y) \wedge \text{Fr}(y, z) \Rightarrow \text{Fr}(x, z)$	0.7
"Friendless people smoke."	$\forall x (\neg(\exists y \text{ Fr}(x, y)) \Rightarrow \text{Sm}(x))$	2.3
"Smoking causes cancer."	$\forall x \text{ Sm}(x) \Rightarrow \text{Ca}(x)$	1.5
"If two people are friends, then either both smoke or neither does."	$\forall x \forall y \text{ Fr}(x, y) \Rightarrow (\text{Sm}(x) \Leftrightarrow \text{Sm}(y))$	1.1

$\text{Fr}()$ is short for $\text{Friends}()$, $\text{Sm}()$ for $\text{Smokes}()$, and $\text{Ca}()$ for $\text{Cancer}()$.

including all of the relevant formulas. Many algorithms, as well as sample datasets and applications, are available in the open source *Alchemy* system¹⁷ (alchemy.cs.washington.edu).

In this article, we describe Markov logic and its algorithms, and show how they can be used as a general framework for combining logical and statistical AI. Before presenting background and details on Markov logic, we first discuss how it relates to other methods in AI.

Markov logic is the most widely used approach to unifying logical and statistical AI, but this is an active research area, and there are many others (see Kimmig et al.¹⁴ for a recent survey with many examples). Most approaches can be roughly categorized as either extending logic programming languages (for example, Prolog) to handle uncertainty, or extending probabilistic graphical models to handle relational structure. Many of these model classes can also be represented efficiently as MLNs (see Richardson and Domingos³² for a discussion of early approaches to statistical relational AI and how they relate to Markov logic). In recent years, most work on statistical relational AI has assumed a parametric factor (par-factor)²⁹ representation which is similar to the weighted formulas in an MLN. Probabilistic soft logic (PSL)¹

uses weighted formulas like Markov logic, but with a continuous relaxation of the variables in order to reason efficiently. In some cases, PSL can be viewed as Markov logic with a particular choice of approximate inference algorithm. One limitation of PSL's degree-of-satisfaction semantics is that more evidence does not always make an event more likely; many weak sources of evidence do not combine to produce strong evidence, even when the sources are independent.

Probabilistic programming²⁸ is another paradigm for defining and reasoning with rich, probabilistic models. Probabilistic programming is a good fit for problems where the data is generated by a random process, and the process can be described as the execution of a procedural program with random choices. Not every domain is well-suited to this approach; for example, we may wish to describe or predict the behavior of people in a social network without modeling the complete evolution of that network. Inference methods for probabilistic programming languages work backwards from the data to reason about the processes that generate the data and compute conditional probabilities of other events. Probabilistic graphical models perform similar reasoning, but typically have more structure to exploit for reasoning at scale.

Deep learning methods⁹ have led to competitive or dominant approaches in a growing number of problems. Deep learning fits complex, nonlinear functions directly from data. For domains where we know something about the problem structure, MLNs and other graphical models make it easier to capture background knowledge about the domain, sometimes specifying competitive models without having any training data at all. Due to their complementary strengths, combining deep learning with graphical models is an ongoing area of research.

First-Order Logic

A *first-order knowledge base* (KB) is a set of sentences or formulas in first-order logic. Formulas are constructed using four types of symbols: constants, variables, functions, and predicates. Constant symbols represent objects in the domain of interest (for example, people: Anna, Bob, and Chris). Variable symbols range over the objects in the domain. Function symbols (MotherOf) represent mappings from tuples of objects to objects. Predicate symbols represent relations among objects in the domain (Friends) or attributes of objects (Smokes). An *interpretation* specifies which objects, functions, and relations in the domain are represented by which symbols.

A *term* is any expression representing an object in the domain. It can be a constant, a variable, or a function applied to a tuple of terms. For example, Anna, x , and $\text{GreatestCommonDivisor}(x, y)$ are terms. An *atomic formula* or *atom* is a predicate symbol applied to a tuple of terms (for example, $\text{Friends}(x, \text{MotherOf}(\text{Anna}))$). Formulas are recursively constructed from atomic formulas using logical connectives and quantifiers. If F_1 and F_2 are formulas, the following are also formulas: $\neg F_1$ (negation), which is true if F_1 is false; $F_1 \wedge F_2$ (conjunction), which is true if both F_1 and F_2 are true; $F_1 \vee F_2$ (disjunction), which is true if F_1 or F_2 is true; $F_1 \Rightarrow F_2$ (implication), which is true if F_1 is false or F_2 is true; $F_1 \Leftrightarrow F_2$ (equivalence), which is true if F_1 and F_2 have the same truth value; $\forall x F_1$ (universal quantification), which is true if F_1 is true for every object x in the domain; and $\exists x F_1$ (existential quantification), which is true if F_1 is true for at least one object x in the domain.

Parentheses may be used to enforce precedence. A *positive literal* is an atomic formula; a *negative literal* is a negated atomic formula. The formulas in a KB are implicitly conjoined, and thus a KB can be viewed as a single large formula. A *ground term* is a term containing no variables. A *ground atom* is an atomic formula all of whose arguments are ground terms. A *grounding* of a predicate or formula is a replacement of all of its arguments by constants (or functions all of whose arguments are constants or other functions, recursively, but we consider only the case of constants in this article).

A *possible world* (along with an interpretation) assigns a truth value to each possible ground atom.

A formula is *satisfiable* if and only if there is at least one world in which it is true.

Determining if a formula is satisfiable is only semidecidable. Because of this, knowledge bases are often constructed using a restricted subset of first-order logic with more desirable properties.

Table 2 shows a simple KB. Notice that although these formulas may be *typically* true in the real world, they are not *always* true. In most domains, it is very difficult to come up with non-trivial formulas that are always true, and such formulas capture only a fraction of the relevant knowledge. Thus, despite its expressiveness, pure first-order logic has limited applicability to practical AI problems.

Many ad hoc extensions to address this have been proposed. In the more limited case of propositional logic, the problem is well solved by probabilistic graphical models such as Markov networks, as we describe next. We will later show how to generalize these models to the first-order case.

Markov Networks

A *Markov network* (also known as *Markov random field*) represents a joint probability distribution over variables $X = \{X_1, X_2, \dots, X_n\}$ as a product of *factors* (also known as *potential functions*):

$$P(X = x) = \frac{1}{Z} \prod_c \phi_c(x_c) = \frac{1}{Z} \Phi(x) \quad (1)$$

where each ϕ_c is a nonnegative, real-valued function defined over variables $X_c \subset X$ and Z is a normalization constant known as the partition function.

For convenience, we also define $\Phi(x)$ as the unnormalized probability distribution, the product of all potential functions. A Markov network can be represented as a graph with one node per variable and an undirected edge between any two variables that appear together in the same factor.

Markov networks are often conveniently represented as *log-linear models*, with each potential function replaced by an exponentiated weighted sum of features of the state, leading to

$$P(X = x) = \frac{1}{Z} \exp \left(\sum_j w_j f_j(x) \right) \quad (2)$$

A feature may be any real-valued function of the state. We will focus on binary features, $f_j(x) \in \{0, 1\}$, typically indicating if the variables are in some particular state or satisfy some logical expression.

Markov networks have been successfully applied to many problems in AI, such as stereo vision, natural language translation, information extraction, machine reading, social network analysis, and more. However, Markov networks only represent probability distributions over propositional domains with a fixed set of variables. There is no standard language for extending Markov networks to variable-sized domains, such as social networks over different numbers of people or documents with different numbers of words. As a result, applying Markov networks to these problems is a labor-intensive process requiring custom implementations.

Markov Logic

A first-order KB can be seen as a set of hard constraints on the set of possible worlds: if a world violates even one formula, it has zero probability. The basic

idea in Markov logic is to soften these constraints: when a world violates one formula in the KB, it is less probable, but not impossible. The fewer formulas a world violates, the more probable it is. Each formula has an associated weight (for example, see Table 2) that reflects how strong a constraint it is: the higher the weight, the greater the difference in log probability between a world that satisfies the formula and one that does not, other things being equal.

DEFINITION 1.³² A *Markov logic network (MLN)* L is a set of pairs (F_i, w_i) , where F_i is a formula in first-order logic and w_i is a real number. Together with a finite set of constants $C = \{c_1, c_2, \dots, c_{|C|}\}$, it defines a Markov network $M_{L,C}$ (Equations 1 and 2) as follows:

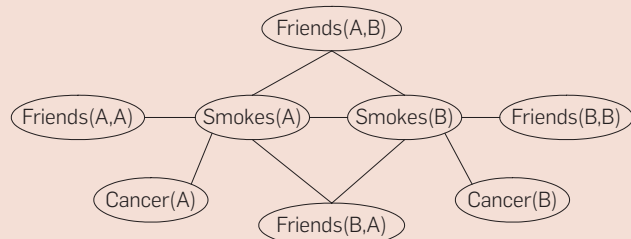
$M_{L,C}$ contains one random variable for each possible grounding of each atom appearing in L . The value of the variable is true if the ground atom is true and false otherwise.

$M_{L,C}$ contains one feature for each possible grounding of each formula F_i in L . The value of this feature is 1 if the ground formula is true and 0 otherwise. The weight of the feature is the w_i associated with F_i in L .

In the graph corresponding to $M_{L,C}$, there is a node for each grounding of each atom, and an edge appears between two nodes if the corresponding ground atoms appear together in at least one grounding of one formula in L .

For example, an MLN containing the formulas $\forall x \text{ Smokes}(x) \Rightarrow \text{Cancer}(x)$ (smoking causes cancer) and $\forall x \forall y \text{ Friends}(x, y) \Rightarrow (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))$ (friends have similar smoking habits) applied to the constants Anna and Bob (or A and B for short) yields the ground Markov network in Figure 1. Its features

Figure 1. Ground Markov network obtained by applying an MLN containing the formulas $\forall x \text{ Smokes}(x) \Rightarrow \text{Cancer}(x)$ and $\forall x \forall y \text{ Friends}(x, y) \Rightarrow (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))$ to the constants Anna(A) and Bob(B).



include $\text{Smokes}(\text{Anna}) \Rightarrow \text{Cancer}(\text{Anna})$. Notice that, although the two formulas are false as universally quantified logical statements, as weighted features of an MLN they capture valid statistical regularities, and in fact represent a standard social network model. Notice also that nodes and links in the social networks are both represented as nodes in the Markov network; arcs in the Markov network represent probabilistic dependencies between nodes and links in the social network (for example, Anna's smoking habits depend on her friends' smoking habits).

An MLN can be viewed as a *template* for constructing Markov networks. From Definition 1 and Equations 1 and 2, the probability distribution over possible worlds x specified by the ground Markov network $M_{L,C}$ is given by

$$P(X=x) = \frac{1}{Z} \exp\left(\sum_{i=1}^F w_i n_i(x)\right) \quad (3)$$

where F is the number of formulas in the MLN and $n_i(x)$ is the number of true groundings of F_i in x . As formula weights increase, an MLN increasingly resembles a purely logical KB, becoming equivalent to one in the limit of all infinite weights. When the weights are positive and finite, and all formulas are simultaneously satisfiable, the satisfying solutions are the modes of the distribution represented by the ground Markov network. Most importantly, Markov logic allows contradictions between formulas, which it resolves simply by weighing the evidence on both sides.

It is interesting to see a simple example of how Markov logic generalizes first-order logic. Consider an MLN containing the single formula $\forall x R(x) \Rightarrow S(x)$ with weight w , and $C = \{A\}$. This leads to four possible worlds: $\{\neg R(A), \neg S(A)\}$, $\{\neg R(A), S(A)\}$, $\{R(A), \neg S(A)\}$, and $\{R(A), S(A)\}$. From Equation 3 we obtain that $P(\{R(A), \neg S(A)\}) = 1/(3e^w + 1)$ and the probability of each of the other three worlds is $e^w/(3e^w + 1)$. (The denominator is the partition function Z ; see Markov Networks.) Thus, if $w > 0$, the effect of the MLN is to make the world that is inconsistent with $\forall x R(x) \Rightarrow S(x)$ less likely than the other three. From the probabilities here we obtain that $P(S(A) | R(A)) = 1/(1 + e^{-w})$. When $w \rightarrow \infty$,



A first-order knowledge base can be seen as a set of hard constraints on the set of possible worlds. The basic idea in Markov logic is to soften these constraints.



$P(S(A) | R(A)) \rightarrow 1$, recovering the logical entailment.

Bayesian networks, Markov networks, and many other propositional models frequently used in machine learning and data mining can be stated quite concisely as MLNs, and combined and extended simply by adding the corresponding formulas.³² Most significantly, Markov logic facilitates the modeling of multi-relational data, where objects are not independent but are related in diverse and complex ways. Such representations are important for social networks, biological networks, natural language understanding, and more. Boltzmann machines and the deep models based on them are special cases of Markov networks.

MLN weights can be normalized globally, as in undirected graphical models, or locally, as in directed ones. The latter option enables MLNs to handle variable numbers of objects and irrelevant objects similar to directed first-order formalisms (counter to Russell's³⁴ claim; see also Domingos⁶). In practice, even globally normalized MLNs are quite robust to these variations, largely because the number of relations per object usually varies much less than the number of objects, and because factors from irrelevant objects cancel out in conditional probabilities.

When working with Markov logic, we typically make three assumptions about the logical representation: different constants refer to different objects (unique names), the only objects in the domain are those representable using the constant and function symbols (domain closure), and the value of each function for each tuple of arguments is always a known constant (known functions). These assumptions ensure the number of possible worlds is finite and that the Markov logic network will give a well-defined probability distribution. These assumptions are quite reasonable in most practical applications, and greatly simplify the use of MLNs. We will make these assumptions in most of the remainder of this article, but Markov logic can be generalized to domains where they do not hold, such as those with infinite objects or continuous random variables.^{36,37} Open-world reasoning methods discussed by Russell³⁴ can also be applied to

Markov logic, as Bayesian networks can be translated into MLNs.

Inference

Given an MLN model, the questions of interest are answered by performing inference on it. (For example, “What are the topics of these Web pages, given the words on them and the links between them?”) Because an MLN acts as a template for a Markov network, we can always apply standard Markov network inference methods on the instantiated network. However, methods that also exploit the logical structure in an MLN can yield tremendous savings in memory and time. We first provide an overview of inference in Markov networks, and then describe how these methods can be adapted to take advantage of the MLN structure.

Markov network inference. The main inference problem in Markov networks is computing the probability of a set of query variables Q given some evidence E :

$$\begin{aligned} P(Q=q | E=e) &= \frac{P(q,e)}{P(e)} \\ &= \frac{\sum_{h'} \Phi(q,e,h')}{\sum_{q',h'} \Phi(q',e,h')} = \frac{Z_{q,e}}{Z_e} \end{aligned} \quad (4)$$

where $H = X - Q - E$ denotes the remaining nonquery, nonevidence variables, Φ is the unnormalized product of potentials from Equation 1, and $Z_{q,e}$ and Z_e are the partition functions of reduced Markov networks, where the query and evidence variables have been fixed to constants. Thus, if we can compute partition functions, then we can answer arbitrary probabilistic queries.

In general, computing Z or answering other queries in a Markov network is #P-complete. When the Markov network has certain structural properties, such as a tree or tree-like structure, inference can be done in polynomial time. For network structures with many variables and loops, exact inference is usually intractable and approximate inference algorithms are used instead. The two most common approaches to approximation are to generate random samples through some random process that converges to the true distribution, or to solve a relaxed problem that captures as many constraints from the original as possible. Examples of the former approach include Markov chain

Monte Carlo (MCMC) and importance sampling, and examples of the latter include loopy belief propagation and variational methods.

Any of these methods could be used to perform inference in an MLN after instantiating the ground network, and many of them have. However, inference remains challenging in Markov networks and even more challenging in MLNs, which are often very large and have many loops. Next we will discuss on one of the most promising inference methods to date, which can take advantage of logical structure, perform exact inference when tractable, and be relaxed to perform approximate inference when necessary.

Weighted model counting.

Computing the partition function in a Markov network can be reduced to a *weighted model counting* (WMC) problem. Weighted model counting finds the total weight of all satisfying assignments to a logical formula F . Following Chavira and Darwiche,² we focus on *literal-weighted* model counting, in which each literal is assigned a real-valued weight and the weight of an assignment is the product of the weights of its literals.

To represent Z as a WMC problem, we need each assignment x to receive weight $\Phi(x)$. Suppose each potential $\phi_i(x)$ evaluates to a constant Θ_i when a logical expression F_i is satisfied and 1 otherwise. (If the Markov network is not already in this form, we can convert it efficiently.) To define the WMC problem, for each potential ϕ_i , we introduce a literal A_i with weight Θ_i . We also introduce a logical formula, $A_i \leftrightarrow F_i$, so that A_i is only true when F_i is satisfied. Thus, the product of the weights of the A_i literals is exactly the product of the original potential functions.

WMC algorithms can then be used to solve the problem and compute Z . One approach is recursive decomposition, in which we break the WMC problem into two subproblems, one where some variable x_i is fixed to true and one where x_i is fixed to false. This requires exponential time in the worst case, but WMC algorithms can often exploit problem structure to solve it much faster in practice. Another approach is to compile the model into a logical form where WMC is tractable, such as d-DNNF, and build an arithmetic circuit based on it.² Once

compiled, the arithmetic circuit can be reused for multiple queries.

Probabilistic theorem proving. WMC is a natural approach to inference in MLNs, as MLNs already use a logical representation for their features. However, MLNs have additional structure to exploit: each formula is instantiated many times with different combinations of constants. For example, suppose we are modeling a social network in which each pair of people is either friends or not. Before introducing any information about the individuals, the probability that any two people are friends must be the same as any other pair. *Lifted inference* exploits these symmetries to reason efficiently, even on very large domains.²⁹

Probabilistic theorem proving (PTP)⁸ applies this idea to perform *lifted* weighted model counting, so that many equivalent groundings of the same formula can be counted at once without instantiating them. As in the propositional case, lifted WMC can also be performed by compiling the first-order knowledge base to a (lifted) arithmetic circuit for repeated querying.⁵

In some cases, lifted inference lets us reason efficiently independent of domain size, so that inferring probabilities over millions of constants and trillions of ground formulas takes no longer than reasoning over hundreds. More often, evidence about individual constants breaks symmetries, so that different groundings are no longer exchangeable. The efficiency gains from lifting depend on both the structure of the knowledge base and the structure of the evidence.

When there is not enough structure and symmetry to perform inference exactly, we can replace some of the recursive conditioning steps in PTP with sampling.⁸ This leads to an approximate lifted inference algorithm, where sampling is used to estimate the weighted count of some of the subformulas.

Learning

Here, we discuss methods for automatically learning weights, refining formulas, and constructing new formulas from data.

Weight learning. In generative learning, the goal is to learn a joint probability distribution over all atoms. A standard approach is to maximize the likelihood of the data

through gradient-based methods. Note that we can learn to generalize from even a single example because the formula weights are shared across their many respective groundings. This is essential when the training data is a single network, such as the Web. Given mild assumptions about the relational dependencies, maximizing the likelihood (or pseudo-likelihood) of a sufficiently large example will recover the parameters that generated the data.³⁸

For MLNs, the gradient of the log-likelihood is the difference between the true formula counts in the data and the expected counts according to the model. When learning a generative probability distribution over all atoms, even approximating these expectations tends to be prohibitively expensive or inaccurate due to the large state space. Instead, we can maximize pseudo-likelihood, which is the conditional probability of each atom in the database conditioned on all other atoms. Computing the pseudo-likelihood and its gradient does not require inference, and is therefore much faster. However, the pseudo-likelihood parameters may lead to poor results when long chains of inference are required. In order to combat overfitting, we penalize each weight with a Gaussian prior, but for simplicity, we ignore that in what follows.

In many applications, we know a priori which atoms will be evidence and which ones will be queried. For these cases, discriminative learning optimizes our ability to predict the query atoms Y given the evidence X . A common approach is to maximize the *conditional likelihood* of Y given X ,

$$P(y|x) = \frac{1}{Z_x} \exp\left(\sum_{i \in F_y} w_i n_i(x, y)\right) \quad (5)$$

where F_y is the set of all MLN formulas with at least one grounding involving a query atom, and $n_i(x, y)$ is the number of true groundings of the i th formula involving query atoms. The gradient of the conditional log-likelihood is

$$\begin{aligned} \frac{\partial}{\partial w_i} \log P_w(y|x) &= n_i(x, y) \\ &- \sum_{y'} P_w(y'|x) n_i(x, y') \\ &= n_i(x, y) - E_w[n_i(x, y)] \end{aligned} \quad (6)$$

where the sum is over all possible databases y' , and $P_w(y'|x)$ is $P(y'|x)$ computed using the current weight vector $w = (\dots, w_i, \dots)$. In other words, the i th component of the gradient is simply the difference between the number of true groundings of the i th formula in the data and its expectation according to the current model.

When computing the expected counts $E_w[n_i(x, y)]$ is intractable, we can approximate them using either the MAP state (that is, the most probable state of y given x) or by averaging over several samples from MCMC. We obtain the best results by applying a version of the scaled conjugate gradient algorithm. We use a small number of samples from MCMC to approximate the gradient and Hessian matrix, and use the inverse diagonal Hessian as a preconditioner (see Lowd and Domingos¹⁸ for more details and results).

MLN weights can also be learned with a max-margin approach, similar to structural support vector machines.¹¹

Structure learning. The structure of a MLN is the set of formulas to which we attach weights. Although these formulas are often specified by one or more experts, such knowledge is not always accurate or complete. In addition to learning weights for the provided formulas, we can revise or extend the MLN structure with new formulas learned from data. We can also learn the entire structure from scratch. The inductive logic programming (ILP) community has developed many methods for this purpose.⁴ ILP algorithms typically search for rules that have high accuracy, or high coverage, among others. However, because an MLN represents a probability distribution, much better results are obtained by using an evaluation function based on pseudo-likelihood.¹⁵ Log-likelihood or conditional log-likelihood are potentially better evaluation functions, but are much more expensive to compute.

Most structure learning algorithms focus on clausal knowledge bases, in which each formula is a disjunction of literals (negated or nonnegated atoms). The classic approach is to begin with either an empty network or an existing KB and perform a combinatorial search for formulas that improve the pseudo-likelihood.¹⁵

Either way, we have found it useful to start by adding all atomic formulas (single atoms) to the MLN. The weights of these capture (roughly speaking) the marginal distributions of the atoms, allowing the longer formulas to focus on modeling atom dependencies. To extend this initial model, we either repeatedly find the best formula using beam search and add it to the MLN, or add all “good” formulas of length l before trying formulas of length $l + 1$. Candidate formulas are formed by adding each predicate (negated or otherwise) to each current formula, with all possible combinations of variables, subject to the constraint that at least one variable in the new predicate must appear in the current formula. Hand-coded formulas are also modified by removing predicates.

A wide variety of other methods for MLN structure learning have been developed, such as generative learning with lifted inference,¹⁰ discriminative structure learning,¹¹ gradient boosting,¹³ and generating formulas using a Markov network²¹ or random walks.¹⁶ For the special case where MLN formulas define a relational Bayesian network, consistent Bayesian network structure learning methods can be extended to consistent structure learning in the relational setting.³⁵

Applications

MLNs have been used in a wide variety of applications, often achieving state-of-the-art performance. Their greatest strength is their flexibility for defining rich models in varied domains.

Collective classification. One of the most common uses of MLNs is for predicting the labels of interrelated entities, as in the friends and smoking example. Applications include labeling Web pages and predicting protein function.³ MLNs can also model collective classification tasks on sequential data, such as segmenting text for information extraction.³⁰

Link prediction. A second common task is to predict unknown or future relationships based on known relationships and attributes. Examples include predicting protein interaction,³ predicting advising relationships in a computer science department,³² and predicting work relationships among directors and actors.²⁰

Knowledge base mapping, integration, and refinement. Reasoning about the world requires combining diverse sources of uncertain information, such as noisy KBs. MLNs can easily represent KBs and soft constraints on the knowledge they represent: facts and rules in the knowledge base can be represented directly as atoms and formulas in the MLN. Ontology alignment can then be formulated as a link prediction problem, predicting which concepts in one ontology map to which concepts in the other. MLN formulas enforce structural similarity, so that related concepts in one ontology map to similarly related concepts in the other.²³ Similar rules can be used for knowledge base refinement, automatically detecting and correcting errors in uncertain knowledge by enforcing consistency among classes and relations.¹²

Semantic network extraction (SNE). A semantic network is a type of KB consisting of a collection of concepts and relationships among them. The SNE³⁹ system uses Markov logic to define a probability distribution over semantic networks. The MLN entities are the relation and object symbols from extracted tuples and the cluster assignments that group them into concepts and relationships. The MLN rules state that the truth of an extracted relationship depends on the clusters of the objects and relation involved. SNE uses a specialized bottom-up clustering algorithm to find the semantic clusters for objects and relations. This lets SNE scale to discover thousands of clusters over millions of tuples in just a few hours.

Semantic parsing. The goal of semantic parsing is to map sentences to logical forms representing the same information. The resulting information can be used to build a medical KB from PubMed abstracts, infer the meaning of a news article, or answer questions from an encyclopedia entry. Unsupervised semantic parsing³¹ learns to map dependency trees from sentences to their logical forms without any explicitly annotated data. The USP system does this by recursively clustering expressions (lambda forms) with similar subexpressions. The MLN for this includes four rules: one to cluster expressions into “core forms,” and three to cluster their



The goal of semantic parsing is to map sentences to logical forms representing the same information.



arguments into “argument forms” of some type and number. A clustering is more probable if expressions in the same cluster tend to have the same number of subexpressions as each other and those subexpressions are in the same clusters. As with SNE, USP uses a clustering algorithm to learn and reason more efficiently.

Extensions

Beyond the capabilities described here, Markov logic has been extended in a variety of ways to satisfy additional properties or accommodate different domains.

For decision theoretic problems, we can extend MLNs to Markov logic decision networks (MLDNs) by attaching a utility to each formula as well as a weight.²² The utility of a world is the sum of the utilities of its satisfied formulas. The optimal decision is the setting of the action predicates that jointly maximizes expected utility.

Domains with continuous as well as discrete variables can be handled by hybrid Markov logic networks (HMLNs).³⁷ HMLNs allow numeric properties of objects as nodes, in addition to Boolean ones, and numeric terms as features, in addition to logical formulas. For example, to reason about distances, we can introduce the numeric property $\text{Distance}(x, y)$. To state that a car should be centered in a lane, we can add terms such as:

$$\begin{aligned} &\text{Car}(c) \wedge \text{LeftLine}(l) \wedge \text{RightLine}(r) \\ &\Rightarrow -(\text{Dist}(c, l) - \text{Dist}(c, r))^2 \end{aligned}$$

When c is a car, l is the left lane boundary, and r is the right lane boundary, this term penalizes differences between the distance to the left and right boundaries. Inference algorithms for HMLNs combine ideas from satisfiability testing, slice-sampling MCMC, and numerical optimization. Weight learning algorithms are straightforward extensions of ones for MLNs.

Markov logic can be extended to infinite domains using Gibbs measures, the infinite-dimensional extension of Markov networks.³⁶ An MLN in an infinite domain is *locally finite* if each ground atom appears in a finite number of ground formulas. Local finiteness guarantees the existence of a probability measure; when the

interactions are not too strong, the measure is unique as well. Nonunique MLNs may still be useful for modeling large systems with strong interactions, such as social networks with strong word-of-mouth effects. In such cases, we can analyze the different “phases” of a nonunique MLN and define a satisfying measure to reason about entailment.

Recursive Markov logic networks or recursive random fields (RRFs)¹⁹ extend MLNs to multiple layers by replacing the logical formulas with MLNs, which can themselves have nested MLNs as features, for as many levels or layers as necessary. RRFs can compactly represent distributions such as noisy DNF, rules with exceptions, and *m*-of-all quantifiers. RRFs also allow more flexibility in revising or learning first-order representations through weight learning. An RRF can be seen as a type of deep neural network, in which the node activation function is exponential and the network is trained to maximize the joint likelihood of its input. In other ways, an RRF resembles a deep Boltzmann machine, but with no hidden variables to sum out.

Tractable Markov logic (TML)²⁴ is a probabilistic description logic where inference time is linear for all marginal, conditional, and MAP queries. TML defines objects in terms of class and part hierarchies, and allows objects to have probabilistic attributes, probabilistic relations between their subparts, and probabilistic existence. Tractability is ensured by having a direct mapping between the structure of the KB and the computation of its partition function: each split of a class into subclasses corresponds to a sum, and each split of a part into subparts corresponds to a product.

Getting Started with Markov Logic

If you would like to try out Markov logic for yourself, there are several open source software packages for learning or reasoning with MLNs. In some cases, software for learning and reasoning with Markov networks or conditional random fields can also be used; however, the task of translating from an MLN to a ground Markov network is left to you, and standard algorithms do not exploit the structure and symmetries present in MLNs.

When choosing the MLN structure, domain knowledge about the relevant relationships is a good place to start.

The oldest MLN toolkit is *Alchemy*,¹⁷ currently in version 2. Compared to other toolkits, *Alchemy* offers the widest variety of algorithms, such as multiple methods for generative and discriminative weight learning, structure learning, and marginal and MAP inference. *Tuffy*²⁵ offers a subset of *Alchemy*’s features but obtains greater scalability by using a database to keep track of groundings. *Tuffy* is the basis of *DeepDive*,²⁶ a system for information extraction, integration, and prediction built on Markov logic. Other implementations of Markov logic include Markov thebeast³³ and *RockIt*.²⁷

When applying Markov logic to new problems, it is usually best to start with a simple model on a small amount of data. High-arity predicates and formulas may have a large number of groundings, resulting in large models, high memory use, and slow inference. It is important to determine which modeling choices are most important for making accurate predictions and how expensive they are. Lifted inference techniques or customized grounding or inference methods can help good models scale to larger data.

When choosing the MLN structure, domain knowledge about the relevant relationships is a good place to start. When such knowledge is available, it is usually better to use it than to learn the structure from scratch. As with other knowledge engineering problems, there are often several ways to represent the same knowledge, and some representations may work better than others. For example, a relationship between smoking and cancer could be represented as equivalence ($\text{Smokes}(A) \Leftrightarrow \text{Cancer}(A)$), implication ($\text{Smokes}(A) \Rightarrow \text{Cancer}(A)$ and $\text{Cancer}(A) \Rightarrow \text{Smokes}(A)$), or conjunction ($\text{Smokes}(A) \wedge \text{Cancer}(A)$ and $\neg \text{Smokes}(A) \wedge \neg \text{Cancer}(A)$).

Conclusion and Directions for Future Research

Markov logic offers a simple yet powerful representation for AI problems in many domains. As it generalizes first-order logic, Markov logic can easily model the full relational structure present in many problems, such as multiple relations and attributes of different types and arities, relational concepts such as transitivity, and background knowledge in first-order

logic. And as it generalizes probabilistic graphical models, Markov logic can efficiently represent uncertainty in the concepts, attributes, relationships, among others required by most AI applications.

For future research, one of the most important directions is improving the efficiency of inference. Lifted inference algorithms obtain exponential speed-ups by exploiting relational symmetries, but can fail when these symmetries are broken by evidence or more complex structures. Tractable Markov logic guarantees efficient inference but constrains model structure. More research is needed to make reasoning work well in a wider range of models. Because most learning methods rely on inference, this will lead to more reliable learning methods as well.

A second key direction is enriching the representation itself. Markov logic is built on first-order logic, which is not always the best way to compactly encode knowledge, even in logical domains. For example, concepts such as “every person has at least five friends” are difficult to express with standard first-order connectives and quantifiers. Markov logic has been extended to handle decision theory, continuous variables, and more. Some new applications may require new extensions.

We hope that Markov logic will be of use to AI researchers and practitioners who wish to have the full spectrum of logical and statistical inference and learning techniques at their disposal, without having to develop every piece themselves. We also hope that Markov logic will inspire development of even richer representations and more powerful algorithms to further integrate and unify diverse AI approaches and applications. More details on Markov logic and its applications can be found in Domingos and Lowd.⁷

Acknowledgments

This research was partly supported by ARO grants W911NF-08-1-0242 and W911NF-15-1-0265, DARPA contracts FA8750-05-2-0283, FA8750-07-D-0185, FA8750-16-C-0158, HR0011-06-C-0025, HR0011-07-C-0060, NBCH-D030010 and AFRL contract FA8750-13-2-0019, NSF grants IIS-0534881 and IIS-0803481, ONR grants N-00014-05-1-0313,

N00014-08-1-0670, N00014-12-1-0312 and N00014-16-1-2697, an NSF CAREER Award (first author), a Sloan Research Fellowship (first author), an NSF Graduate Fellowship (second author) and a Microsoft Research Graduate Fellowship (second author). The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, DARPA, AFRL, NSF, ONR, or the United States Government. **C**

References

- Bach, S., Broecheler, M., Huang, B., Getoor, L. Hinge-loss Markov random fields and probabilistic soft logic. *J. Mach. Learn. Res.* 18, 109 (2017), 1–67.
- Chavira, M., Darwiche, A. On probabilistic inference by weighted model counting. *Artif. Intell.* 6–7, 172 (2008), 772–799.
- Davis, J., Domingos, P. Deep transfer via second-order Markov logic. In *Proceedings of the 26th International Conference on Machine Learning*. ACM Press, Montréal, Canada.
- De Raedt, L. *Logical and Relational Learning*. Springer, Berlin, Germany, 2008.
- Van den Broeck, G., Taghipour, N., Meert, W., Davis, J., De Raedt, L. Lifted probabilistic inference by first-order knowledge compilation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)* (2011). Barcelona, Spain.
- Domingos, P., Kersting, K., Mooney, R., Shavlik, J. What about statistical relational learning? *Commun. ACM* 58, 12 (2015), 8.
- Domingos, P., Lowd, D. *Markov Logic: An Interface Layer for AI*. Morgan & Claypool, San Rafael, CA, 2009.
- Gogate, V., Domingos, P. Probabilistic theorem proving. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI-11)*. AUAI Press, Barcelona, Spain, 2011.
- Goodfellow, I., Bengio, Y., Courville, A. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- Van Haaren, J., Van den Broeck, G., Meert, W., Davis, J. Lifted generative learning of Markov logic networks. *Mach. Learn.* 103 (2015), 27–55.
- Huynh, T., Mooney, R. Discriminative structure and parameter learning for Markov logic networks. In *Proceedings of the 25th International Conference on Machine Learning* (2008). ACM Press, Helsinki, Finland, 416–423.
- Jiang, S., Lowd, D., Dou, D. Ontology matching with knowledge rules. In *Proceedings of the 26th International Conference on Database and Expert Systems Applications (DEXA 2015)* (2015). Springer, Valencia, Spain.
- Khot, T., Natarajan, S., Kersting, K., Shavlik, J. Gradient-based boosting for statistical relational learning: the Markov logic network and missing data cases. *Mach. Learn.* 100 (2015), 75–100.
- Kimmig, A., Mihalikova, L., Getoor, L. Lifted graphical models: a survey. *Mach. Learn.* (1), 99 (2015), 1–45.
- Kok, S., Domingos, P. Learning the structure of Markov logic networks. In *Proceedings of the 22nd International Conference on Machine Learning* (2005). ACM Press, Bonn, Germany, 441–448.
- Kok, S., Domingos, P. Learning Markov logic networks using structural motifs. In *Proceedings of the 27th International Conference on Machine Learning* (2010). ACM Press, Haifa, Israel.
- Kok, S., Sumner, M., Richardson, M., Singla, P., Poon, H., Lowd, D., Domingos, P. The Alchemy system for statistical relational AI. Technical Report. Department of Computer Science and Engineering, University of Washington, Seattle, WA, 2000. <http://alchemy.cs.washington.edu>.
- Lowd, D., Domingos, P. Efficient weight learning for Markov logic networks. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases* (2007). Springer, Warsaw, Poland, 200–211.
- Lowd, D., Domingos, P. *Recursive random fields*. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (2007). AAAI Press, Hyderabad, India, 950–955.
- Mihalikova, L., Huynh, T., Mooney, R.J. Mapping and revising Markov logic networks for transfer learning. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence* (2007). AAAI Press, Vancouver, Canada, 608–614.
- Mihalikova, L., Mooney, R. Bottom-up learning of Markov logic network structure. In *Proceedings of the 24th International Conference on Machine Learning* (2007). ACM Press, Corvallis, OR, 625–632.
- Nath, A., Domingos, P. A language for relational decision theory. In *Proceedings of the International Workshop on Statistical Relational Learning* (2009). Leuven, Belgium.
- Niepert, M., Meilicke, C., Stuckenschmidt, H. A probabilistic-logical framework for ontology matching. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence* (2010). AAAI Press.
- Niepert, M., Domingos, P. Learning and inference in tractable probabilistic knowledge bases. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence* (2015). AUAI Press, Brussels, Belgium.
- Niu, F., Ré, C., Doan, A., Shavlik, J. Tuffy: scaling up statistical inference in Markov logic networks using an RDBMS. *PVLDB* 4 (2011), 373–384.
- Niu, F., Zhang, C., Ré, C., Shavlik, J. DeepDive: web-scale knowledge-base construction using statistical learning and inference. In VLDS, 2012.
- Noessner, J., Niepert, M., Stuckenschmidt, H. RockIT: exploiting parallelism and symmetry for MAP inference in statistical relational models. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence* (2013). AAAI Press, Bellevue, WA.
- Pfeffer, A. *Practical Probabilistic Programming*. Manning Publications, 2016.
- Poole, D. First-order probabilistic inference. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (2003). Morgan Kaufmann, Acapulco, Mexico, 985–991.
- Poon, H., Domingos, P. Joint inference in information extraction. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence* (2007). AAAI Press, Vancouver, Canada, 913–918.
- Poon, H., Domingos, P. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing* (2009). ACL, Singapore.
- Richardson, M., Domingos, P. Markov logic networks. *Mach. Learn.* 62 (2006), 107–136.
- Riedel, S. Improving the accuracy and efficiency of MAP inference for Markov logic. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence* (2008). AUAI Press, Helsinki, Finland, 468–475.
- Russell, S. Unifying logic and probability. *Commun. ACM* 58, 7 (2015), 88–97. <https://doi.org/10.1145/2699411>
- Schulte, O., Gholami, S. Locally consistent Bayesian network scores for multi-relational data. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI-17)* (2017). Melbourne, Australia, 2693–2700.
- Singla, P., Domingos, P. Markov logic in infinite domains. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence* (2007). AUAI Press, Vancouver, Canada, 368–375.
- Wang, J., Domingos, P. Hybrid Markov logic networks. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence* (2008). AAAI Press, Chicago, IL, 1106–1111.
- Xiang, R., Neville, J. Relational learning with one network: An asymptotic analysis. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics* (2011), 779–788.
- Kok, S. and Domingos, P. Extracting semantic networks from text via relational clustering. In *Proceedings of ECML/PKDD-08* (Antwerp, Belgium, Sept. 2008). Springer, 624–639.

Pedro Domingos (pedrod@cs.washington.edu) is a professor in the Allen School of Computer Science and Engineering at the University of Washington, Seattle, WA, USA.

Daniel Lowd (lowd@cs.uoregon.edu) is an associate professor in the Department of Computer and Information Science at the University of Oregon, Eugene, OR, USA.

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

research highlights

P. 85

Technical Perspective Do You Know Why Your Web Pages Load Faster?

By Costin Raiciu

P. 86

Taking a Long Look at QUIC: An Approach for Rigorous Evaluation of Rapidly Evolving Transport Protocols

By Arash Molavi Kakhki, Samuel Jero, David Choffnes,
Cristina Nita-Rotaru, and Alan Mislove

Technical Perspective

Do You Know Why Your Web Pages Load Faster?

By Costin Raiciu

TO KEEP UP with demand and ensure users get quick access to information on the World Wide Web, Internet service providers have been adding capacity continuously, interconnecting more users and companies and at faster speeds. For home users, the progression has seen capacity increase from dial-up (56Kbps) to fiber (1Gbps), while for mobile users cellular speeds have increased from GPRS (~100Kbps) to LTE (~100Mbps).

As with Moore's Law for computing, and despite continuous investment in capacity, we have reached a point where adding more capacity will not necessarily make the Web faster. The fundamental reason is that *propagation latency*—the time it takes information to travel from one point to another on the Internet—is lower bounded by the time it takes light to travel the same distance, and thus cannot be lowered.

The time required to download a small Web page is dominated by propagation latency between the client and the server, and not throughput. If a client from Bucharest wishes to visit a Web page hosted in Silicon Valley, the download time will be lower bounded by round-trip time, which is the latency to cross the Atlantic twice and cannot be faster than 100ms. In practice, latency is quite a bit higher than this theoretical optimum.

To reduce this latency, content distribution networks (such as Akamai) appeared around 2000 that placed servers all around the globe to move content physically closer to users. In my example, the content hosted in Silicon Valley would be replicated on CDN servers in Romania such that the client can reach the content in tens instead of hundreds of milliseconds. CDNs are now ubiquitous, but they do not solve the latency problem completely: they work really well for static content, but less so for dynamically generated content.

More importantly, the protocols sending information over the Internet

have not been optimized for latency, and require many round-trip times between the client and the server to download a Web page. In fact, to download a small Web page over the prevalent transport protocol stack (HTTP2 running over TLS version 1.2 over TCP) requires at least four RTTs, severely inflating Web latency. Higher latencies lead to disgruntled users and less business, so there is a strong push to reduce Web latency.

To reduce the number of RTTs and thus Web latency, non-trivial changes to the base protocols (HTTP, TLS, and TCP) are required. While capacity enhancements or CDN deployments were implemented by a single entity (for example, ISPs), protocol changes require multiple stakeholders to agree as they first require standardization, then implementation by multiple operating systems and finally deployment on user devices. Following this approach, changes to TCP were introduced over the past six years to allow zero-RTT connection setup and TLS version 1.3 is significantly faster than 1.2. Unfortunately, such changes to existing protocols have limited impact because they must obey the layered architecture (HTTP/TLS/TCP), they need to support legacy applications, and require huge development resources and many years to get deployed.

QUIC is a novel protocol proposed by Google that reduces latency by replacing the entire HTTP/TLS/TCP stack with a single protocol that runs on top of UDP. The key benefit of running atop UDP is the protocol stack can be implemented as a user-space application, rather than in the kernel as needed when changing TCP, for instance. This implies that QUIC protocol changes can be pushed as easily as changing an application.

Google's QUIC approach is radical because it bypasses all the hurdles faced by incremental protocol changes: as Google controls both the servers and the client stack it can simply implement the protocol and deploy it both

on its servers and the clients (through Chrome), as often as it wishes, without external factors delaying the process. QUIC was first deployed in 2012 and has since been continuously updated. Today, QUIC is widely used and it carries a large fraction of Google's traffic; it is also undergoing standardization to enable other companies to use it, but standardization follows deployment, not the reverse.

QUIC's organic development has left people scratching their heads in both the research and standardization communities. QUIC's advocates point to impressive performance numbers in its favor, mostly reported by Google. Its detractors complain about the lack of justification for the chosen protocol mechanisms, and in general the lack of understanding of the reasons *why* QUIC outperforms TCP; the argument is that without such understanding, QUIC's gains could prove elusive when the network evolves in the future.

The following paper is a bold attempt to unearth the reasons why QUIC works better than TCP. The authors provide a unique and comprehensive insight into QUIC's behavior and how it compares to HTTP2/TLS/TCP. In contrast to many other studies of QUIC's performance, the work by Kakhki et al. does not only focus on the latest version of QUIC, but examines all versions comparatively, contrasting code changes to varying performance. Furthermore, the paper fights the lack of documentation by extracting the QUIC state machine from the code itself. The work is interesting because it sets the basis for a thorough understanding of why QUIC works so well and it should be equally interesting for computer science researchers outside networking. 

Costin Raiciu is an associate professor in the Computer Science Department at the University Politehnica of Bucharest, Romania.

Copyright held by author/owner.

Taking a Long Look at QUIC

An Approach for Rigorous Evaluation of Rapidly Evolving Transport Protocols

By Arash Molavi Kakhki,* Samuel Jero, David Choffnes, Cristina Nita-Rotaru, and Alan Mislove

Abstract

Google's Quick UDP Internet Connections (QUIC) protocol, which implements TCP-like properties at the application layer atop a UDP transport, is now used by the vast majority of Chrome clients accessing Google properties but has no formal state machine specification, limited analysis, and ad-hoc evaluations based on snapshots of the protocol implementation in a small number of environments. Further frustrating attempts to evaluate QUIC is the fact that the protocol is under rapid development, with extensive rewriting of the protocol occurring over the scale of months, making individual studies of the protocol obsolete before publication.

Given this unique scenario, there is a need for alternative techniques for understanding and evaluating QUIC when compared with previous transport-layer protocols. First, we develop an approach that allows us to conduct analysis across multiple versions of QUIC to understand how code changes impact protocol effectiveness. Next, we instrument the source code to infer QUIC's state machine from execution traces. With this model, we run QUIC in a large number of environments that include desktop and mobile, wired and wireless environments and use the state machine to understand differences in transport- and application-layer performance across multiple versions of QUIC and in different environments. QUIC generally outperforms TCP, but we also identified performance issues related to window sizes, re-ordered packets, and multiplexing large number of small objects; further, we identify that QUIC's performance diminishes on mobile devices and over cellular networks.

1. INTRODUCTION

Transport-layer congestion control is one of the most important elements for enabling both fair and high utilization of Internet links shared by multiple flows. As such, new transport-layer protocols typically undergo rigorous design, analysis, and evaluation—producing public and repeatable results demonstrating a candidate protocol's correctness and fairness to existing protocols—before deployment in the Operating System (S) kernel at scale.

Because this process takes time, years can pass between development of a new transport-layer protocol and its wide deployment in operating systems. In contrast, developing an *application-layer* transport (i.e., one not requiring OS

kernel support) can enable rapid evolution and innovation by requiring only changes to application code, with the potential cost due to performance issues arising from processing packets in userspace instead of in the kernel.

The Quick UDP Internet Connections (QUIC) protocol, initially released by Google in 2013,⁴ takes the latter approach by implementing reliable, high-performance, in-order packet delivery with congestion control at the application layer (and using UDP as the transport layer).^a Far from just an experiment in a lab, QUIC is supported by all Google services and the Google Chrome browser; as of 2016, more than 85% of Chrome requests to Google servers use QUIC.^{21 b} In fact, given the popularity of Google services (including search and video), QUIC now represents a substantial fraction (estimated at 7%¹⁵) of all Internet traffic. While initial performance results from Google show significant gains compared to TCP for the slowest 1% of connections and for video streaming,⁹ there have been very few repeatable studies measuring and explaining the performance of QUIC compared with standard HTTP/2+TCP.^{8, 11, 17} In addition to Google's QUIC, an IETF working group established in 2016 is working on standardizing QUIC and there are more than 20 QUIC implementations in progress.² Our study focuses on Google's QUIC implementation.

Our overarching goal is to understand the benefits and tradeoffs that QUIC provides. In this work, we address a number of challenges to properly evaluate QUIC and make the following key contributions.

First, we identify a number of pitfalls for application-layer protocol evaluation in emulated environments and across multiple QUIC versions. Through extensive calibration and validation, we identify a set of configuration parameters that fairly compare QUIC, as deployed by Google, with TCP-based alternatives.

Second, we develop a methodology that automatically generates network traffic to QUIC- and TCP-supporting servers in a way that enables head-to-head comparisons. Further, we instrument QUIC to identify the root causes

^a It also implements TLS and SPDY, as described in the next section.

^b Newer versions of QUIC running on servers are incompatible with older clients, and ISPs sometimes block QUIC as an unknown protocol. In such cases, Chrome falls back to TCP.

The original version of this paper was published in the *Proceedings of the 2017 ACM Internet Measure Conference* (London, U.K., Nov. 1–3, 2017).

* Work done while at Northeastern University.

behind observed performance differences and to generate inferred state machine diagrams. We make this code (and our dataset) publicly available at <http://quic.ccs.neu.edu>.

Third, we conduct tests using a variety of emulated network conditions, against our own servers and those run by Google, from both desktop and mobile-phone clients, and using multiple historical versions of QUIC. This analysis allows us to understand how QUIC performance evolved over time, and to determine how code changes impact relevant metrics. In doing so, we produce the first state machine diagrams for QUIC based on execution traces.

Summary of findings. Our key findings covered in this article are as follows:

- In the desktop environment, QUIC outperforms TCP+HTTPS in nearly every scenario. This is due to factors that include 0-RTT connection establishment and recovering from loss quickly.
- QUIC is sensitive to out-of-order packet delivery. QUIC interprets such behavior as loss and performs significantly worse than TCP in many scenarios.
- QUIC's performance gains are diminished on phones due to its reliance on application-layer packet processing and encryption.
- QUIC outperforms TCP in scenarios with fluctuating bandwidth. This is because QUIC's Acknowledgment (ACK) implementation eliminates ACK ambiguity, resulting in more precise RTT and bandwidth estimations.
- When competing with TCP flows, QUIC is unfair to TCP by consuming more than twice its fair share of the bottleneck bandwidth.
- QUIC performance has improved since 2016 mainly due to a change from a conservative maximum congestion window to a much larger one.
- We identified a bug affecting the QUIC server included in Chromium version 52 (the stable version at the time of our experiments), where the initial congestion window and Slow Start threshold led to poor performance compared with TCP.

2. BACKGROUND AND RELATED WORK

Google's QUIC protocol is an application-layer transport protocol that is designed to provide high performance, reliable in-order packet delivery, and encryption.⁴ The protocol was introduced in 2013, and has undergone rapid development by Google developers. QUIC is included as a separate module in the Chromium source; at the time of our experiments, the latest stable version of Chrome is 60, which supports QUIC versions up to 37.12 versions of QUIC have been released during our study, that is, between September 2015 and January 2017.^c

QUIC motivation. The design of QUIC is motivated largely by two factors. First, experimenting with and deploying new transport layers in the OS is difficult to do

quickly and at scale. On the other hand, changing application-layer code can be done relatively easily, particularly when client and server code are controlled by the same entity (e.g., in the case of Google). As such, QUIC is implemented at the application layer to allow Google to more quickly modify and deploy new transport-layer optimizations at scale.

Second, to avoid privacy violations as well as transparent proxying and content modification by middleboxes, QUIC is encrypted end-to-end, protecting not only the application-layer content (e.g., HTTP) but also the transport-layer headers.

QUIC features. QUIC implements several optimizations and features borrowed from existing and proposed TCP, TLS, and HTTP/2 designs. These include:

- *"0-RTT" connection establishment:* Clients that have previously communicated with a server can start a new session without a three-way handshake, using limited state stored at clients and servers.
- *Reduced "head of line blocking":* HTTP/2 allows multiple objects to be fetched over the same connection, using multiple streams within a single flow. If a loss occurs in one stream when using TCP, all streams stall while waiting for packet recovery. In contrast, QUIC allows other streams to continue to exchange packets even if one stream is blocked due to a missing packet.
- *Improved congestion control:* QUIC implements better estimation of connection Round-trip Time (RTTs) and detects and recovers from loss more efficiently.

Other features include forward error correction^d and improved privacy and flow integrity compared to TCP.

Most relevant to this work are the congestion and flow control enhancements over TCP, which have received substantial attention from the QUIC development team. QUIC currently^e uses the Linux TCP Cubic congestion control implementation,²⁰ and adds with several new features. Specifically, QUIC's ACK implementation eliminates ACK ambiguity, which occurs when TCP cannot distinguish losses from out-of-order delivery. It also provides more precise timing information that improves bandwidth and RTT estimates used in the congestion control algorithm. QUIC includes packet pacing to space packet transmissions in a way that reduces bursty packet losses, tail loss probes¹² to reduce the impact of losses at the end of flows, and proportional rate reduction¹⁶ to mitigate the impact of random loss on performance.

2.1. Related work

QUIC emulation results. Several papers explore QUIC performance and compare it with TCP.^{8, 11, 17} However, prior work have a number of shortcomings including lack of proper configuration of QUIC, limited test environments,

^c Throughout this paper, unless stated otherwise, we use QUIC version 34, which we found to exhibit identical performance to versions 35 and 36. Changelogs and source code analysis confirm that none of the changes should impact protocol performance.

^d This feature allows QUIC to recover lost packets without needing retransmissions. Due to poor performance it is currently disabled.²²

^e Google is developing a new congestion control called BBR,¹⁰ which is not yet in general deployment.

and absence of root cause analysis for reported observations. We refer the reader to our full paper¹⁴ for detailed discussion on these works.

Google-reported QUIC performance. The only large-scale performance results for QUIC in production come from Google. This is mainly due to the fact that at the time of writing, Google is the only organization known to have deployed the protocol in production. Google claims that QUIC yields a 3% improvement in mean page load time (PLT) on Google Search when compared to TCP, and that the slowest 1% of connections load one second faster when using QUIC.⁹ In addition, in a recent paper¹⁵ Google reported that on average, QUIC reduces Google search latency by 8% and 3.5% for desktop and mobile users respectively and reduces video rebuffer time by 18% for desktop and 15.3% for mobile users. Google attributes these performance gains to QUIC's lower-latency connection establishment (described below), reduced head-of-line blocking, improved congestion control, and better loss recovery.

In contrast to our work, Google-reported results are aggregated statistics that do not lend themselves to repeatable tests or root cause analysis. *This work takes a complementary approach, using extensive controlled experiments in emulated and operational networks to evaluate Google's performance claims (Section 4) and root cause analysis to explain observed performance.*

3. METHODOLOGY

We now describe our methodology for evaluating QUIC and comparing it to the combination of HTTP/2, TLS, and TCP. The tools we developed for this work and the data we collected are publicly available.

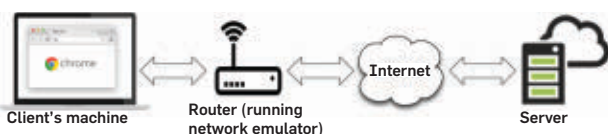
3.1. Testbed

We conduct our evaluation on a testbed that consists of a device machine running Google's Chrome browser^f connected to the Internet through a router under our control (Figure 1). The router runs OpenWRT (Barrier Breaker 14.07, Linux OpenWrt 3.10.49) and includes Linux's Traffic Control and Network Emulation tools, which we use to emulate network conditions including available bandwidth, loss, delay, jitter, and packet reordering.

Our clients consist of a desktop (Ubuntu 14.04, 8GB memory, Intel Core i5 3.3GHz) and two mobile devices: a Nexus 6 (Android 6.0.1, 3GB memory, 2.7GHz quad-core) and a MotoG (Android 4.4.4, 1GB memory, 1.2GHz quadcore).

^f The only browser supporting QUIC at the time of this writing.

Figure 1. Testbed setup. The server is an EC2 virtual machine running both QUIC and Apache server. The empirical RTT from client to server is 12ms and loss is negligible.



Our servers run on Amazon EC2 (Kernel 4.4.0-34-generic, Ubuntu 14.04, 16GB memory, 2.4GHz quad-core) and support HTTP/2 over TCP (using Cubic and the default Linux TCP stack configuration) via Apache 2.4 and over QUIC using the standalone QUIC server provided as part of the Chromium source code. To ensure comparable results between protocols, we run our Apache and QUIC servers on the same virtual machine and use the same machine/device as the client. We increase the UDP buffer sizes if necessary to ensure there are no networking bottlenecks caused by the OS. As we discuss in Section 4.1, we configure QUIC so it performs identically to Google's production QUIC servers.

QUIC uses HTTP/2 and encryption on top of its reliable transport implementation. To ensure a fair comparison, we compare QUIC with HTTP/2 over TLS, atop TCP. Throughout this paper we refer to such measurements that include HTTP/2+TLS+TCP as "TCP".

Our servers add all necessary HTTP directives to avoid any caching of data. We also clear the browser cache and close all sockets between experiments to prevent "warmed up" connections from impacting results. However, we do *not* clear the state used for QUIC's 0-RTT connection establishment.

3.2. Experiments and performance metrics

Experiments. Unless otherwise stated, for each evaluation scenario (network conditions, client, and server) we conduct at least 10 measurements of each transport protocol (TCP and QUIC). To mitigate any bias from transient noise, we run experiments in 10 rounds or more, each consisting of a download using TCP and one using QUIC, back-to-back. We present the percent differences in performance between TCP and QUIC and indicate whether they are statistically significant. All tests are automated using Python scripts and Chrome's debugging tools. We use Android Debug Bridge for automating tests running on mobile phones.

Application. We test QUIC performance using the Chrome browser that currently integrates the protocol.

For Chrome, we evaluate QUIC performance using Web pages consisting of static HTML that references JPG images (various number and sizes of images) *without* any other object dependencies or scripts. While previous work demonstrates that many factors impact load times and user-perceived performance for typical, popular Web pages,^{3,18,23} the focus of this work is only on transport protocol performance. Our choice of simple pages ensures that PLT measurements reflect *only the efficiency of the transport protocol* and not browser-induced factors such as script loading and execution time. Furthermore, our simple Web pages are essential for isolating the impact of parameters such as size and number of objects on QUIC multiplexing. We leave investigating the effect of dynamic pages on performance for future work.

Performance metrics. We measure throughput, "page load time" (i.e., the time to download all objects on a page), and video quality metrics that include time to start, rebuffering events, and rebuffering time. For Web content, we use

^g Chrome "races" TCP and QUIC connections for the same server and uses the one that establishes a connection first. As such, the protocol used may vary from the intended behavior.

Chrome’s remote debugging protocol¹ to load a page and then extract HARs¹⁹ that include all resource timings and the protocol used (which allows us to ensure that the correct protocol was used for downloading an object⁸).

4. ANALYSIS

In this section, we conduct extensive measurements and analysis to understand and explain QUIC performance. We begin by focusing on the protocol-layer behavior, QUIC’s state machine, and its fairness to TCP. We then evaluate QUIC’s application-layer performance, using PLT as example application metric.

4.1. Calibration and instrumentation

In order to guarantee that our evaluation framework result in sound comparisons between QUIC and TCP, and be able to explain any performance differences we see, we (a) carefully configured our QUIC servers to match the performance of Google’s production QUIC server and (b) compiled QUIC client and server from source and instrumented them to gain access to the inner workings of the protocol (e.g., congestion control states and window sizes). Prior work did no such calibration and/or instrumentation, which explains their reported poor QUIC performance in some scenarios, and lack of root cause analysis. We refer the reader to our full paper¹⁴ for detailed discussion on our calibration and instrumentation.

4.2. State machine and fairness

In this section, we analyze high-level properties of the QUIC protocol using our framework.

State machine. QUIC has only a draft formal specification and no state machine diagram or formal model; however, the source code is made publicly available. Absent such a model, we took an empirical approach and used traces of QUIC execution to infer the state machine to better understand the dynamics of QUIC and their impact on performance.

Specifically, we use Synoptic⁷ for *automatic* generation of QUIC state machine. While static analysis might generate a more complete state machine, a complete model is not necessary for understanding performance changes. Rather, as we show in Section 4.3, we only need

to investigate the states visited and transitions between them at runtime.

Figure 2a shows the QUIC state machine automatically generated using traces from executing QUIC across all of our experiment configurations. The diagram reveals behaviors that are common to standard TCP implementations, such as connection start (Init, SlowStart), congestion avoidance (CongestionAvoidance), and receiver-limited connections (ApplicationLimited). QUIC also includes states that are non-standard, such as a maximum sending rate (CongestionAvoidanceMaxed), tail loss probes, and proportional rate reduction during recovery.

Note that capturing the empirical state machine requires instrumenting QUIC’s source code with log messages that capture transitions between states. In total, this required adding 23 lines of code in five files. While the initial instrumentation required approximately 10 hours, applying the instrumentation to subsequent QUIC versions required only about 30 minutes. To further demonstrate how our approach applies to other congestion control implementations, we instrumented QUIC’s experimental BBR implementation and present its state transition diagram in our full paper.¹⁴ This instrumentation took approximately 5 hours. Thus, our experience shows that our approach is able to adapt to evolving protocol versions and implementations with low additional effort.

We used inferred state machines for root cause analysis of performance issues. In later sections, we demonstrate how they helped us understand QUIC’s poor performance on mobile devices and in the presence of deep packet reordering.

Fairness. An essential property of transport-layer protocols is that they do not consume more than their fair share of bottleneck bandwidth resources. Absent this property, an unfair protocol may cause performance degradation for competing flows. We evaluated whether this is the case for the following scenarios, and present aggregate results over 10 runs in Table 1. We expect that QUIC and TCP should be relatively fair to each other because *they both use the Cubic congestion control protocol*. However, we find this is not the case at all.

- **QUIC vs. QUIC.** We find that two QUIC flows are fair to each other. We also found similar behavior for two TCP flows.

Figure 2. State transition diagram for QUIC’s Cubic CC.

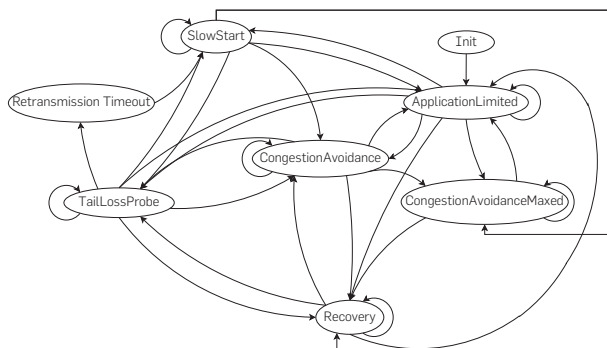


Table 1. Average throughput (5Mbps link, buffer = 30KB, averaged over 10 runs) allocated to QUIC and TCP flows when competing with each other. Despite the fact that both protocols use Cubic congestion control, QUIC consumes nearly twice the bottleneck bandwidth than TCP flows combined, resulting in substantial unfairness.

Scenario	Flow	Avg. throughput (std. dev.)
QUIC vs. TCP	QUIC	2.71 (0.46)
	TCP	1.62 (1.27)
QUIC vs. TCP×2	QUIC	2.8 (1.16)
	TCP 1	0.7 (0.21)
	TCP 2	0.96 (0.3)

- **QUIC vs. TCP.** QUIC multiplexes requests over a single connection, so its designers attempted to set Cubic congestion control parameters so that one QUIC connection emulates N TCP connections (with a default of $N = 2$ in QUIC 34, and $N = 1$ in QUIC 37). We found that N had little impact on fairness. As Figure 3a shows, QUIC is unfair to TCP as predicted, and consumes approximately twice the bottleneck bandwidth of TCP even with $N = 1$. We repeated these tests using different buffer sizes, including those used by Carlucci et al.,⁸ but did not observe any significant effect on fairness. This directly contradicts their finding that larger buffer sizes allow TCP and QUIC to fairly share available bandwidth.
- **QUIC vs. multiple TCP connections.** When competing with M TCP connections, one QUIC flow should consume $N/(M + 1)$ of the bottleneck bandwidth. However, as shown in Table 1, QUIC still consumes more than 50% of the bottleneck bandwidth even with two competing TCP flows. Thus, QUIC is not fair to TCP even assuming two-connection emulation.

To ensure fairness results were not an artifact of our test-bed, we repeated these tests against Google servers. The unfairness results were similar.

We further investigate why QUIC is unfair to TCP by instrumenting the QUIC source code, and using `tcpprobe`⁵

Figure 3. Timeline showing unfairness between QUIC and TCP when transferring data over the same 5Mbps bottleneck link (RTT = 36ms, buffer = 30KB).

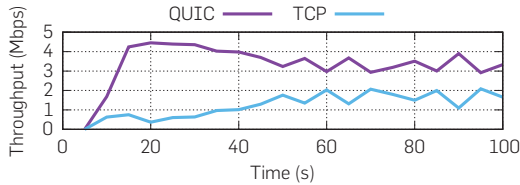
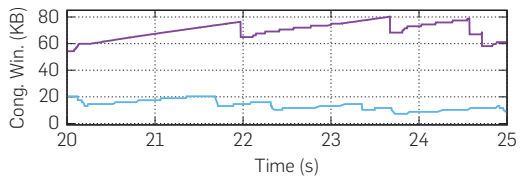
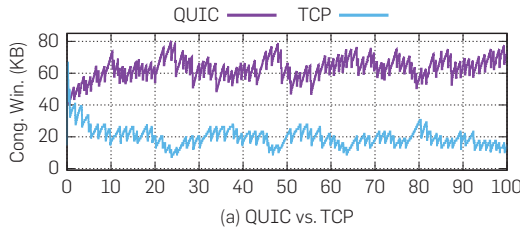


Figure 4. Timeline showing congestion window sizes for QUIC and TCP when transferring data over the same 5Mbps bottleneck link (RTT = 36ms, buffer = 30KB).



for TCP, to extract the congestion window sizes. Figure 4a shows the congestion window over time for the two protocols. When competing with TCP, QUIC is able to achieve a larger congestion window. Taking a closer look at the congestion window changes (Figure 4b), we find that while both protocols use Cubic congestion control scheme, QUIC increases its window more aggressively (both in terms of slope, and in terms of more frequent window size increases). As a result, QUIC is able to grab available bandwidth faster than TCP does, leaving TCP unable to acquire its fair share of the bandwidth.

4.3. Page load time

This section evaluates QUIC performance compared to TCP for loading Web pages (i.e., page load time, or PLT) with different sizes and numbers of objects. Recall from Section 3 that we measure PLT using information gathered from Chrome, that we run TCP and QUIC experiments back-to-back, and that we conduct experiments in a variety of emulated network settings. Note that our servers add all necessary HTTP directives to avoid caching content. We also clear the browser cache and close all sockets between experiments to prevent “warmed up” connections from impacting results. However, we do *not* clear the state used for QUIC’s 0-RTT connection establishment. Furthermore, our PLTs do not include any DNS lookups. This is achieved by extracting resource loading time details from Chrome and excluding the DNS lookup times.

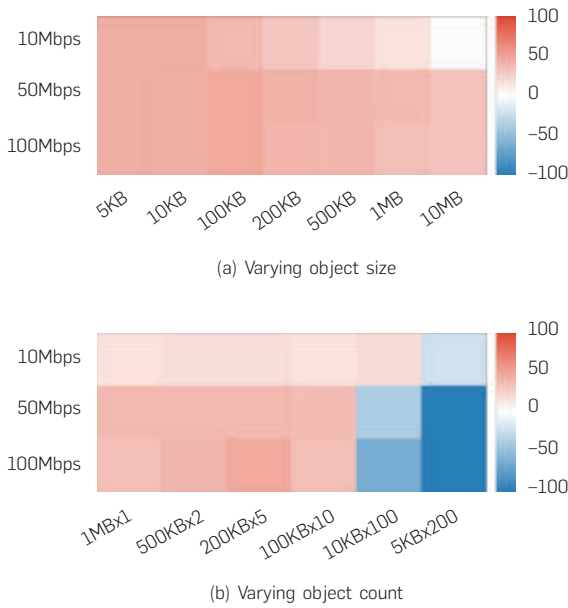
In the results that follow, we evaluate whether the observed performance differences are statistically significant or simply due to noise in the environment. We use the *Welch’s t-test*,⁶ a two-sample location test which is used to test the hypothesis that two populations have equal means. For each scenario, we calculate the *p*-value according to the Welch’s *t*-test. If the *p*-value is smaller than our threshold (0.01), then we reject the null hypothesis that the mean performance for TCP and QUIC are identical, implying the difference we observe between the two protocols is statistically significant. Otherwise the difference we observe is not significant and is likely due to noise.

Desktop environment. We begin with the desktop environment and compare QUIC with TCP performance for different rates, object sizes, and object counts—without adding extra delay or loss (RTT = 36ms and loss = 0%). Figure 5 shows the results as a heatmap, where the color of each cell corresponds to the percent PLT difference between QUIC and TCP for a given bandwidth (vertical dimension) and object size/number (horizontal direction). Red indicates that QUIC is faster (smaller PLT), blue indicates that TCP is faster, and white indicates statistically insignificant differences.

Our key findings are that QUIC outperforms TCP in every scenario except in the case of large numbers of small objects. QUIC’s performance gain for smaller object sizes is mainly due to QUIC’s 0-RTT connection establishment—substantially reducing delays related to secure connection establishment that corresponds to a substantial portion of total transfer time in these cases.

To investigate the reason why QUIC performs poorly for large numbers of small objects, we explored different values for QUIC's *Maximum Streams Per Connection (MSPC)* parameter to control the level of multiplexing (the default is 100 streams). We found there was no statistically significant impact for doing so, except when setting the MSPC value to a very low number (e.g., one), which worsens performance substantially.

Figure 5. QUIC (version 34) vs. TCP with different rate limits for (a) different object sizes and (b) with different numbers of objects. Each heatmap shows the percent difference between QUIC over TCP. Positive numbers—colored red—mean QUIC outperforms TCP and has smaller page-load time. Negative numbers—colored blue—mean the opposite. White cells indicate no statistically significant difference.



Instead, we focused on QUIC's congestion control algorithm and identified that in such cases, QUIC's *Hybrid Slow Start*¹³ causes early exit from Slow Start due to an increase in the minimum observed RTT by the sender, which Hybrid Slow Start uses as an indication that the path is getting congested. This can hurt the PLT significantly when objects are small and the total transfer time is not long enough for the congestion window to increase to its maximum value. Note that the same issue (early exit from Hybrid Slow Start) affects the scenario with a large number of *large* objects, but QUIC nonetheless outperforms TCP because it has enough time to increase its congestion window and remain at high utilization, thus compensating for exiting Slow Start early.^h

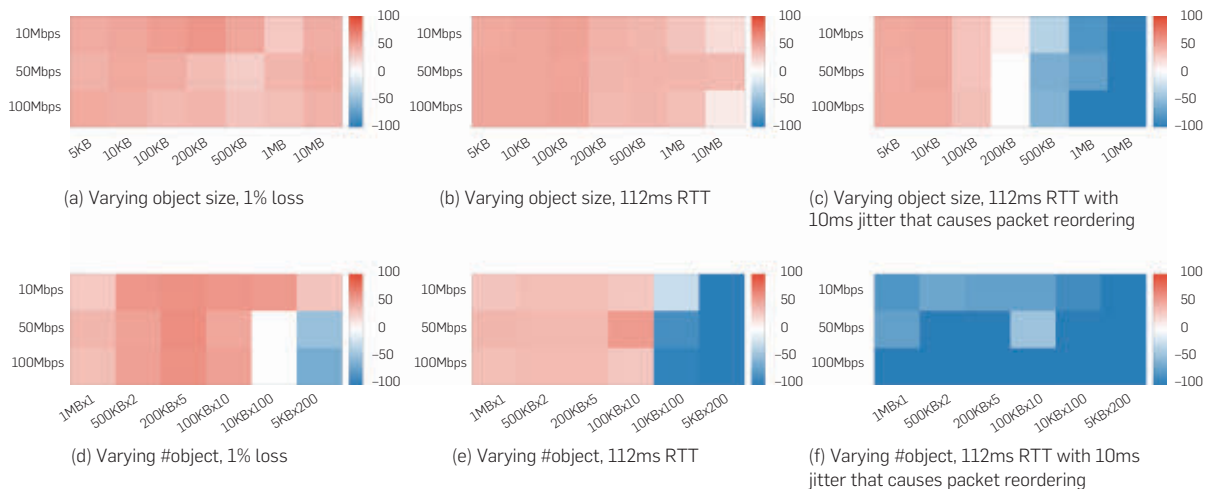
Desktop with added delay and loss. We repeat the experiments in the previous section, this time adding loss, delay, and jitter. Figure 6 shows the results, again using heatmaps.

Our key observations are that QUIC outperforms TCP under loss (due to better loss recovery and lack of HOL blocking), and in high-delay environments (due to 0-RTT connection setup). However, in the case of high latency, this is not enough to compensate for QUIC's poor performance for large numbers of small objects. Figure 7 shows the congestion window over time for the two protocols at 100Mbps and 1% loss. Similar to Figure 4, under the same network conditions QUIC better recovers from loss events and adjusts its congestion window faster than TCP, resulting in a larger congestion window on average and thus better performance.

Under variable delays, QUIC performs *significantly worse* than TCP. Using our state machine approach, we observed that under variable delay QUIC spends significantly more time in the recovery state compared to

^h We leave investigating the reason behind sudden increase in the minimum observed RTT when multiplexing many objects to future work.

Figure 6. QUIC v34 vs. TCP at different rate limits, loss, and delay for different object sizes (a, b, and c) and different numbers of objects (d, e, and f).



relatively stable delay scenarios. To investigate this, we instrumented QUIC's loss detection mechanism, and our analysis reveals that variable delays cause QUIC to incorrectly infer packet loss when jitter leads to out-of-order packet delivery. This occurs in our testbed because *netem*, which we use for network emulation, adds jitter by assigning a delay to each packet, then queues each packet based on *the adjusted send time*, not the packet arrival time—thus causing packet reordering.

The reason that QUIC cannot cope with packet reordering is that it uses a fixed threshold for number of Negative Acknowledgment (NACKs) (default 3) before it determines that a packet is lost and responds with a fast retransmit. Packets reordered deeper than this threshold cause false positive loss detection.ⁱ In contrast, TCP uses the Duplicate Selective Acknowledgment (DSACK) algorithm²⁴ to detect packet reordering and adapt its NACK threshold accordingly. As we will show later in this section, packet reordering occurs in the cellular networks we tested, so in such cases QUIC will benefit from integrating DSACK. We quantify the impact of using larger DSACK values in Figure 8, demonstrating that in the presence of packet reordering larger NACK thresholds substantially improve end to end performance compared to smaller NACK thresholds. We shared this result with a QUIC engineer, who subsequently informed us that the QUIC team is experimenting with

ⁱ Note that reordering impact when testing small objects is insignificant because QUIC does not falsely detect losses until a sufficient number of packets are exchanged.

Figure 7. Congestion window over time for QUIC and TCP at 100Mbps rate limit and 1% loss.

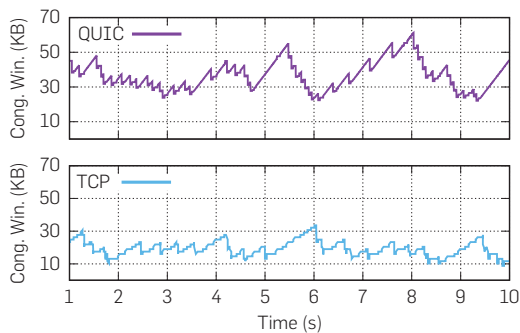
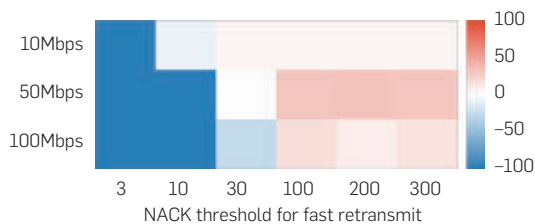


Figure 8. QUIC vs. TCP when downloading a 10MB page (112ms RTT with 10ms jitter that causes packet reordering). Increasing the NACK threshold for fast retransmit allows QUIC to cope with packet reordering.



dynamic threshold and time-based solutions to avoid falsely inferring loss in the presence of reordering.

Desktop with variable bandwidth. The previous tests set a static threshold for the available bandwidth. However, in practice such values will fluctuate over time, particularly in wireless networks. To investigate how QUIC and TCP compare in environments with variable bandwidth, we configured our testbed to change the bandwidth randomly within specified ranges and with different frequencies.

Figure 9 shows the throughput over time for three back-to-back TCP and QUIC downloads of a 210MB object when the bandwidth randomly fluctuates between 50 and 150Mbps. As shown in this figure, QUIC is more responsive to bandwidth changes and is able to achieve a higher average throughput compared to TCP. We repeated this experiment with different bandwidth ranges and change frequencies and observed the same behavior in all cases.

Mobile environment. Due to QUIC's implementation in userspace (as opposed to TCP's implementation in the OS kernel), resource contention might negatively impact performance independent of the protocol's optimizations for transport efficiency. To test whether this is a concern in practice, we evaluated an increasingly common resource-constrained deployment environment: smartphones. We use the same approach as in the desktop environment, controlling Chrome (with QUIC enabled) over two popular Android phones: the Nexus 6 and the MotoG. These phones are neither top-of-the-line, nor low-end consumer phones, and we expect that they approximate the scenario of a moderately powerful mobile device.

Figure 10 shows heatmaps for the two devices when varying bandwidth and object size.^j We find that, similar to the desktop environment, in mobile QUIC outperforms TCP in most cases; however, *its advantages diminish across the board*.

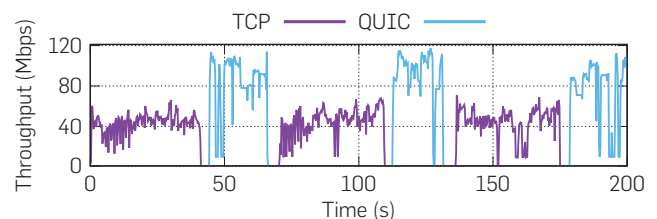
To understand why this is the case, we investigate the QUIC congestion control states visited most in mobile and non-mobile scenarios under the same network conditions.

^j We omit 100Mbps because our phones cannot achieve rates beyond 50Mbps over WiFi, and we omit results from varying the number of objects because they are similar to the single-object cases.

^k A parallel study from Google¹⁵ using aggregate data identifies the same performance issue but does not provide root cause analysis.

Figure 9. QUIC vs. TCP when downloading a 210MB object.

Bandwidth fluctuates between 50 and 150Mbps (randomly picks a rate in that range every one second). Averaging over 10 runs, QUIC is able to achieve an average throughput of 79Mbps (std. dev. = 31) while TCP achieves an average throughput of 46Mbps (std. dev. = 12).



We find that in mobile QUIC spends most of its time (58%) in the “Application Limited” state, which contrasts substantially with the desktop scenario (only 7% of the time). The reason for this behavior is that QUIC runs in a user-space process, whereas TCP runs in the kernel. As a result, QUIC is unable to consume received packets as quickly as on a desktop, leading to suboptimal performance, particularly when there is ample bandwidth available.^k Table 2 shows the fraction of time (based on server logs) QUIC spent in each state in both environments for 50Mbps with no added latency or loss. By revealing the changes in time spent in each state, such inferred state machines help diagnose problems and develop a better understanding of QUIC dynamics.

5. CONCLUDING DISCUSSION

In this paper, we address the problem of evaluating an application-layer transport protocol that was built

without a formal specification, is rapidly evolving, and is deployed at scale with nonpublic configuration parameters. To do so, we use a methodology and testbed that allows us to conduct controlled experiments in a variety of network conditions, instrument the protocol to reason about its performance, and ensure that our evaluations use settings that approximate those deployed in the wild. We used this approach to evaluate QUIC, and found cases where it performs well and poorly—both in traditional desktop and mobile environments. With the help of an inferred protocol state machine and information about time spent in each state, we explained the performance results we observed.

Additionally, we performed a number of other experiments that were omitted from this paper due to space limitations. These included testing QUIC’s performance for video streaming, tests in operational mobile networks, and impact of proxying. For more information on these experiments and our findings, we refer the reader to our full paper.¹⁴

Lessons learned. During our evaluation of QUIC, we identified several key challenges for repeatable, rigorous analyses of application-layer transport protocols in general. Below we list a number of lessons learned while addressing these challenges:

- *Proper empirical evaluation is easy to get wrong:* A successful protocol evaluation and analysis requires proper configuration, calibration, workload isolation, coverage of a wide array of test environments, rigorous statistical analysis, and root cause analysis. While this may seem obvious to the seasoned empiricist, it took us much effort and many attempts to get them right, so we leave these lessons as reminders for a general audience.
- *Models are essential for explaining performance:* Transport protocol dynamics are complex and difficult to summarize via traditional logging. We found that building an inferred state machine model and using transitions between states helped tame this complexity and offer insight into root causes for protocol performance.
- *Plan for change. As the Internet evolves, so too will transport protocols:* It is essential to develop evaluation techniques that adapt easily to such changes to provide consistent and fair comparisons over time.
- *Do not forget to look at the big picture:* It’s easy to get caught up in head-to-head comparisons between a flow from one protocol versus another. However, in the wide area there may be thousands or more flows competing over the same bottleneck link. In our limited fairness study, we found that protocol differences in isolation are magnified at scale. Thus, it is important to incorporate analysis of interactions between flows when evaluating protocols.

Figure 10. QUICv34 vs. TCP for varying object sizes on a Nexus6 smartphone (using WiFi). We find that QUIC’s improvements diminish or disappear entirely when running on mobile devices.

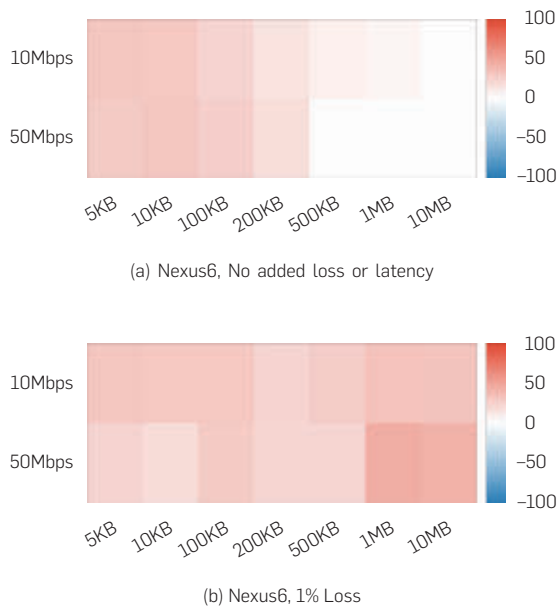


Table 2. The fraction of time QUIC spent in each state on MotoG vs. Desktop. QUICv34, 50Mbps, no added loss or delay. The table shows that poor performance for QUIC on mobile devices can be attributed to applications not processing packets quickly enough. Note that the zero probabilities are due to rounding.

	Desktop	Mobile
Init	0.01%	0.01%
Slow start	1.65%	0.42%
Application limited	7.05%	58.84%
Congestion avoidance	91.28%	40.55%
Tail loss probe	0.00%	0.00%
Recovery	0.00%	0.18%

References

1. Chrome debugging protocol. <https://developer.chrome.com/devtools/docs/debugger-protocol>.
2. <https://github.com/quicwg/baserafts/wiki/implementations>.
3. I. grigorik. Deciphering the critical rendering path. <https://calendar.perfplanet.com/2012/deciphering-the-critical-rendering-path/>.
4. QUIC at 10,000 feet. <https://docs.google.com/>

- document/d/1gY9-YNdNAB1eip-RTpBqphgYwSvNSDHLq9D5Bty4FSU.
5. TCP probe. <https://wiki.linuxfoundation.org/networking/tcpprobe>.
 6. Welch's t-test. https://en.wikipedia.org/wiki/Welch%27s_t-test.
 7. Beschastnikh, I., Brun, Y., Schneider, S., Sloan, M., Ernst, M.D. Leveraging existing instrumentation to automatically infer invariant-constrained models. In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering* (2011).
 8. Carlucci, G., De Cicco, L., Mascolo, S. HTTP over UDP: An experimental investigation of QUIC. In *Proc. of SAC* (2015).
 9. Chromium Blog. A QUIC update on Google's experimental transport, April 2015. <http://blog.chromium.org/2015/04/a-quic-update-on-googles-experimental.html>.
 10. Cimpanu, C. Google creates new algorithm for handling TCP traffic congestion control, September 2016. <http://news.softpedia.com/news/google-creates-new-algorithm-for-handling-tcp-traffic-congestion-control>.
 11. Das, S.R. Evaluation of QUIC on Web page performance. Master's thesis, Massachusetts Institute of Technology (2014).
 12. Dukkkipati, N., Cardwell, N., Cheng, Y., Mathis, M. Tail loss probe (TLP): An algorithm for fast recovery of tail losses, February 2013. <https://tools.ietf.org/html/draft-dukkkipati-tcpm-tcp-loss-probe-01>.
 13. Ha, S., Rhee, I. Taming the elephants: New TCP slow start. *Comput. Netw.*, 2011.
 14. Kakhki, A.M., Jero, S., Choffnes, D., Nita-Rotaru, C., Mislove, A. Taking a long look at QUIC: An approach for rigorous evaluation of rapidly evolving transport protocols. In *Proceedings of the 2017 Internet Measurement Conference* (2017).
 15. Langley, A., Riddoch, A., Wilk, A., Vicente, A., Krasic, C., Zhang, D., Yang, F., Kouranov, F., Swett, I., Iyengar, J., Bailey, J., Dorfman, J., Kulik, J., Roskind, J., Westin, P., Tennen, R., Shade, R., Hamilton, R., Vasiliev, V., Chang, W.-T., Shi, Z. The QUIC transport protocol: Design and Internet-scale deployment. In *Proc. of ACM SIGCOMM* (2017).
 16. Mathis, M., Dukkkipati, N., Cheng, Y. Proportional rate reduction for TCP, May 2013. <https://tools.ietf.org/html/rfc6937>.
 17. Megyesi, P., Krämer, Z., Molnár, S. How quick is QUIC? In *Proc. of ICC* (May 2016).
 18. Nikraves, A., Yao, H., Xu, S., Choffnes, D.R., Mao, Z.M. Mobilyzer: An open platform for controllable mobile network measurements. In *Proc. of MobiSys* (2015).
 19. Odvarko, J., Jain, A., Davies, A. HTTP Archive (HAR) format, August 2012. <https://dvc.w3.org/hg/webperf/raw-file/tip/specs/HAR/Overview.html>.
 20. Swett, I. QUIC congestion control and loss recovery. <https://docs.google.com/presentation/d/1T9GtMz1CvPpZtmF8g-W7j9XHZBOCp9cu1fW0sMsmppoo>.
 21. Swett, I. QUIC deployment experience "Google, 2016. <https://www.ietf.org/proceedings/96/slides/slides-96-quic-3.pdf>.
 22. Swett, I. QUIC FEC v1, February 2016. <https://docs.google.com/document/d/1Hg1SaLEl6T4rEU9j-isoVCo8VEjnuCPTcLnJewj7Nk/edit>.
 23. Wang, X.S., Balasubramanian, A., Krishnamurthy, A., Wetherall, D. How speedy is SPDY? In *Proc. of USENIX NSDI* (2014).
 24. Zhang, M., Karp, B., Floyd, S., Peterson, L. RR-TCP: A reordering-robust TCP with DSACK. In *Proceedings of the 11th IEEE International Conference on Network Protocols, ICNP '03* (Washington, DC, USA, 2003). IEEE Computer Society.

Arash Molavi Kakhki (arash@thousandeyes.com), ThousandEyes, Boston, MA, USA.

Samuel Jero (sjero@purdue.edu), Purdue University, West Lafayette, IN, USA.

David Choffnes and Alan Mislove ([choffnes, amislove]@ccs.neu.edu), College of Computer and Information Science, Northeastern University, Boston, MA, USA.

Cristina Nita-Rotaru (c.nitarotaru@neu.edu), College of Computer and Information Science, Northeastern University, Boston, MA, USA.

© 2019 ACM 0001-0782/19/7 \$15.00



上海科技大学
ShanghaiTech University



TENURE-TRACK AND TENURED POSITIONS

ShanghaiTech University invites highly qualified candidates to fill multiple tenure-track/tenured faculty positions as its core founding team in the School of Information Science and Technology (SIST). We seek candidates with exceptional academic records or demonstrated strong potentials in all cutting-edge research areas of information science and technology. They must be fluent in English. English-based overseas academic training or background is highly desired.

ShanghaiTech is founded as a world-class research university for training future generations of scientists, entrepreneurs, and technical leaders. Boasting a new modern campus in Zhanjiang Hightech Park of cosmopolitan Shanghai, ShanghaiTech shall trail-blaze a new education system in China. Besides establishing and maintaining a world-class research profile, faculty candidates are also expected to contribute substantially to both graduate and undergraduate educations.

Academic Disciplines: Candidates in all areas of information science and technology shall be considered. Our recruitment focus includes, but is not limited to: computer architecture, software engineering, database, computer security, VLSI, solid state and nano electronics, RF electronics, information and signal processing, networking, security, computational foundations, big data analytics, data mining, visualization, computer vision, bio-inspired computing systems, power electronics, power systems, machine and motor drive, power management IC as well as inter-disciplinary areas involving information science and technology.

Compensation and Benefits: Salary and startup funds are highly competitive, commensurate with experience and academic accomplishment. We also offer a comprehensive benefit package to employees and eligible dependents, including on-campus housing. All regular ShanghaiTech faculty members will join its new tenure-track system in accordance with international practice for progress evaluation and promotion.

Qualifications:

- Strong research productivity and demonstrated potentials;
- Ph.D. (Electrical Engineering, Computer Engineering, Computer Science, Artificial Intelligence, Financial Engineering, Signal Processing, Operation Research, Applied Math, Statistics or related field);
- A minimum relevant (including PhD) research experience of 4 years.

Applications: Submit (in English, PDF version) a cover letter, a 2-page research plan, a CV plus copies of 3 most significant publications, and names of three referees to: sist@shanghaitech.edu.cn. For more information, visit <http://sist.shanghaitech.edu.cn/2017/0426/c2865a23763/page.htm>

Deadline: The positions will be open until they are filled by appropriate candidates.

Advertise with ACM!

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

Request a media kit
with specifications and pricing:

Ilia Rodriguez

+1 212-626-0686

acmm mediasales@acm.org



Association for
Computing Machinery

ACM Journal of Data and Information Quality

Providing Research and Tools for Better Data

ACM Journal of Data and Information Quality (JDIQ) is a multi-disciplinary journal that attracts papers ranging from theoretical research to algorithmic solutions to empirical research to experiential evaluations. Its mission is to publish high impact articles contributing to the field of data and information quality (IQ). Research contributions can range from modeling and measurement of quality, to improvement of quality with data cleansing methods, to organizational management of quality, to evaluations of quality in real scenarios. Given the diversity of disciplines and author interests, we also welcome experience papers, typically submitted by a practitioner or industrial researcher who has a compelling application, interesting dataset or valuable teaching tool, to share with our readers. Finally, we are accepting two-page vision papers that describe a major research challenge to the JDIQ community.

JDIQ welcomes high-quality research contributions from the following areas, but not limited to:

- Concepts, Methods and Tools
- Organizations and IQ
- Measurement, Improvement and Assurance of IQ
- Information Quality for Specialized Domains and Applications



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

Subscribe at www.acm.org/subscribe

[CONTINUED FROM P. 96] **Question.** Suppose T insists on the Distributed strategy all the time. Can L capture all the smugglers in 20 days possibly by delaying some containers from leaving the port by a day or two and reallocating agents?

Solution. By allocating all 1000 agents to one port for two days— d and $d+1$ —the agents can go through all the containers of day d and identify all the smugglers. This is called the Carpeting Strategy.

So if the game is played long enough, the Carpeting Strategy should encourage the trafficker T to use the Centralized strategy, because more smugglers will avoid arrest for longer. However, a compromise is possible: T might use the “Semi-Distributed strategy” in which 10 smugglers each receive 10 opioid packages and each smuggler puts those packages in one of that smuggler’s containers. This has the same expected loss to T as the Distributed strategy in terms of the number of opioid packages that can be lost to agents (if there are 100 agents in that port), but it puts fewer smugglers in danger if L uses the Carpeting Strategy.

Upstart 1. Given the setting of this problem as described here and given that L uses the Random with Teamwork strategy and keeps 100 agents at every port, how many days must the game be played for Semi-Distributed to be worse for T than Centralized measured in terms of the total number of opioid packages delivered?

Upstart 2. Answer the same question when T uses the Carpeting Strategy.

Upstart 3. If L uses the Carpeting Strategy for every pair of days with a certain probability p but otherwise uses the Random with Teamwork Strategy, what is the best strategy for T over some number k days where T can choose between Centralized, Distributed, and Semi-Distributed? □

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, USA, as well as the chronicler of his good friend the omniheurist Dr. Ecco.

Copyright held by author.



DOI:10.1145/3332802

Dennis Shasha

Upstart Puzzles

Opioid Games

THERE ARE TWO sources of illegal opioids: legitimately manufactured ones that have been diverted to drug dealers, and criminally manufactured ones that were always intended for dealers. A special video dispenser, markings on the pills, and a little machine vision can go a long way toward dealing with the legitimately manufactured ones. This column, however, is concerned about with the criminally manufactured opioid pills.

The setting is 10 ports of entry. For simplicity each port has 1000 importers of whom 100 are smugglers (but law enforcement does not know who they are). Each importer brings in 10 containers per day (so 10,000 containers per port per day) and there are 10 packages per container. One agent can inspect five containers per day. Law enforcement (L) has 1000 agents altogether for all 10 ports.

The trafficker (T) wants to bring in 100 opioid packages a day through each port and may allocate them in many ways among smugglers. Law enforcement (L) wants to capture as many opioid packages as possible.

Good strategies for each party depend on how many days this “game” goes on.

If we consider a game of just one day and L distributes 100 agents to each port, then consider two extremal strategies between which T could choose for each port:

- (Centralized) Give all 100 opioid packages to one smuggler to put in evenly among that smuggler’s 10 containers; and
- (Distributed) Give one opioid package to each of 100 smugglers. Each smuggler will put the opioid package in a random one of the 10 containers that smuggler imports.



Suppose L allocates its agents randomly to 100 of the 1000 importers and each agent inspects five containers of that importer. L instructs the agents to modify the purely random strategy with what L calls the “Teamwork Modification:” if an agent finds an opioid package then another agent will help the first one so together they will inspect all the containers of that importer. We call this combined strategy the Random with Teamwork strategy.

Warm-up. What is the expected number of opioid packages that L will capture, when using the Random with Teamwork strategy, based on each extremal strategy of T ?

Solution to Warm-Up. The expectation for the Distributed strategy is the number of agents times the probability that an agent is inspecting a smuggler’s containers times the probability that the five containers inspected by the

agent contain at least one opioid package times the number of opioid packages the agent will find: $100 * (1/10) * (1/2) * 1 = 5$ opioid packages. For the Centralized strategy, the expected capture rate is $100 * (1/1000) * 100 = 10$ opioid packages, because the Teamwork Modification strategy suggests that other agents will help the one who finds a first opioid package. Together, they will find all 100 in the 1-in-10 chance that some agent finds at least one opioid package.

Conclusion. The Distributed strategy enjoys a greater expected number of opioid packages that get through for T than the Centralized strategy.

The consequences for the smugglers are somewhat different however. With the Distributed strategy, approximately five smugglers would be captured. With the Centralized strategy, at most one and only with probability $1/10$. [CONTINUED ON P. 95]



2019 SYDNEY, AUSTRALIA

**NOVEMBER
12TH - 15TH, 2019**

VIRTUAL

REALITY

SOFTWARE

AND TECHNOLOGY

**FOR MORE INFO VISIT
VRST.ACM.ORG/VRST2019**



Association for
Computing Machinery



SIGCHI



ACM SIGGRAPH



WESTERN SYDNEY
UNIVERSITY



**SIGGRAPH
ASIA 2019
BRISBANE**

The 12th ACM SIGGRAPH Conference
and Exhibition on Computer Graphics
and Interactive Techniques in Asia

DREAM ZONE!

Conference 17 - 20 November 2019
Exhibition 18 - 20 November 2019

Brisbane Convention & Exhibition Centre (BCEC),
Brisbane, Australia

Sponsored by:



Organized by

