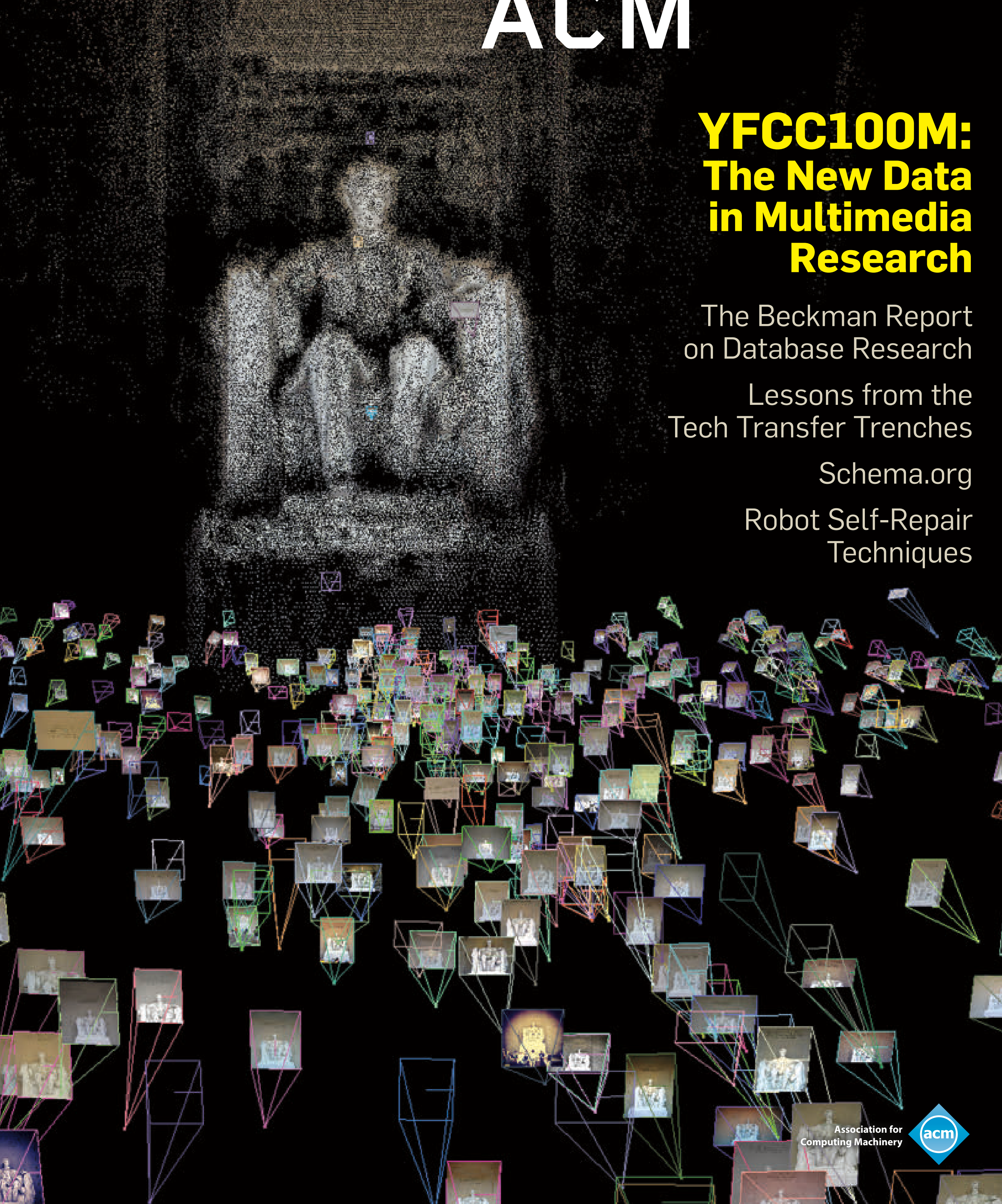
COMMUNICATIONS OF THE ACM

CACM.ACM.ORG

02/2016 VOL.59 NO.02



YFCC100M: The New Data in Multimedia Research

The Beckman Report on Database Research

Lessons from the Tech Transfer Trenches

Schema.org

Robot Self-Repair Techniques

Association for Computing Machinery

10th ACM International Conference on Distributed and Event-based Systems

Irvine, CA
June 20-24, 2016

Photo: Crystal Cove State Beach by brent, CC BY

Call for Papers

Supported by

Over the past decade, the DEBS conference has become the premier venue for contributions in the fields of distributed and event-based systems. Its objectives are to provide a forum dedicated to the dissemination of original research, the discussion of practical insights, and the reporting of experiences relevant to distributed systems and event-based computing. The conference aims at providing a forum for academia and industry to exchange ideas through industry papers and demo papers.

Starting this year, DEBS is extending its scope to embrace a broader set of topics. Topics of particular interest may include (but are not limited to) models, architectures and paradigms of distributed and event-based systems, middleware systems and frameworks, and applications, experiences and requirements. The scope of the DEBS conference covers all topics relevant to distributed and event-based computing ranging from those discussed in related disciplines (e.g., software systems, distributed systems, distributed data processing, data management, dependability, knowledge management, networking, programming languages, security and software engineering), to domain-specific topics of event-based computing (e.g., real-time analytics, mobile computing, social networking, pervasive, green computing and ubiquitous computing, sensors networks, user interfaces, big data processing, spatio-temporal processing, cloud computing, the Internet of things, peer-to-peer computing, embedded systems and stream processing), to enterprise-related topics (e.g., complex event detection, enterprise application integration, real-time enterprises and web services). In addition to these traditional topics, the scope of DEBS 2016 will include the increasingly important area of Internet of Things. New advances in distributed and event-based systems pose a great potential for a major contribution in this area.

DEBS 2016 will be organized along six tracks:
1. **Research Track**, presenting original research contributions.
2. **Industry and Experience Reports Track**, reporting on innovative deployments of event-based systems.
3. **Tutorial Track**, where experts in the field present their tutorials on relevant emerging areas of research.
4. **Poster and Demo Track**, reporting on work in progress and/or demonstrations of event-based systems.
5. **Doctoral Symposium Track**, meant for doctoral candidates conducting research in event-based systems.
6. **Grand Challenge Track**, setting out a challenge problem and seeking innovative approaches for its solution.

Important Dates

Abstract submission: February 22, 2016
Full papers submission: February 27, 2016
Notifications: April 20, 2016

Doctoral Symposium: June 20, 2016
Conference: June 21-23, 2016
Workshops: June 24, 2016

Conference Committee

General Chairs
Avigdor Gal, Technion
Matthias Weidlich, Humboldt-Universität zu Berlin
Program Chairs
Vana Kalogeraki, Athens Univ. of Economics and Business
Nalini Venkatasubramanian, UC Irvine
Industry Chairs
Alejandro Buchmann, Technische Universität Darmstadt
Malu Castellanos, HP
Doctoral Symposium Chairs
David Eyers, University of Otago
Leonardo Querzoni, Sapienza University
Tutorial Chairs
Gianpaolo Cugola, Politecnico di Milano
Bugra Gedik, Bilkent University
Proceedings Chair
Thomas Heinze, SAP

Grand Challenge Chairs
Zbigniew Jerzak, SAP AG
Vincenzo Gulisano, Chalmers University of Technology
Holger Ziekow, University of Applied Sciences Furtwangen
Demo and Poster Track Chairs
Ioannis Katakis, University of Athens
Nesime Tatbul, Intel Labs/MIT
Sponsorship Chairs
Christoph Emmersberger, University of Regensburg
Opher Etzion, Yezreel Valley College
Publicity Chairs
Mohammad Sadoghi, IBM T.J. Watson Research Center
Izchak Tzachi Sharfman, Technion
Vinay Setty, MPI
Web Chairs
Matthew Forshaw, Newcastle University
Ye Zhao, Google

http://www.debs2016.org/

ACM Books

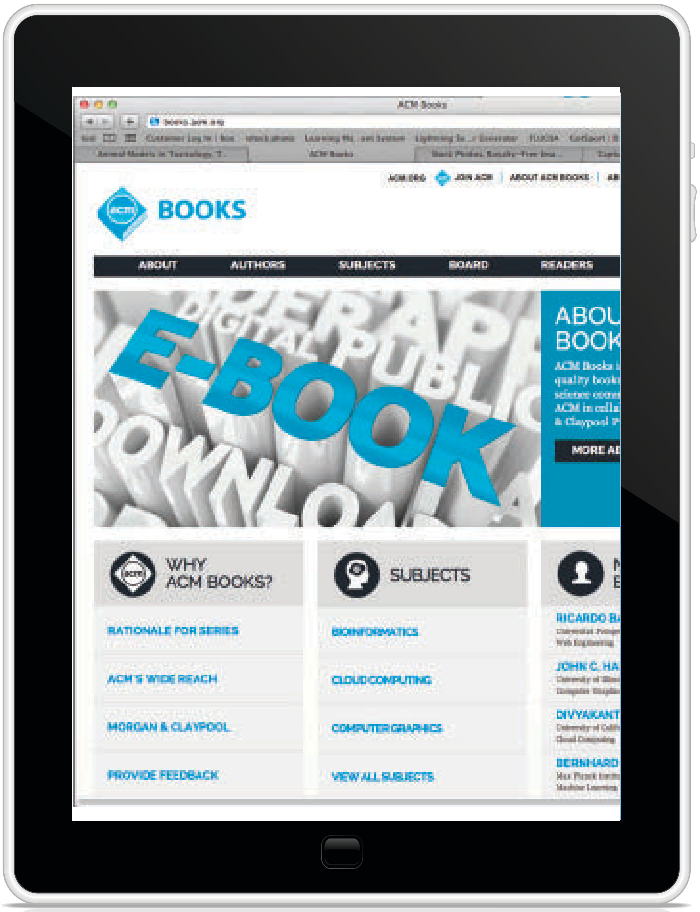
MORGAN & CLAYPOOL PUBLISHERS

Publish your next book in the ACM Digital Library

ACM Books is a new series of advanced level books for the computer science community, published by ACM in collaboration with Morgan & Claypool Publishers.

> *I'm pleased that ACM Books is directed by a volunteer organization headed by a dynamic, informed, energetic, visionary Editor-in-Chief (Tamer Özsu), working closely with a forward-looking publisher (Morgan and Claypool).*
>
> —Richard Snodgrass, University of Arizona

books.acm.org



ACM Books

- will include books from across the entire spectrum of computer science subject matter and will appeal to computing practitioners, researchers, educators, and students.
- will publish graduate level texts; research monographs/overviews of established and emerging fields; practitioner-level professional books; and books devoted to the history and social impact of computing.
- will be quickly and attractively published as ebooks and print volumes at affordable prices, and widely distributed in both print and digital formats through booksellers and to libraries and individual ACM members via the ACM Digital Library platform.
- is led by EIC M. Tamer Özsu, University of Waterloo, and a distinguished editorial board representing most areas of CS.

Proposals and inquiries welcome!

Contact: **M. Tamer Özsu,** Editor in Chief

booksubmissions@acm.org

Association for Computing Machinery

Advancing Computing as a Science & Profession

COMMUNICATIONS OF THE ACM

Departments

Last Byte

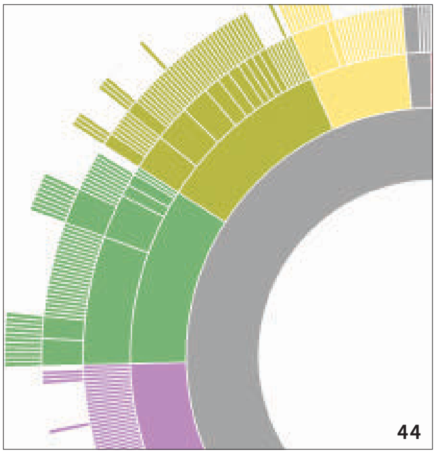
Association for Computing Machinery
Advancing Computing as a Science & Profession

News

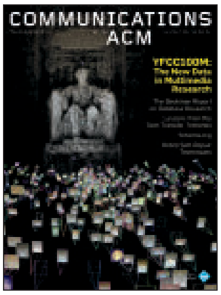
Viewpoints

PHOTO BY JOCK FISTICK

Practice



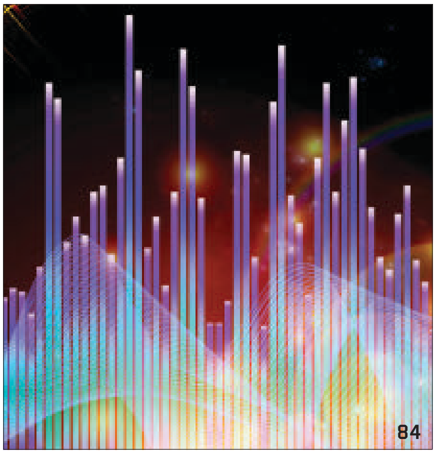
Articles' development led by acmqueue
queue.acm.org



About the Cover: This month's cover story tells of one of the largest datasets ever created and freely shared. The Yahoo Flickr Creative Commons 100M Dataset consists of 100 million media objects and is the largest public multimedia collection. The cover offers an example of what can be created using the YFCC100M dataset, in this case a 3D-reconstruction based on 1,000 Flickr photos as assembled by co-author David A. Shamma.

IMAGES BY FABIO VALSECCHI; AGSANDREW

Contributed Articles



Watch the authors discuss their work in this exclusive *Communications* video. http://cacm.acm.org/videos/yfcc100m-the-new-data-in-multimedia-research

Review Articles

Research Highlights

Watch the authors discuss their work in this exclusive *Communications* video. http://cacm.acm.org/videos/pinocchio-nearly-practical-verifiable-computation

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

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

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

Executive Director and CEO
Bobby Schnabel
Deputy Executive Director and COO
Patricia Ryan
Director, Office of Information Systems
Wayne Graves
Director, Office of Financial Services
Darren Ramdin
Director, Office of SIG Services
Donna Cappo
Director, Office of Publications
Bernard Rous
Director, Office of Group Publishing
Scott E. Delman

ACM COUNCIL

President
Alexander L. Wolf
Vice-President
Vicki L. Hanson
Secretary/Treasurer
Erik Altman
Past President
Vinton G. Cerf
Chair, SGB Board
Patrick Madden
Co-Chairs, Publications Board
Jack Davidson and Joseph Konstan
Members-at-Large
Eric Allman; Ricardo Baeza-Yates; Cherri Pancake; Radia Perlman; Mary Lou Soffa; Eugene Spafford; Per Stenström
SGB Council Representatives
Paul Beame; Barbara Boucher Owens

BOARD CHAIRS

Education Board
Mehran Sahami and Jane Chu Prey
Practitioners Board
George Neville-Neil

REGIONAL COUNCIL CHAIRS

ACM Europe Council
Fabrizio Gagliardi
ACM India Council
Srinivas Padmanabhuni
ACM China Council
Jiaguang Sun

PUBLICATIONS BOARD

Co-Chairs
Jack Davidson; Joseph Konstan
Board Members
Ronald F. Boisvert; Anne Condon; Nikil Dutt; Roch Guerrin; Carol Hutchins; Yannis Ioannidis; Catherine McGeoch; M. Tamer Ozsu; Mary Lou Soffa; Alex Wade; Keith Webster

ACM U.S. Public Policy Office
Renee Dopplick, Director
1828 L Street, N.W., Suite 800
Washington, DC 20036 USA
T (202) 659-9711; F (202) 667-1066

Computer Science Teachers Association
Lissa Clayborn, Acting Executive Director

STAFF

DIRECTOR OF GROUP PUBLISHING
Scott E. Delman
cacm-publisher@cacm.acm.org

Executive Editor
Diane Crawford
Managing Editor
Thomas E. Lambert
Senior Editor
Andrew Rosenbloom
Senior Editor/News
Larry Fisher
Web Editor
David Roman
Rights and Permissions
Deborah Cotton

Art Director
Andrij Borys
Associate Art Director
Margaret Gray
Assistant Art Director
Mia Angelica Balaquiot
Designer
Iwona Usakiewicz
Production Manager
Lynn D'Addesio
Director of Media Sales
Jennifer Ruzicka
Publications Assistant
Juliet Chance

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

CONTACT POINTS

Copyright permission
permissions@cacm.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

AUTHOR GUIDELINES

http://cacm.acm.org/

ACM ADVERTISING DEPARTMENT

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

Director of Media Sales
Jennifer Ruzicka
jen.ruzicka@hq.acm.org

Media Kit acmmediasales@acm.org

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

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

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

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

COMMUNICATIONS OF THE ACM

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

POSTMASTER

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

Printed in the U.S.A.

Association for Computing Machinery

PLEASE RECYCLE THIS MAGAZINE

DOI:10.1145/2870659

Moshe Y. Vardi

The Moral Hazard of Complexity-Theoretic Assumptions

IN NOVEMBER 2015, the computing world was abuzz with the news that László Babai proved the Graph-Isomorphism Problem, that is, the long-standing open problem of checking whether two given graphs are isomorphic, can be solved in quasi-polynomial time. (While polynomial means *n* raised to a fixed degree, quasi-polynomial means *n* raised to a poly-logarithmic degree.) The mainstream media noticed this news; *Chicago Magazine* wrote "Computer Scientists and Mathematicians Are Stunned by This Chicago Professor's New Proof."

If Babai's result holds under scrutiny, it is likely to become one of the most celebrated results in theoretical computer science of the past several decades. Graph isomorphism has long tantalized researchers as it was not known to be solvable in polynomial time, yet there are good reasons to believe it is not NP-complete. The new algorithm provides further evidence of that!

As excited as I was about this news, I was quite annoyed by the reports in the media that "A new algorithm efficiently solves the graph-isomorphism problem." Computational-complexity theory focuses on classifying computational problems according to their inherent difficulty. The theory relies on some fundamental abstractions, including that it is useful to classify algorithms according to their worst-case complexity, as it gives us a universal performance guarantee, and that polynomial functions display moderate growth. The complexity class PTIME, often abbreviated as P, consists of the problems that can be solved, in the worst case, in polynomial time. This class is now conventionally considered to be the class of efficiently solvable problems. But this identification of P with the class of efficiently solvable problems is just a mathematically convenient abstraction. In practice, polynomials with degree larger than four grow quite fast. It is unrealistic to consider algorithms with high-degree polynomial bounds as efficient. To consider quasi-polynomial-time algorithms as efficient is simply ignorant of the reality of algorithmic engineering.

The slide from considering polynomial-time algorithms as efficient to considering quasi-polynomial-time algorithms as efficient illustrates, I believe, what I'd like to call the "moral hazard of complexity-theoretic assumptions." In economics, moral hazard occurs when one person takes more risks because someone else bears the burden of those risks. Here, I would like to use the term to imply as one gets used to the "risk" of making complexity-theoretic assumptions, it gets easier to make stronger assumptions. The slide from polynomial time as efficient to quasi-polynomial time as efficient is an instance, I believe, of such moral hazard in action.

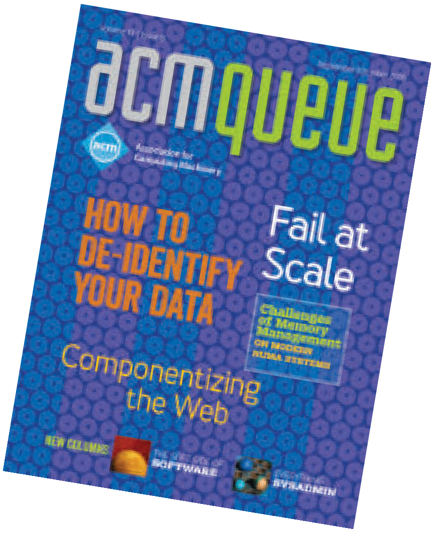
Another example of such a hazardous slide can be seen in a news article from August 2015 with the title "For 40 Years, Computer Scientists Looked for a Solution that Doesn't Exist." The story refers to a paper by Arturs Backurs and Piotr Indyk, "Edit Distance Cannot Be Computed in Strongly Subquadratic Time (unless SETH is false)." What is SETH? The conventional wisdom is that P is different from NP. Thus, this is now taken as a standard assumption, even though the belief in it is mostly built around the questionable identification of P with efficiently solvable. In some cases, however, the P≠NP assumption is not strong enough. The Strong Exponential-Time Hypothesis (SETH) is a stronger assumption that asserts that Boolean Satisfiability (SAT) cannot be solved in strongly subexponential time (see the Backurs-Indyk paper for a precise definition).

Proving that SETH implies that edit distance requires quadratic time is already a very nice result. But the title of the paper implies one should not expect SETH to be false, which is the way the result was broadly interpreted. But SETH is a much stronger assumption than the P≠NP assumption. The evidence for this assumption is that so far no one has been able to come up with a strongly subexponential algorithm for SAT. But most progress in complexity theory over the past 40 years has been through the discovery of clever new algorithms, such as Khachian's Ellipsoid Algorithm for linear programming, Shor's quantum poly-time algorithm for factoring, and now Babai's quasi-polytime algorithm for graph isomorphism. If I had to bet, I would bet on further progress in improved upper bounds than on progress in improved lower bounds.

The bottom line is that complexity-theoretic assumptions are mathematical abstractions. They need to be refined further to bring them into better alignment with real-world tractability. As I have written in the past, there is a growing gap between current theory and practice of complexity and algorithms. Bridging that gap is an important research challenge!

Follow me on Facebook, Google+, and Twitter.

Moshe Y. Vardi, EDITOR-IN-CHIEF


acmqueue
HOW TO DE-IDENTIFY YOUR DATA
Fail at Scale
Challenges of Memory Management on Modern NUMA Systems
Componentizing the Web
NEW COLUMNS
SOFTWARE
SYSADMIN

DOI:10.1145/2872420

Vinton G. Cerf

Apps and the Web

Much has already been written about the differences and even the rivalry between native mobile and mobile Web applications. Part of this seems explainable owing to the limited display and interaction capacity of a mobile. Specialized applications that use limited display space and soft keyboards have been crafted to match the typical mobile environment. Ironically, mobile apps and browser-based applications both rely on servers on the Internet for much of their functionality. Actions by users of either interface reach common ground at Internet servers and the results may be visible by either the Web or mobile operating system interfaces (consider email and social networking applications, for example). While there is standardization in the form of HTTP and various versions of HTML for Web-based applications that allow a browser to pull or push content from or to Web servers, interaction among mobile apps is rarer for lack of commonality other than sharing the interface to the mobile platform itself and shared access to common information provided by the servers.

There are trade-offs to be found between a native mode implementation of an application and a browser-based implementation. State information is often kept within a native mode application so it can still function to some degree in the absence of access to the Internet while a browser is typically dependent on state information that is maintained at the Web server with the browser acting largely as a display service. Ironically, early implementations such as the Netscape browser and all subsequent ones have provisions for storing state information in the form of cookies so servers need not to retain session state across intervals when the network connection is broken or users abandon a session. The reality is both kinds of application implementations have local processing capacity, especially with the advent of Java, JavaScript, HTML5, and other high-level language interpreters. Web-based applications often have the property that users can move from platform to platform (mobile, to tablet, to laptop) transparently because critical state has been kept on the server. Email is a good example. On the other hand, an ereader application for a mobile may be attractive for precisely the reason the ebook is locally stored and can be read, regardless of the status of access to the Internet.

In theory, Web-based applications are more portable across mobile operating systems to the extent these systems are consistent about interpreting the high-level language programs. Native mode applications, compiled to run in particular operating system environments (for example, Apple's iOS or Google's Android), may prove to be more efficient but have to be crafted to fit the operating system application programming interfaces and services available. Dependency on persistent Internet access and/or on substantial data transport between a mobile and its Internet server can have economic and performance ramifications. If the mobile data rates are limited, if there are limits to the total (for example, monthly) amount of data transferred without penalty, or if connectivity is spotty, the resulting performance may be unsatisfactory. Battery life is another major consideration. Implementations of applications that are sparing of computing and data transfer requirements will be attractive to mobile users who do not want to run out of power in the middle of a busy day.

It is also true the question is not binary. It is possible to implement hybrid applications in which some code is native mode and some is HTML-based by concealing the native code in an HTML wrapper. I am not sure how common such implementations might be and would be very interested to hear from readers with implementation experience whether or not this is a common practice. Indeed, I am very interested to hear whether this question has become moot, owing to increased mobile capacity, higher speed, more reliable access to the Internet, and longer battery life.

As the implementation of IPv6 penetrates further into the Internet, mobiles will have the ability to implement end-to-end Internet connections with other mobiles, with servers, and with Internet-enabled appliances. This introduces the possibility of peer-to-peer interactions between mobiles. Would that change the equation with regard to native mode or mobile Web implementation? Mobiles are also becoming the user interface of choice for interacting with the Internet of Things and one begins to wonder about the roles of Bluetooth, LTE, and Wi-Fi in this context, but we will have to leave that topic for another column. C

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

DOI:10.1145/2871050

Expect 'Ungoverned Actors' to Use AI-Supported Weapons, Too

Both sides of the Point/Counterpoint "The Case for Banning Killer Robots" (Dec. 2015) over lethal autonomous weapons systems (LAWS) seemed to agree the argument concerns weapons " … that once activated, would, as Stephen Goose wrote in his "Point," be able to select and engage targets without further human involvement." Arguments for and against LAWS share this common foundation, but where Goose argued for a total ban on LAWS-related research, Ronald Arkin, in his "Counterpoint," favored a moratorium while research continues. Both sides accept international humanitarian law (IHL) as the definitive authority concerning whether or not LAWS represents a humane weapon.

If I read them correctly, Goose's position was because LAWS would be able to kill on their own initiative they differ in kind from other technologically enhanced conventional weapons. That difference, he said, puts them outside the allowable scope of IHL and therefore ought to be banned. Arkin agreed LAWS differs from prior weapons systems but proposed the difference is largely their degree of autonomy and their lethal capability can be managed remotely when required. Arkin also said continued research will improve deficiencies in LAWS, thereby likely reducing the number of noncombatant casualties.

Stepping back from the debate about IHL and morality, LAWS appear to be the latest example of off-the-shelf (more-or-less) algorithms and hardware systems to be integrated into weapons systems. Given this, the debate over LAWS fundamentally concerns how far AI research should advance when it results in dual-use technologies. AI technologies clearly support driverless vehicles, aerial drones, facial recognition, and sensor-driven robotics, and are already in the public domain.

These technologies can be integrated into weapons of all sorts relatively cheaply and with only modest technical skills when equally modest levels of accuracy and reliability are acceptable. One must look only at the success of the AK-47 automatic assault rifle and Scud missiles to know relatively inexpensive weapons are often as useful as their higher-priced counterparts. A clear implication of the debate is AI research already enables development and use of LAWS-like weapons by rogue states and terrorists.

No one can expect AI researchers to stop work on what possibly could become dual-use technologies solely on the basis of such a possibility. LAWS may be excluded from national armories, but current AI technology almost assures their inevitable development and use by ungoverned actors.

Anthony Fedanzo, Corte Madera, CA

'AI Summers' Do Not Take Jobs

Artificial intelligence is a seasonal computer science field. Summers and winters appear every 15 years or so. Perhaps now we have reached an endless summer. Or not. A healthy discussion could keep expectations manageable. In his blog@cacm post "What Do We Do When the Jobs Are Gone?" (Dec. 2015), Moshe Y. Vardi wrote, "Herbert Simon was probably right when he wrote in 1956 that 'machines will be capable … of doing any work a man can do.'" Simon was not right. Our admiration for Simon will not be lessened by considering his full statement: "Machines will be capable, within twenty years, of doing any work a man can do." But 20 years passed, and then 40, and now almost 60. Some people today say it will happen within the next 20 years. Want to bet? Even the most intelligent of us can underestimate the difficulty of creating an intelligent machine. Simon was not alone; every AI summer is marked by such pronouncements. AI advances will benefit everyone in small ways. Some jobs will be eliminated. Others will be created. More technology-driven solar and wind energy jobs are created than coal-mining jobs are lost. For more than five years the U.S. has added jobs every month, more than two million each year, despite the development of more capable machines. Humans are creative and resourceful.

Jonathan Grudin, Redmond, WA

What NBA Players' Tweets Say About Emotion

The article "Hidden In-Game Intelligence in NBA Players' Tweets" by Chenyan Xu et al. (Nov. 2015) lacked, in my opinion, a complete understanding of the topics it covered. The measures it cited were not adequately reported; for example, not clear was what the dependent variable consisted of, so readers were unable to judge what the coefficients mean or the adequacy of a 1% adjusted R2 in Table 5, an effect size that was most likely meaningless.

Moreover, the sample size was not explained clearly. There were initially 91,659 tweets in the sample, and 266 players tweeted at least 100 tweets during the season in question. Other than in a small note in Table 1, the article did not mention there are only 82 games in a regular NBA season, resulting in at least 1.22 tweets per game for those 266 players; this is not an appropriate sample size, and the distribution is most likely a long tail. With 353 players tweeting and 82 games, the sample size should be 28,946 player-games (the unit of analysis), yet the reported sample size was a fraction of that—3,443 or 3,344. That would be fewer than 10 games per player and not an adequate sample size.

Also unclear was if players with more tweets before a game can have higher emotion scores, as this measure seems to be an aggregate; the article said, "The *total score* represents a player's mood ... The higher the *aggregated* score, the more positive the player's mood," emphasis added. More tweets do not mean more emotion. The article also did not address if there is a difference between original tweets and replies to other tweets.

The article also made a huge assumption about the truthfulness of tweets. NBA players are performers and know their tweets are public. The article dismissed this, saying, "Its confounding effect is minimal due to players' spontaneous and genuine use of Twitter," yet offered no evidence, whether statistical, theoretical, or factual, that this is so.

The article coded angry emoticons (such as >:-o) to the negative mood condition, as in Table 3. This emoticon-mood mapping is incorrect, as anger can be positively channeled into focus and energy on the court. Smileys and frowns were given a weighting of +/–2 on a scale of +5 to –5, but not explained was why this is theoretically defensible.

NBA coaches do not seek to maximize performance at the level of an individual player but at the level of a team as a whole across an entire game and season. Bench players usually cannot replace starting players; the starters start for very good reasons.

The article's conclusion said the authors had analyzed 91,659 tweets, yet footnote b said, "Of the 51,847 original posts, 47,468 were in English," implying they analyzed at most 87,280 tweets. Restating the number 91,659 was itself misleading, as tweets were not the unit of analysis—player-games were—and the authors had only 3,443 such observations, at most.

The one claim reviewers and editors should definitely have caught is in footnote c: A metric that can capture the unquantifiable? I am so speechless I might have to use an emoticon myself.

Nathaniel Poor, Brooklyn, NY

Authors' Response:

Our study explored whether and how NBA players' tweets can be used to extract information about players' pre-game emotional state (X) based on the psychology and sports literature and how it might affect players' in-game performance (Y). To generate X for a player before a game, we purged pure re-tweets, information-oriented tweets, and non-English tweets. Based on the remaining valid tweets, we then extracted, aggregated, and normalized the data, as in Table 5. We still find it intriguing X explains up to 1% of the total variations in Y, whereas other standard variables explain only 4%.

Chenyan Xu, Galloway, NJ,
Yang Yu, Rochester, NY, and
Chun K Hoi, Rochester, NY

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

Coming Next Month in COMMUNICATIONS

Hopes, Fears, and Obfuscation

A Lightweight Methodology for Rapid Ontology Engineering

Time to Reset Paper Presentation at Conferences

Lessons Learned from 30 Years of MINIX

Repeatability in Computer Systems Research

A Differential Approach to Undefined Behavior Detection

Riding the API Hype Cycle

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

Plus the latest news about natural language processing, algorithmic biases, and detecting crop diseases.

SHAPE THE FUTURE OF COMPUTING.
JOIN ACM TODAY.

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

SELECT ONE MEMBERSHIP OPTION

ACM PROFESSIONAL MEMBERSHIP:

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

ACM STUDENT MEMBERSHIP:

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

❑ **Join ACM-W:** ACM-W supports, celebrates, and advocates internationally for the full engagement of women in all aspects of the computing field. Available at no additional cost.

Priority Code: CAPP

Payment Information

Name

ACM Member #

Mailing Address

City/State/Province

ZIP/Postal Code/Country

Email

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

❑ AMEX ❑ VISA/MasterCard ❑ Check/money order

Total Amount Due

Credit Card #

Exp. Date

Signature

Purposes of ACM

ACM is dedicated to:

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

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

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

Satisfaction Guaranteed!

BE CREATIVE. STAY CONNECTED. KEEP INVENTING.

acm
Association for
Computing Machinery

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

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

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

BLOG@CACM

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.

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

DOI:10.1145/2852229 http://cacm.acm.org/blogs/blog-cacm

Drumming Up Support For AP CS Principles

Mark Guzdial issues a "call to action" to encourage high schools to offer, and students to take, the Advanced Placement course in Computer Science Principles.

Mark Guzdial
"A Call to Action for Higher Education to Make AP CS Principles Work"
http://bit.ly/1YFwWv0
November 15, 2015

In May 2017, the Advanced Placement (AP) examination in *CS Principles* (AP CSP) will be offered for the first time (see the AP CS Principles website at http://apcsprinciples.org/). The first high school classes to prepare students for the exam will be starting this coming Fall 2016. The existing AP CS Level A exam is not going away (see the AP CS website at http://bit.ly/1QcgLSZ).

The AP CSP course and exam were created to provide a more welcoming, a more generally useful, and a less programming-intense introduction to computer science. Individual states make up elementary and secondary school policy in the U.S. Creating a new AP course in CS is a way of providing a new path into CS to everyone. U.S. high schools want to offer AP courses to their students. Having more high schools offering a more accessible AP computing class can help to increase access to high-quality computing education at the high school level—if students take the class.

It is in the interest of computing departments in higher education to support AP CSP, in order to get a more diverse student body interested in CS and coming to our campuses. For AP CSP work, CS faculty at colleges and universities need to take action. Let me explain what I mean in terms of three questions—whose answers I do not know.

Question 1: Will colleges and universities offer CSP-equivalent classes?

The Wikipedia page on Advanced Placement (at https://en.wikipedia.org/wiki/Advanced_Placement) explains the program this way:

Advanced Placement (AP) is a program in the U.S. and Canada, created by the College Board, which offers college-level curricula and examinations to high school students. American colleges and universities often grant placement and course credit to students who obtain high scores on the examinations. Some colleges use AP test scores to exempt students from introductory coursework, others use them to place students in higher designated courses, and some do both.

In general, AP courses are designed to replicate existing college-level introductory courses. AP CS A is explicitly designed to be like existing introductory computer science courses ("CS1") across the U.S. and Canada. CSP is different, because it is being created from scratch by the College Board, with U.S. National Science Foundation (NSF) funding.

Will CS departments start offering CSP-like courses? Based on Philip Guo's recent analysis of introductory courses at universities (at http://bit.ly/W0vtox), there is not much sign that CSP courses are being created (see the blog post at http://bit.ly/21sTYYt). I am not even sure those who initially offered CSP will keep doing so. My institution was one of the pilot sites that offered CSP at the college level (you may access a list of pilot sites at http://www.csprinciples.org/home/pilot-sites). There are no current plans at Georgia Tech to offer the course ever again. Since it was not a requirement for anyone, few students signed up for the pilot offering. Given the massive enrollment surge, there is little appetite for creating and offering additional classes—especially when no degree programs on our campus require anything like CSP.

Question 2: Will colleges and universities give placement or credit for a course they do not offer?

I do not know how all universities deal with AP credit. At Georgia Tech, we can only give credit for an existing course. An AP course might count as taking some course, or might allow you to skip to a more advanced course. If we do not offer a CSP-like course, we cannot give credit for it.

We do offer a Media Computation introductory course in computing for non-CS majors (http://coweb.cc.gatech.edu/mediaComp-teach.

If a student was admitted to Georgia Tech having passed the AP CSP exam, we might give them credit for the Media Computation. The problem is that our non-majors course has much more programming in it than AP CSP, and there is a pathway from the Media Computation course into other CS courses. If students come in with CSP credit and choose to take more CS courses on that pathway, they will not have the background to succeed.

Question 3: Will high school students take AP CSP if it doesn't count for credit or placement?

Students take AP classes for a variety of reasons. My daughter is a high school senior, and she has been taking AP classes to demonstrate to college admissions officers that she can handle rigorous courses, but she is picking AP classes that she thinks are relevant to her college plans.

Some high school teachers have told me their students choose AP courses in order to decrease their future college costs. High school AP classes are typically far cheaper than college classes. Taking equivalent classes at the high school level buys college credits at a lower cost. If the AP class has no college credit equivalent, it may be less attractive to the students who care about the credit or placement.

Some high school teachers have told me their students choose AP courses in order to decrease their future college costs. Taking equivalent classes at the high school level buys college credits at a lower cost.

Action Item: Come up with an AP CSP Plan

U.S. college and university CS departments need to figure out their plans for how they will handle students who are admitted having passed the AP CS Principles exam. We need to be able to explain how AP CSP will count in our programs. In my institution, some possible options (like creating new classes, or getting other degree programs to offer credit for new classes) take a long lead time.

For students who care whether AP courses count for credit or placement, we should have answers for them soon, as they plan to register for the Fall 2016 school year. We need to be able to tell high school principals and teachers it is worthwhile to offer the course, and tell high school students it is worthwhile to take the course. The time to figure that out is now.

Comments

From the outset, I have liked the content of the Principles course.

But from the outset, I have raised the objection that Principles will be something of an orphan course because it is not and will not be an intro course in the CS major.

There was a time when all of this might have been worked out. Way back at SIGCSE 2011 in Dallas, there was a presentation on the not-yet-finished new version of the CS curriculum guidelines. I argued then that the Powers That Be who were looking at curriculum in the universities ought to be figuring out where Principles might fit in, and the Principles people needed to be working with the curriculum people in order to negotiate a place.

Apparently that did not happen. The two groups seem to have followed independent paths. Did the Principles people get involved in Curricula 2013? Did they then get rebuffed? If so, on what basis?

I do not think it makes sense to say that higher education MUST change just because there is this new course. If change is necessary, then change should be justified based on educational merits that could have been argued years ago. I would like to hear the history of why the content of Principles did not make it into Curricula 2013.

—Duncan Buell

You could have asked question 4: Why do university CS programs demand Calculus and Physics, but not CS at high school?

—Andrew Williams

Duncan, higher education should take this as an opportunity. AP CS Principles is a good course. By giving some kind of credit or placement in higher education, we encourage more schools to offer AP CSP and encourage more students to take AP CSP, which gives us more and more diverse students in higher education. It is a good deal for us.

Andrew, if university CS programs were to demand CS at high school, we would accept very few students. For example, less than 10% of high schools in New York City offer any CS at all (see http://nyti.ms/1NGh8Xe) and less than 10% of high schools nationwide offer AP CS. I ask a different question, Andrew. Why aren't we requiring CS of all undergraduates? It is cheaper and easier to do than changing elementary and high schools, and leads to greater long-term change—see http://cacm.acm.org/blogs/blog-cacm/108448-if-you-want-high-school-cs-require-undergraduate-cs/fulltext.

—Mark Guzdial

Mark Guzdial is a professor in the College of Computing at the Georgia Institute of Technology.

Inviting Young Scientists

HEIDELBERG LAUREATE FORUM

acm Association for Computing Machinery

Meet Great Minds in Computer Science and Mathematics

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

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

The Heidelberg Laureate Forum is **September 18–23, 2016** in Heidelberg, Germany.

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

Who can participate?

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

How to apply:

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

What is the schedule?

Application deadline—**February 3, 2016**.

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

Successful applicants will be notified by **end of March/early April 2016**.

More information available on Heidelberg social media

PHOTOS: ©HLFF / B. Kreutzer (top);
©HLFF / C. Flemming (bottom)

Science | DOI:10.1145/2852231

Chris Edwards

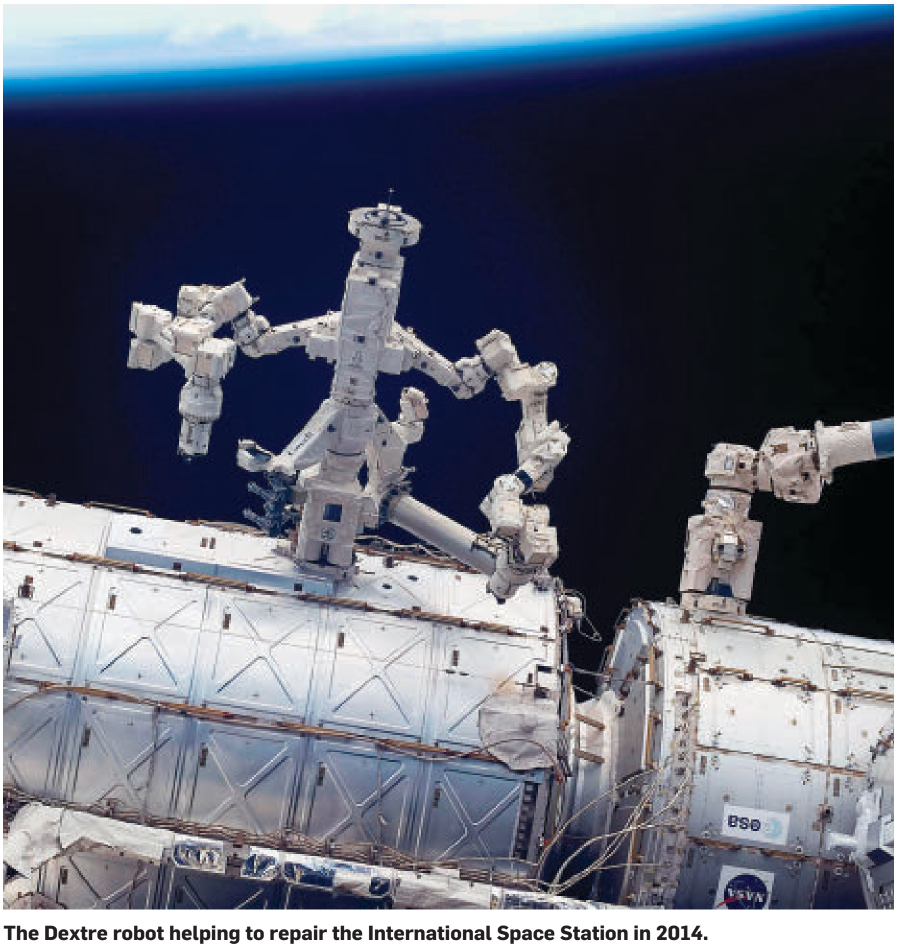
Self-Repair Techniques Point to Robots That Design Themselves

Robots are being taught to brainstorm alternatives when damaged.

WHEN RESEARCHERS AT the Pierre and Marie Curie University (UPMC) in Paris, France, deliberately damaged two of the legs of their hexapod robot, the machine discovered for itself a novel hopping gait that not only overcame its injury, but proved to be faster than its original walking program. Injured another way, the robot found it could move around more easily on its back. The work was part of efforts to make robots that can work around damage and repair themselves when there is no human to help them.

David Johan Christensen, associate professor at the Technical University of Denmark, observes: "In the future, physical self-repair could become critical in applications where no humans are around to assist or repair the robots; for example, in space or underwater applications."

Robotic repairs are already being performed in space, where it is too expensive or dangerous for astronauts to perform the job. In 2014, the Canadian Dextre robot attached to the International Space Station replaced a faulty camera on the arm that normal-



The Dextre robot helping to repair the International Space Station in 2014.

IMAGE COURTESY OF NASA

ly carries it into position to perform repairs of other systems on the orbiting platform. Those repairs were performed under the guidance of human operators on the ground. Uses further afield that may be prone to communications failures, such as undersea cave exploration or the lengthy time delays encountered on deep space missions, make it important to give the robot the ability to repair itself or, if that is not possible, to work around damage.

Jean-Baptiste Mouret, associate researcher at UPMC, says, "You have two time scales in recovery. One is short-term, just like someone going to the hospital for treatment; that is what we are doing with our experiments. By adapting to damage, a robot could limp back to its base. But then something is needed to get it back to full speed. You could have a 'mother robot' that can supply other modules and reassemble the damaged robot in a new way, or maybe it could make new modules to improve the robot once it has gained experience of what is needed."

Alan Winfield, Hewlett-Packard professor of electronics at the University of the West of England, points to the idea of the "evo-factory" in which the mother robot takes account of the damage and, if unable to replace broken components, will come up with alternatives made on the spot using techniques such as three-dimensional (3D) printing.

"Even more exotically, you can certainly imagine a robot that has bits of 3D printing technology incorporated into it. You need not just work around a broken leg, but repair it using some exotic 3D printing organ that you could embed in the robot. People are thinking about this," Winfield says.

Among those people are researchers such as Kyrre Glette and colleagues at the University of Oslo. The superstructures of their robots are made from 3D-printed parts, although they are hand-assembled and combined with off-the-shelf motors so they can move independently. The team has developed algorithms to create the 3D-printed parts from a basic set of shapes and elements to deal with different environments. The aim is to use the experience developed from evolving robot parts to ultimately build a robot that can build additional limbs when needed.

Today, the research into robotic self-repair is in its earliest phase with few successes beyond basic proof-of-concept experiments, such as work on swarms of tiny robots that reconfigure their relationships to each other based on external pressure. "And people have made strange Heath Robinson (unnecessarily complex or implausible) robots with glue guns attached to them. The results can be a bit like toddlers gluing themselves to the floor," says Winfield.

The swarm approach could lead to modular approaches to robotic design in which the machines evolve novel shapes and behaviors when faced not just with injury, but with problems for which they were never designed. A modular robot that walks across the ruins of a building in the wake of an earthquake may find it needs to form itself into a snake-like shape to crawl into a gap in the structure to locate buried survivors.

Says Winfield, "I'm very attracted by the modular and cellular approach to robotics, but we still can't build reliable-enough and flexible-enough modules. You need cells that have a degree of autonomy and which are also capable of self-assembly."

One of the challenges in developing robots that are able to adapt to damage is the amount of time it takes to come up with a solution. After an injury, the robot has to work out what to do next with its remaining limbs and motors.

The swarm approach could lead to modular approaches to robotic design, allowing the machines to evolve novel shapes or behaviors in response to injury or unanticipated obstacles.

Researchers have largely turned to evolutionary algorithms that progressively refine movements using random changes, discarding those that do not work and optimizing those that show promise. A big problem with traditional approaches to evolutionary design is the need for the algorithm inside the robot to start from scratch before coming up with a theoretically viable solution. "It would search for 20 minutes or so and then try something and then find it doesn't work," says Mouret.

Using the basic evolutionary approach, a robot could be inactive for hours after an injury. A robot in a dangerous situation, such as negotiating a sudden landslide, will not be able to afford those long delays.

Mouret says, "The key thing is that searching these huge spaces of different types of motion from the beginning takes time, so we thought we should start with some previous knowledge. Then we realized that this approach makes sense because it's the way that most humans and animals work; they use their previous experience. That's what we thought we should do with our robots."

The solution developed by the UPMC team, together with Jeff Clune from the University of Wyoming at Laramie, was to arm the robot with basic knowledge of the types of movement it could adopt and use those as "seeds" to explore different types of locomotion after an arbitrary part was damaged. They built a six-dimensional map of different types of movement based on extensive simulations performed by a virtual robot, scoring them on expected speed.

"We store about 13,000 different gaits," says Mouret. "Each gait uses 36 parameters, with each one of those parameters needing about 4 bytes of data. That's very small compared to the amount of storage we have in a device such as a cellphone. I don't think we will be limited by space with this approach."

When injured, the robot picks from the map, favoring those types of motion that scored well from the simulations and which are still available to it. Often the robot finds that, with its injury, the high-scoring simulations do not perform well under real-world conditions, so the algorithm reduces the score of the approach it tried and those in its vicinity on the map before selecting another at random. Over time the map changes, and the robot focuses attention on techniques that show some level of success, using self-learning algorithms and trial and error to optimize its motion.

Simulation-based techniques are also being developed that will help robots deal better with obstacles and potentially avoid damage in situations where they push themselves too far. Sehoon Ha and Karen Liu of the Georgia Institute of Technology created an algorithm to help humanoid robots fall in ways that minimize damage by planning trajectories such as rolls that attempt to turn one sharp impact into a series of smaller, less-damaging contacts with the ground. However, in its current form, the algorithm is too slow to help a physical robot decide which falling strategy to use in real time. Planning can take up to 10 seconds.

As well as processing time, a key concern among robot designers is the effectiveness of algorithms tested primarily in the virtual world. Simulation-based techniques have their limits and need to be augmented by physical 'embodied' experimentation, says Christensen. "In my opinion, the models on which simulations are based can never truly capture the complexity of the interactions between the robot and its environment. Therefore simulations can be used as a starting point, to bootstrap and speed up the adaptation, but are too limited to fully replace embodied experimentation."

A form of embodied experimentation was demonstrated by the UPMC and Wyoming teams: an unusually slippery floor in the French lab, polished for a visit by politicians, provided the opportunity for the robot built by Mouret and colleagues to find it had a problem walking and needed to evolve a new set of movements that could cope with the unexpected surface. The experience of the robot flipping over on its back in another run of the experiment caused the team to look at a new design. The experience of adaptation can, as a result, inform future designs.

"It's inspired our next robot. We have a new robot that's a bit bigger and which has feet on both sides," Mouret says.

A key concern among robot designers is the effectiveness of algorithms tested primarily in the virtual world.

To come up with more robust solutions, Winfield argues robotics needs to emulate biological evolution more closely. Today, evolutionary algorithms work by progressively altering a single robot design in simulation. Biological evolution works on populations of organisms in the real world. "We need populations of robots to think of ideas, then try them out." C

Further Reading

Cully, A., Clune, J., Tarapore, D., and Mouret, J-B.
Robots That Can Adapt Like Animals, *Nature*, Vol. 521, p503 (2015)

Christensen, D.J., Larsen, J.C., and Stoy, K.
Fault-Tolerant Gait Learning and Morphology Optimization of a Polymorphic *Walking Robot, Evolving Systems 03/2013;* 5(1)

Glette, K., Johnsen, A.L., and Samuelsen, E.
Filling the Reality Gap: Using Obstacles to Promote Robust Gaits in Evolutionary Robotics, *Proceedings of the 2014 IEEE International Conference on Evolvable Systems (ICES)*, pp181-186, (2014)

Ha, S., and Liu, C.K.
Multiple Contact Planning for Minimizing Damage of Humanoid Falls, *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2015)

Videos

UPMC adapting robot
https://www.youtube.com/watch?v=T-c17RKh3uE

Humanoid Robot Fall Planning:
https://www.youtube.com/watch?v=gvKRlgc9pJI

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

Curabit Semper

ACM Member News

SECURITY, PRIVACY ARE NOT EASY: CRANOR

Lorrie Faith Cranor, a professor in Carnegie Mellon University (CMU)'s School of Computer Science and Engineering, almost bypassed computer science (CS) in favor of journalism.

Now director of CMU's Usable Privacy and Security Laboratory, Cranor has authored over 100 research papers on online privacy, phishing and semantic attacks, spam, electronic voting, anonymous publishing, and usable access control. "Privacy is a value I believe in, but it's not easy because it interferes with things people want," Cranor says.

Her father, who worked with computers at Walter Reed National Military Medical Center in Bethesda, MD, taught her to program early, but her aptitude for CS was tempered by an aversion to "computer science geeks with poor social skills." That is why Cranor worked for her school newspaper and planned a journalism career, but when she was not chosen as the paper's editor-in-chief, she changed course.

She earned a bachelor of science degree in Engineering and Public Policy, master's degrees in Technology and Human Affairs, a master of science degree in CS, and a doctorate in Engineering and Policy, from Washington University in St. Louis. After graduation she worked in AT&T's research lab, joined an industry consortium for Web privacy standards, and developed the "Privacy Bird" interface tool.

At CMU, Cranor founded the Symposium on Usable Privacy and Security (SOUPS). She also co-directs the privacy engineering master's program at CMU, the only program of its kind. "A lot of password security is based on mythology; having longer passwords is better than complex passwords."

—*Laura Didio*

Technology | DOI:10.1145/2852235

Logan Kugler

How a Supervillain (or a Hacker in His Basement) Could Destroy the Internet

Network experts share their greatest fears about attacks and accidents that could destroy the Internet.

U.S. SENATOR TED STEVENS (R-AK) sealed his legacy in 2006 when he infamously referred to the Internet as a "series of tubes." This was a simplistic description in the extreme for the varied and intricate architecture that comprises the Internet's physical and virtual infrastructure. While it might be excused as a gaffe from an older generation, Stevens was partially responsible for regulating the Internet at the time he made the remarks.

Unfortunately, Stevens, who passed away in 2010, was not alone in his misunderstanding of how the Internet works, and ignorance in this area—far from being bliss—is dangerous. The series of tubes and other machinery that make the Internet possible can and have been disrupted. They can even be destroyed, say experts.

One such expert is Samy Kamkar, and he knows all about Internet security. After he hacked the social network MySpace in 2005 with the Samy worm and was caught and convicted of a felony, he agreed to a plea deal with the U.S. Secret Service that banned him from even touching a computer for three years. The virus Kamkar created spread so alarmingly fast that "the entire Internet was freaking out about [it]," reported media site Fusion.

People like Kamkar do not idly think about ways the Internet could be disrupted or completely destroyed; they get paid for it. Today Kamkar is a white-hat hacker, helping companies patch security flaws before they can be exploited. Following are some of the biggest threats Kamkar and people like Thomas Savundra, cofounder of ultra-secure cloud storage service Sync, see to Internet infrastructure—and what we might be able to do to help prevent them.

Samy Kamkar, who hacked the MySpace social network with the Samy worm in 2005, is today a white-hat hacker, helping companies catch security flaws before they can be exploited.

PHOTO BY JOCK FISTICK

Cutting the Cord

Software crashes and cybersecurity threats dominate the headlines, but Kamkar says there is another way to attack the Internet that could do more damage on a wider scale.

"A physical attack that is starting to affect small areas (but significant numbers of users) is criminals physically cutting fiber optic cables," Kamkar says. "Typically just knowing where to go and what to do is enough to cause major disruptions."

This may seem like a mundane and unsexy doomsday scenario, but it is one authorities are beginning to take seriously. *The Washington Post* reported in July 2015 that the U.S. Federal Bureau of Investigation (FBI) was investigating a series of attacks on fiber optic cables in California that disrupted Internet service in parts of San Francisco and Sacramento. These attacks probably were not the work of petty vandals or common criminal elements; the FBI believes the attacks required expertise, as the perpetrator(s) broke into underground bunkers that housed those cables. This was the eleventh attack within 12 months in this particular case; a similar attack took place in Arizona early last year.

In each case, an attack on fiber optic cables caused local Internet disruptions. Consider what might happen if the cables tampered with served a larger area.

That has happened several times over the last decade to the fiber optic cables under the Mediterranean Sea, when cables delivering Internet service to entire Middle Eastern and Asian countries have been the victim of a variety of attacks. Some are known to have been accidental, such as when a ship's anchor cuts through submarine cables; others, however, are potentially criminal. All have worldwide implications. One such attack in 2008 stopped Internet service in Egypt, Pakistan, Kuwait, and India—four countries together inhabited by nearly 1.5 billion people or one-fifth of our planet's population, according to a report from *Wired*.

Destroying undersea infrastructure may require a specific set of skills, but it does not require an army. One of the cable-cutting incidents in the Mediterranean was the work of just three men. The relative ease with which malicious parties can disrupt Internet service on a global scale does not end there. While undersea cables are marginally thicker and sturdier than their landborne counterparts, they still are shockingly easy to access.

Andrew Blum, author of *Tubes: A Journey to the Center of the Internet*, told *Wired*, "Other than obscurity and a few feet of sand, [the cables] are just there. The staff at a cable landing station might patrol the path to the beach landing once or twice a day, but otherwise I've never heard of or seen any constant security."

Attacks on enough poorly defended cables at the same time could bring the Internet crashing to its knees.

When the Lights Go Down in the City

Another area malicious parties could attack to undermine the Internet is the electrical grid, says Kamkar. "This could happen by attacking the industrial systems that control the electrical grid, such as SCADA systems," he says.

In a complete disruption of the Internet, stock market activity would cease, online transactions would grind to a halt, and work at most modern businesses would be impossible.

SCADA (Supervisory Control And Data Acquisition) industrial control systems govern major functions in everything from factories to refineries to power generation stations. Someone who wanted to take down the Internet could infiltrate and destroy these systems, wreaking havoc on the machinery that keeps power plants running and leaving their service areas without power and Internet.

Obviously, a power grid disruption causes huge problems beyond taking down the Internet. A generator or alternate power source can bring homes and businesses back online after some initial turmoil in the event of a grid failure. The downtime is costly, to be sure, but in a complete disruption of the Internet, for example, stock market activity would largely cease, online banking, commerce, and transactions would grind to a halt, and work at most modern businesses would be impossible.

On a wider scale, targeting facilities that service key Internet infrastructure could destroy huge swaths of the Net. "We have to consider the underlying tubes and fiber, the key network exchange hubs and datacenters hosting Internet content," says Thomas Savundra, cofounder of ultra-secure cloud storage service Sync. A disruption to the power supplied to any of these facilities could knock out a key piece of the machine that keeps the Internet running.

Disruption is not limited to instances in which a facility's power source fails; its machinery is also a target.

In 2010, the world was introduced to Stuxnet, a computer virus sometimes called the first digital weapon. It is believed the worm was created by U.S. and Israeli intelligence services, because it targeted Iranian centrifuges used to enrich uranium. Stuxnet burrowed its way into the centrifuges after unwittingly being introduced to the machinery via a corrupted USB drive. Once present, Stuxnet damaged the centrifuges from the inside. The highly guarded centrifuges were air-gapped (physically isolated) from the Internet, which is why the virus had to be introduced via USB. That may be a rarity, however.

"More likely in the future it would be an attack that happens online, as more industrial systems are further connected to the Internet," says Kamkar.

According to a report by the World Economic Forum, the number of Internet-enabled sensors in use increased more than five times between 2012 and 2014—from 4.2 billion shipped to 23.6 billion. That is not just for smart homes and snazzy gadgets; governments and huge corporations are bringing key infrastructure and plants online, too. Stuxnet attacked machinery's actuators, which are under threat in any factory where the machinery is directly connected to the Internet. As offline machines are augmented with online sensors, those sensors may enable damage indirectly when viruses cause problems with them (such as replacing real data with fake data).

That means a future saboteur would not need to be physically present to do serious damage; he or she or they could introduce a Stuxnet-like worm into one or more facilities using the machinery's connection to the Internet, while working from anywhere in the world. This next generation of cyber-terrorist could even shut down parts of the Internet by attacking the power sources of key Internet infrastructure.

While the Internet of Things might deliver untold economic and productivity benefits, companies and governments need to prevent such attacks before they occur by bringing industrial infrastructure online intelligently—no matter how fast the technology deploys or how much pressure they face from citizens and shareholders.

"Connecting critical infrastructure to the Internet shouldn't be done without proper security measures," Kamkar says.

Saving the Internet from Itself

There is one other less tangible, but no less important, piece of infrastructure that could—if damaged heavily enough—destroy the Internet as we know it, says Savundra. "The threats are less likely to be centered on physical infrastructure, but rather software-layer trust and confidence," he says. "I think privacy and security are going to be the major issues. Individuals and companies are already beginning to change their behavior because of widespread hacks, unauthorized surveillance, and the erosion of online privacy."

Such nefarious activities are starting to break some of the Internet's most deeply cherished tenets, namely the ability to surf safely and anonymously, whether for business or for pleasure.

That could be the harbinger of a future Internet that has been hollowed out by hacks and spying. Imagine a situation in which consumers could not trust payment details to companies because of threats to online databases. Think how the Internet's value would diminish if people could not reasonably assume their identity would not be stolen the moment they exchange basic information with another party online. What if the free speech that Internet communities enable became leverage used by governments against their citizens? All of these issues could hamstring the Internet's effectiveness as the "network of networks."

What if the free speech that Internet communities enable became leverage used by governments against their citizens?

"I think if the erosion of privacy becomes widespread," Savundra says, "people will start using Internet services differently."

Savundra sees regular and healthy discourse about privacy, hacking, and safety issues online as a major solution to the problem. "The Internet is designed by humans for humans. There are plenty of smart people who will defend their rights, either via policy or technology, when their freedoms are jeopardized," he says.

Kamkar, perhaps as a cynical souvenir from his time in the shadows as a hacker, has different advice to address serious threats to the Internet's basic infrastructure: "The biggest way to prevent these issues is to understand that most of these systems are insecure by default."

Further Reading

Chang, Alexandra
Why Undersea Internet Cables Are More Vulnerable Than You Think. *Wired*, http://www.wired.com/2013/04/how-vulnerable-are-undersea-internet-cables

Hill, Kashmir
10 years after his epic MySpace hack, Samy Kamkar is trying to turn hackers into heroes. *Fusion*, 2015; http://fusion.net/story/180919/samy-kamkar-is-a-white-hat-hacking-hero

Greenberg, Will
String of West Coast attacks on Internet fiber optic cables leads to FBI investigation. *The Washington Post*, 2015; http://http://wapo.st/1TStnhI

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

Milestones

Computer Science Awards, Appointments

COMPUTER SCIENTISTS AMONG NEWEST ROYAL SOCIETY FELLOWS

Two computer scientists were among 47 new Fellows and 10 new Foreign Members announced by the Royal Society.

Clifford Cocks, chief mathematician at U.K. security and intelligence organization GCHQ, was named a Royal Society Fellow for his work in cryptography. Cocks was first to devise a practical implementation of public key cryptography, as well as a practical scheme for identity-based public key encryption.

Zoubin Ghahramani, a professor of information engineering in the Department of Engineering of the University of Cambridge, was named a Fellow for his status as "a world leader in the field of machine learning, significantly advancing the state of the art in algorithms that can learn from data." Ghahramani is known for fundamental contributions to probabilistic modeling and Bayesian nonparametric approaches to machine learning systems, and to the development of approximate variational inference algorithms for scalable learning. He is a pioneer of semi-supervised learning methods, active learning algorithms, and sparse Gaussian processes.

The Royal Society's fundamental purpose is to recognize, promote, and support excellence in science and to encourage the development and use of science for the benefit of humanity.

MARTONOSI RECEIVES WOMEN IN EDA ACHIEVEMENT AWARD

Margaret R. Martonosi, Hugh Trumbull Adams '35 Professor of Computer Science at Princeton University, has received the Marie R. Pistilli Women in Electronic Design Automation (EDA) Achievement Award for 2015 for her technical leadership of high-impact research projects in the design of power-efficient computer architectures and sensor systems, as well as her creation and organization of career development programs for women and minorities.

Martonosi holds a bachelor's degree from Cornell University and master's and Ph.D. degrees from Stanford University, all in Electrical Engineering. Since 1994 she has been on Princeton's faculty, where she holds an affiliated faculty appointment in Princeton Electrical Engineering.

Donatella Sciuto of Politecnico di Milano and chairperson of Women in Electronic Design called Martonosi "a force to be reckoned with," adding, "we are honored to present her with the Marie Pistilli award in recognition of her notable contributions to research and technology and the impact she has made on career development programs for women and minorities."

Society | DOI:10.1145/2852233

Tom Geller

In Privacy Law, It's the U.S. vs. the World

Snowden revelations force changes, but Facebook (and others) resist.

facebook class action
Sign In
JOIN NOW
Join the Facebook class action!
Enter your country of residence.
Note: For legal reasons US and Canadian residents cannot participate.
Choose country of residence...
Are you an adult?
Note: For legal reasons underage persons cannot participate.
○ No ○ Yes
Problems?

An app that allows users to participate in the class action lawsuit against Facebook in Europe, to try to enforce the right to data protection.

IMAGE © 2011 EUROPE-V-FACEBOOK.ORG

Google is forced to wipe a Spanish citizen's past financial troubles from its records. The Belgian Privacy Commission tells Facebook it must "bend or break" to abide by the country's privacy laws. A plaintiff presses privacy cases against Facebook in both Austrian and European courts.

In each instance, national privacy laws collide with the international nature of the Internet, and with American business expectations. Cross-border issues of the online world are not new, of course: the European Data Protection Directive at the center of many such cases was enacted in 1995, and online jurisdiction cases go back at least as far (the 1996 *U.S. vs. Thomas* decision affirmed a California bulletin board system (BBS) operator must obey "community standards" for Tennessee subscribers). Yet few cases have cited the European law to prosecute U.S. companies.

Then came Edward Snowden. His 2013 exposé of spying practices revealed the U.S. was secretly collecting protected European data, often via U.S. companies like Facebook. The global community upped the ante in response, with new laws proposed or enacted in countries as diverse as Madagascar, Thailand, and Chile. In Europe, individual countries have used the 1995 law to challenge American practices, and the European Commission plans changes to the law it claims will "strengthen online privacy rights and boost Europe's digital economy."

"What has changed over the past few years is that this issue has become much more political" said Omer Tene, vice president of Research and Education for the International Association of Privacy Professionals. "European politicians started presenting it as a competitive difference between U.S. and European industry. In their presentation of the world, if you store or provide your data to U.S. companies, U.S. national security authorities will put their hands on it, whereas if you work with Europeans, they won't." However, he said, "that's far from the reality, because European national security authorities have powers that are not wholly different. In many cases they have laws that are even stronger, and often with less transparency."

The U.S. as a Special Case in Europe

Data transfer is a major issue at the core of the dispute. The European Data Protection Directive says, among other things, that personal data can only be transferred in non-restricted form to 10 countries outside the European Union (EU) whose laws provide "adequate protection." The list currently includes Argentina, Switzerland, Israel, New Zealand, and Uruguay, as well as five tiny countries each with populations under 100,000 (Andorra, Guernsey, the Isle of Man, the Faroe Islands, and Jersey). Canada and Australia are additionally certified, with limitations.

The U.S. uniquely benefitted from a "safe harbor" provision that allowed domestic companies to self-certify that they comply with certain principles relating to: notice (when personal information is collected); the choice to opt out of such collection, and access to data collected. The safe harbor law further required such data to be kept secure; to be used only for a specified purpose; and to be kept from being recklessly transferred to third parties. Finally, U.S. companies in the safe harbor program had to implement ways for Europeans to enforce their rights under the provision.

That exception was struck down by the Court of Justice of the European Union (CJEU) as a result of a two-year-long case filed by Max Schrems, a doctoral law student in Austria who prosecuted Facebook in Austrian and European courts. In the lead-up to the Court's decision, Advocate General Yves Bot cited Snowden, as "the law and practices of the United States offer no real protection against surveillance by the United States of the data transferred to that country. … The surveillance carried out by the United States intelligence services is mass, indiscriminate surveillance." Schrems also linked the CJEU's decision to U.S. spying, writing that "The judgment makes it clear that U.S. businesses cannot simply aid U.S. espionage efforts in violation of European fundamental rights."

Overlapping Jurisdictions

Just as "states' rights" arguments complicate American law, relationships between EU statutes and those of member countries complicate Continental law. Donald Aplin, who covers these issues as managing editor of *Privacy & Data Security Law Report* at Bloomberg BNA, explained one important distinction: "European Directives like the current Data Protection Directive are things where the European Commission says, 'here, everybody should follow this in the EU.' Then each independent member state has to adopt that Directive into its own national laws. A regulation, on the other hand, is something that is law for the whole EU from the second it's passed." Since 2012, the European Commission has been planning to replace the Data Protection Directive with a new General Data Protection Regulation. With passage possible within the next year, Aplin believes "That regulation will fundamentally change a lot of how the EU enforces privacy."

Individual European countries have their own privacy laws, some of which precede the Directive: The Belgian Privacy Commission referred to a domestic 1992 law when it slammed Facebook in a May 2015 report. Among other things, that 28-page Recommendation particularly called out the company's practice of broadly tracking users throughout the Internet, even those who have deactivated their accounts or opted out of receiving targeted ads. In response, Facebook claimed it is bound only by the national data protection laws of Ireland, where its European operations are based, for all its users in Europe. The Belgian Court of First Instance ruled against Facebook, fining the company 250,000 euros per day that it continues to track non-members. Facebook has said it will appeal the ruling.

Belgian Privacy Commission President Willem Debeuckelaere challenged Facebook's assertion on two points. "First, Facebook has five million members in Belgium," he said. "It's a small country, with only 12 million inhabitants, so that's around 40% of our population. The Belgian data protection authority is in charge of this particular question, just as the Federal Trade Commission is in the U.S. Second, European legislation states a country can use its proper powers if there is an establishment of the company or organization in your country. That's the case with Facebook Belgium, which is one part of the whole Facebook structure. There are only four or five people working there, but no matter; they're here! This nexus is enough, and is the legal basis for us to go to Belgian courts."

The Belgian Court of First Instance ruled against Facebook, fining the company 250,000 euros per day that it continues to track non-members. Facebook has said it will appeal the ruling.

Repercussions Beyond Europe

Europe has been the most active region for privacy law so far, but governments around the world have been busy creating and enacting laws, even if enforcement often follows only years later. For example, news items posted to Aplin's publication in a recent one-week period reflect developments in Macau, Hong Kong, South Korea, Germany, and the U.K.

Global law firm DLA Piper's interactive "Data Protection Laws of the World" site labels data protection regulation and enforcement as "moderate" or stronger in the world's most far-flung countries. In particular, data protection in South Korea and Canada stand out as "heavy," while Australia, New Zealand, Argentina, Japan, and Morocco appear as "robust," along with the U.S.

Asia and Latin America are a mixed bag; China is still a big unknown, although the country's January 2014 release of "Measures for the Administration of Online Transactions" may indicate future directions.

Africa has remained comparatively inactive, although that may be changing. The 54 members of the African Union introduced a Convention on Cybersecurity and Data Protection in June 2014, although ratification appears to have stalled. (As of June 2015, however, 18 African nations have "comprehensive privacy laws regulating the collection and use of personal information by the private sector," according to Bloomberg BNA.)

Aplin believes the delay may be a matter of priorities. "There's a lot going on in the EU because Europeans have the luxury of thinking about things like privacy, which is a classic First-World issue," he said. "All the big African countries recognize they probably should be dealing with this issue, but they're also worried about getting the roads and phones to work, and labor law, and environmental issues."

Regardless of economy, former colonies tend to have laws that reflect their heritage, said Aplin. "When you look at Africa, you can see which ones were the French colonies. Côte D'Ivoire is a good example; their data protection is very much based on what the French do."

Getting Legal

So what can a company like Facebook do? According to Jim Halpert, partner and chair of the U.S. Data Protection and Privacy Group at DLA Piper, "Being in compliance all over is a fairly Herculean task, given the complexity of requirements around the world. If you're exposed in different countries, you may pick one high-water mark country like Germany and establish German requirements; that will stand you in pretty good stead. There will be other formal filing requirements in countries like South Korea and the United States, so there's still some localization needed."

For Debeuckelaere, the ability to prosecute Facebook in Belgium is a matter of basic rights. "If Facebook's Belgian company should disappear, we could transfer the question to Irish or Dutch or German authorities. Even now, we could go to the Irish courts. But why should we do that?

"The European Convention on Human Rights gives every citizen the possibility to ask a national judge to enforce fundamental rights. One of these is the right to privacy. So why should we go to Ireland or the Netherlands or Germany or California when we can do it here? It's cheaper, it's easier, and it's in our own languages."

> **"There's a lot going on in the EU because Europeans have the luxury of thinking about things like privacy, which is a classic First-World issue."**

Further Reading

Rich, C.:
"Privacy Laws in Africa and the Middle East" (et sim.), *Bloomberg BNA Privacy and Security Law Report*, http://www.mofo.com/people/r/rich-cynthia-j?tabs=publications

Belgian Privacy Commission, "On 13 May the Belgian Privacy Commission adopted a first recommendation of principle on Facebook", http://www.privacycommission.be/en/news/13-may-belgian-privacy-commission-adopted-first-recommendation-principle-facebook (Unofficial English translation, 2015)

Belgian Official Journal, "Act of 8 December 1992 on the protection of privacy in relation to the processing of personal data," http://www.privacycommission.be/sites/privacycommission/files/documents/Privacy_Act_1992.pdf (Unofficial English translation, 2014)

European Commission "Protection of Personal Data" website, http://ec.europa.eu/justice/data-protection/index_en.htm (English version)

European Commission, "Reform of the data protection legal framework in the EU," http://ec.europa.eu/justice/data-protection/reform/index_en.htm

European Commission, "Factsheet on the 'Right to be Forgotten' Ruling (C-131/12)", 3 June 2014, http://ec.europa.eu/justice/newsroom/data-protection/news/140602_en.htm.

U.S. Government (multiple agencies), "Welcome to the U.S.-EU & U.S.-Swiss Safe Harbor Frameworks," http://www.export.gov/safeharbor/

Schrems, Max
"Europe versus Facebook,"
http://europe-v-facebook.org

DLA Piper,
"Data Protection Laws of the World,"
http://dlapiperdataprotection.com

Tom Geller is an Oberlin, OH-based technology and business writer.

Appointment

Forbes Takes Chair of ACM Education Policy Committee

ACM recently named Jeffrey R.N. Forbes, an associate professor of the Practice of Computer Science at Duke University, as Chair of the ACM Education Policy Committee.

Created in 2007, the ACM Education Policy Committee is a high-level committee of computer scientists and educators dedicated to improving opportunities for quality education in computer science and computing education around the world.

Forbes succeeds Robert B. (Bobby) Schnabel, founding chair of the ACM Education Policy Committee, who assumed the position of CEO of ACM in November. In leading the committee, Forbes will play a major role in developing initiatives aimed at shaping education policies that impact the computing field.

"Not only is computer science education one of Jeff's core research interests, but he has extensive experience directing educational programs in this area," said Alexander L. Wolf, president of ACM. "... Jeff Forbes is the perfect person to ensure that ACM will continue to be a leading voice in promoting and shaping computer science education at all levels."

Forbes, who also is an associate dean of the Trinity College of Arts and Sciences at Duke University, served as program director for the Education and Workforce program in the National Science Foundation's Directorate for Computer and Information Science and Engineering. His research interests include computer science education, social information processing, and learning analytics.

Said Forbes, "Ensuring expanded access to quality computer science and computing education is key to every country's future. As the world's leading computing society, ACM offers indispensable expertise, insight, and guidance on computing education and workforce development. We look forward to working with policy leaders, and all stakeholders, to improve inclusive access to high-quality computer science education."

Milestones | DOI:10.1145/2856228

ACM Inducts Fellows

ACM has recognized 42 of its members for significant contributions to the development and application of computing, in areas ranging from data management and spoken-language processing to robotics and cryptography. The achievements of the 2015 ACM Fellows are fueling advances in computing that are driving the growth of the global digital economy.

"Whether they work in leading universities, corporations, or research laboratories, these newly minted ACM Fellows are responsible for the breakthroughs and industrial innovations that are transforming society at every level," said ACM President Alexander L. Wolf. "At times, the contributions of a Fellow may include enhancements to a device that immediately impacts our daily lives. At other times, new research discoveries lead to theoretical advances that, while perhaps not immediately perceptible, have substantial long-term impacts."

The 2015 ACM Fellows have been cited for contributions to key computing fields including software research, data mining, computer graphics, computer and mobile systems, system security, multiprocessor and memory architecture design, and research in sensor networks.

ACM will formally recognize the 2015 Fellows at the annual Awards Banquet, to be held in San Francisco in June. Additional information about the 2015 ACM Fellows, the awards event, as well as previous ACM Fellows and award winners is available on the ACM Awards site at http://fellows.acm.org.

2015 ACM Fellows

Anastasia Ailamaki
EPFL

Nancy M. Amato
Texas A&M University

David M. Blei
Columbia University

Naehyuck Chang
KAIST

Hsinchun Chen
University of Arizona

Mary Czerwinski
Microsoft Research

Giuseppe De Giacomo
Universita' di Roma "La Sapienza"

Paul Dourish
University of California, Irvine

Cynthia Dwork
Microsoft Research

Kevin Fall
Carnegie Mellon University

Babak Falsafi
EPFL

Michael Franz
University of California, Irvine

Orna Grumberg
Technion

Ramanathan Guha
Google, Inc.

Jayant R. Haritsa
Indian Institute of Science, Bangalore

Julia Hirschberg
Columbia University

Piotr Indyk
Massachusetts Institute of Technology

Tei-Wei Kuo
Research Center for
Information Technology Innovation,
Academia Sinica

Xavier Leroy
INRIA

Chih-Jen Lin
National Taiwan University

Bing Liu
University of Chicago

Yunhao Liu
Tsinghua University

Michael George Luby
Qualcomm Inc.

Michael Rung-Tsong Lyu
The Chinese University
of Hong Kong

Ueli Maurer
ETH Zurich

Patrick McDaniel
Penn State University

Victor Miller
IDA Center for Communications
Research

Elizabeth D. Mynatt
Georgia Institute of Technology

Judea Pearl
UCLA

Jian Pei
Simon Fraser University

Frank Pfenning
Carnegie Mellon University

Dragomir R. Radev
University of Michigan

Sriram Rajamani
Microsoft Research, India

Pablo Rodriguez
Telefonica

Mooly Sagiv
Tel Aviv University

Peter Schröder
California Institute of Technology

Assaf Schuster
Technion

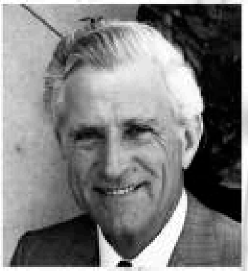
Kevin Skadron
University of Virginia

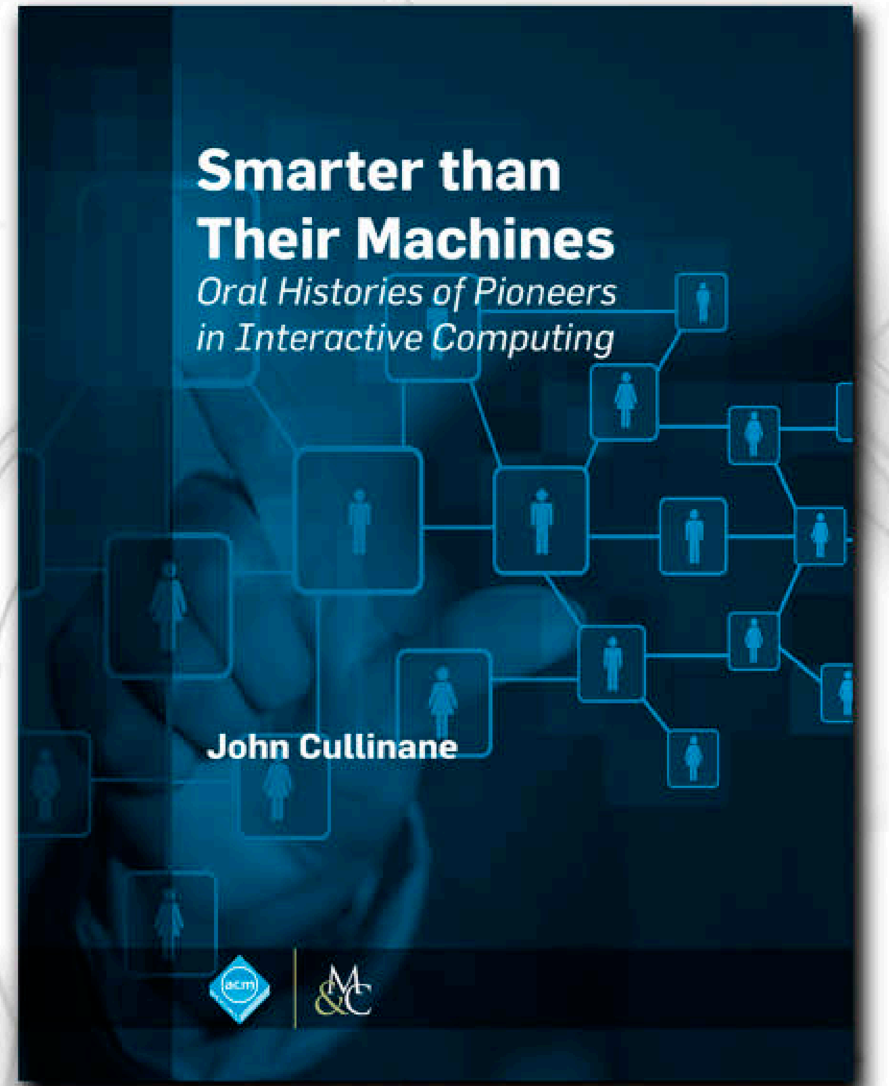
Wang-Chiew Tan
University of California Santa Cruz

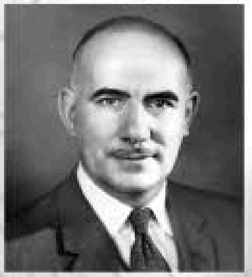
Santosh Vempala
Georgia Institute of Technology

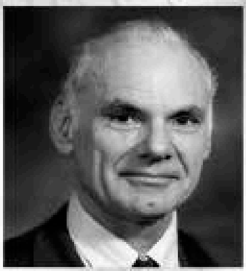
Tandy Warnow
University of Illinois
at Urbana-Champaign

Michael Wooldridge
University of Oxford




Smarter than
Their Machines
Oral Histories of Pioneers
in Interactive Computing
John Cullinane
acm
M&C





V viewpoints

DOI:10.1145/2856447

Peter C. Evans and Rahul C. Basole

Economic and Business Dimensions

Revealing the API Ecosystem and Enterprise Strategy via Visual Analytics

Seeking better understanding of digital transformation.

Valuable insights can be gained by applying visual analytic techniques to understand complex, emerging ecosystem dynamics and evolving enterprise strategies.[1,2] One such context is the application programming interface (API) ecosystem. APIs have grown dramatically in the past five years. These bits of code act as digital control points that set the terms for which data and services can be efficiently shared or "called" over the Internet.[11] There are now over 12,000 open APIs available across a wide range of market sectors, a thirtyfold increase since 2006.[a] While the ability to connect to digital resources using APIs has been a feature of computing for decades, the prominence of digital platforms, the rise of mobile computing, lower cost

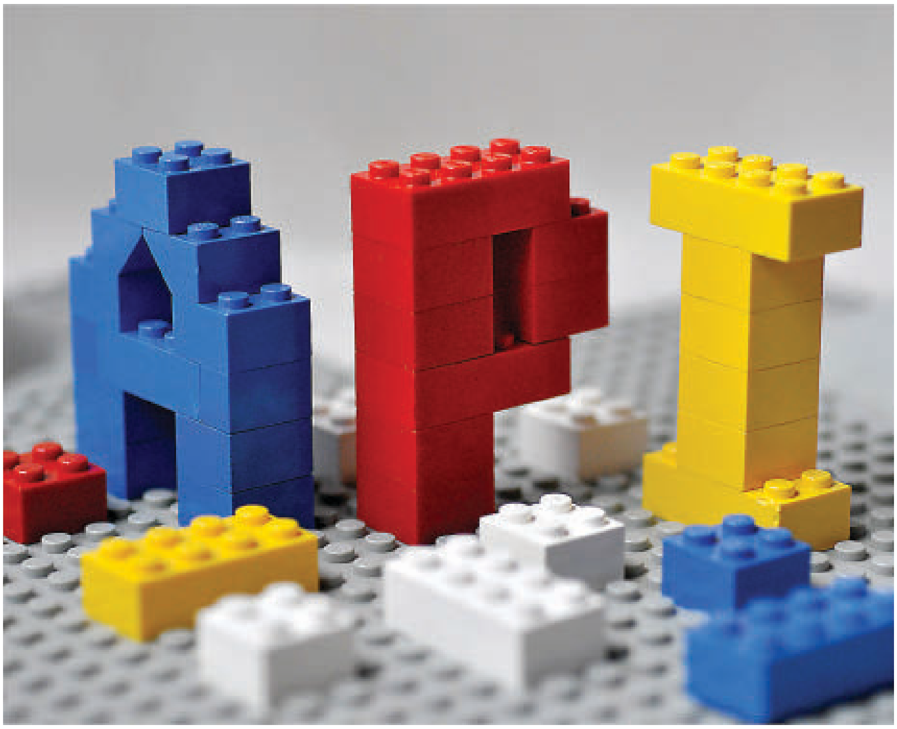
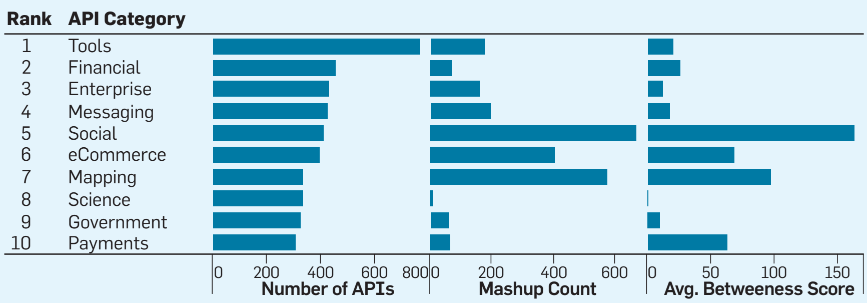


IMAGE BY ALICIA KUBISTA/ANDRIJ BORYS ASSOCIATES

[a] See ProgrammableWeb, http://www.programmableweb.com/

Figure 1. Open APIs by top 10 sectors.



Source: Authors calculations with data from ProgrammableWeb, 2015

of data storage, and just the sheer usefulness of automating how digitally encoded information can be made available and exchanged has helped to spur faster growth. Some of the most popular APIs handle a staggering number of calls. For example, Twitter, Google, Facebook, Netflix, AccuWeather, eBay, and Sabre all handle over a billion API calls per day.[4]

Firms are finding APIs to be beneficial in a variety of ways. The initial provider can use an API as a way to create new revenue streams, by offering access to already existing digital information through a range of different business models (including subscription, license, freemium, or pay-as-you-go). For example, Thomson Reuters, which has an extensive 45 million over-the-counter exchange-traded instruments globally, is building out a set of APIs to make it easier for other companies to access this data.[3] APIs also make it possible for third parties to build entirely new digital applications and services by creating "mashups" that integrate existing APIs. More than 6,000 such mashups have been created in recent years.

While open APIs promise to create value, boost productivity, and offer strategic advantages for firms that embrace their use, they are not deployed evenly across firms or industry sectors. What sectors have attracted the most APIs? And what firms are situated at the core of the API ecosystem and which remain at the periphery?

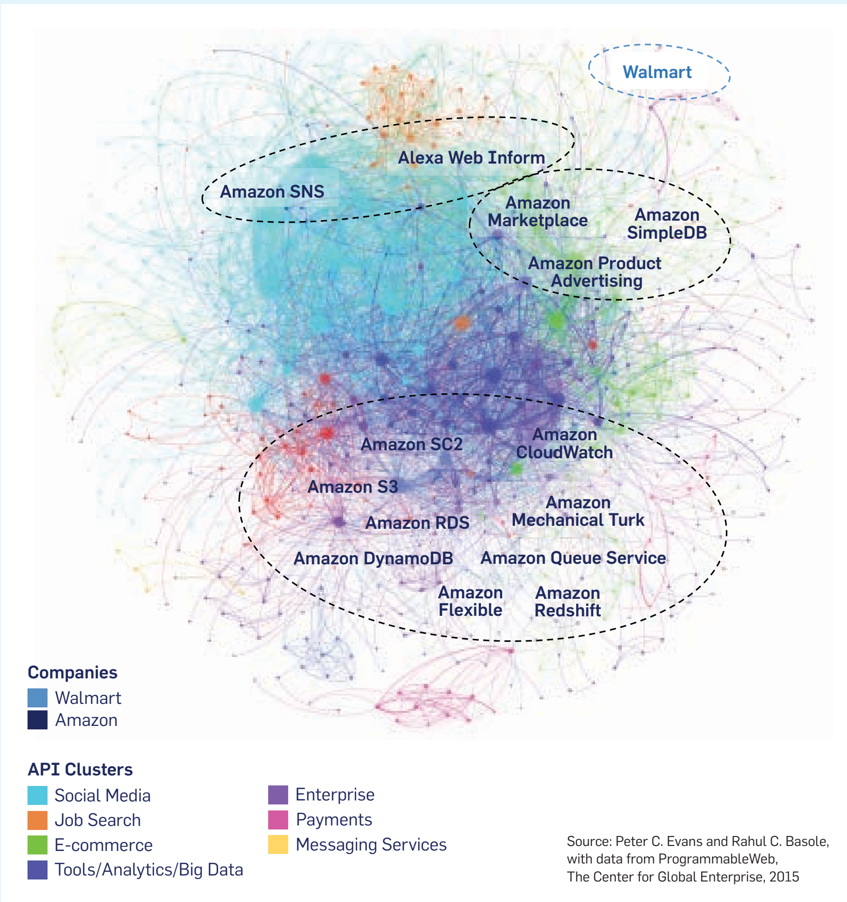
To better understand the broader structure of the API ecosystem, we leveraged a comprehensive curated dataset of over 11,000 APIs and 6,000 mashups, across 329 sectors. We converted the API data into a mashup network, where nodes represent APIs and edges represent if two APIs have been used jointly in a mashup. Edges are scaled proportional to the total number of mashups: the thicker the line, the more mashups were created using the corresponding two APIs. We then computed important network properties, including various centrality measures, to understand the position, prominence, and influence of APIs in the network. Finally, we visualized this network using a cluster-emphasizing force-directed layout algorithm (OpenORD),[7] identified and colored communities within this graph using a modularity-based approach,[9] and sized nodes according to their influence.[6]

The network analysis reveals interesting sectoral differences. The highest concentrations of APIs are primarily found in software tools, finance, enterprise, messaging, social networking, and e-commerce (see Figure 1), ranging between 400 to over 750 per sector. Mapping, science, government, and payments are also prominent sectors, each containing between 300 to 400 open APIs. But a large number of open APIs does not mean there will necessarily be a large number of API mashups. The most active areas for building new mashups use social, e-commerce, and mapping APIs. For example, to date there are 671 mashups built on social media APIs, and another 576 built on top of mapping APIs. Surprisingly, while there are more than 450 open finance APIs available, there are only 70 mashups that leverage these interfaces. Relatively low ratios of new mash-up generation are also found in health and payments. Comparing the average betweenness centrality scores of APIs in each category, a measure of relative network prominence, the most central sectors to the API network include social, images, search, and mapping sectors. This is followed by e-commerce, cloud computing, video, and payments.

Although it does not yet rise into the top 10 ranking of API sectors (by count), our analysis reveals transportation-related APIs are growing rapidly. These APIs offer a variety of functionality. For example, PlugShare's API enables owners of Teslas and other electric cars to locate charging stations and to a community of other electric vehicle users. The PlugShare Station API responds to over a million queries a month to locate and input recharging facilities and is now moving with other providers to build a common payment system through the Open Charge Point Protocol (OCPP). A growing number of local governments have moved to offer APIs to improve the information around public transportation, such as the Swiss Public Transport, Toronto Transit Commission, and the BART metro system in San Francisco. As the trend toward smart cities and connected vehicles grows, transportation-focused open APIs are only likely to grow. The Ford Motor Company and General Motors have established API programs that allow third-party software developers to build apps that will enable vehicles to include a wide range of "connected car" features ranging from voice recognition to advanced vehicle diagnostics.[8] Moreover, startups like MetroTech Net are building APIs that publish highly accurate and dynamic data on traffic patterns retrieved and analyzed from traffic cameras and other sensor data. Given its size and the value of improvements in information flows, the transportation segment appears destined to move from the periphery to the core of the API economy.

Our visualization also provides insight into variation in API strategies used by firms. Our visual analytic analysis revealed that only few traditional firms are active in the open API economy. For example, few if any major companies appear in the core component whether they are from banking, insurance, pharmaceuticals, food, transportation, or energy. Instead, we see the API economy is dominated by relatively recently established digital companies. Most central to this emerging eco-

Figure 2. API economy visualized: Amazon.com versus Walmart.



Source: Peter C. Evans and Rahul C. Basole, with data from ProgrammableWeb, The Center for Global Enterprise, 2015

system are companies that have built businesses around areas such as social, mapping, search, online payment, image sharing, video, and messaging. This includes digital platform companies like Google, Microsoft, Facebook, Amazon, eBay, Yahoo, Salesforce, and Twilio, as well as lesser-known companies like Quova, Anedot, and Zapier.

The retail sector provides a fascinating example of these stark differences.[5] Consider the position of Amazon.com and Walmart in the core API ecosystem (see Figure 2). Amazon.com has had an explicit policy of creating open APIs. The results of our visual analysis support that. Amazon has over 33 open APIs, which have been combined with many other APIs to create more than 300 mashups. Walmart, by contrast, has only one API that has yielded only one mashup. Considering the centrality scores of Amazon's APIs compared to that of Walmart, it is not surprising to observe that Amazon sits near the core of the API economy whereas Walmart is more peripheral.

This is not to say that traditionally brick-and-mortar retailers do not actively use API tools and services to support a range of approaches aimed at optimizing and personalizing device and screen experiences. In fact, they do. Macy's, for instance, has tapped Twitter's Audience Platform to reach more customers and boost sales. However, the number of open APIs that Macy's itself has established is very small. Among this small number, only one other mashup has been established. Amazon, by contrast, has a large and growing number of open APIs. This is particularly true in the e-commerce space where there are now 140 mashups built on Amazon APIs. Amazon is also clearly branching out beyond e-commerce into areas such as cloud, enterprise tools, mapping, messaging, networking, and payments. These areas are generally considered fundamental information infrastructure services for the emerging Internet of Things industry. As a result, it may be important to consider whether it will be necessary to reclassify Amazon's industry peer group in time.

The growth of the API economy, however, is not without its risks. For several years a battle has been waged over whether APIs can be copyrighted or are they exempt and more appropriately subject to the doctrine of "fair use." A key attractiveness of APIs is the ability to copy and repeat the best bits.[10] Another concern is that an API provider can abruptly change pricing terms or even turn off an API that has become critical input into services that others have created. The degree to which these risks represent speed bumps or something more serious remains to be seen. Meanwhile, investment in APIs is taking place across a wide range of sectors and companies discover new ways APIs can drive productivity, reduce costs, and enhance flexibility of their operations and services. As API networks grow richer and more complex, visual analytic techniques will provide a valuable tool for discovering, tracking, and sensemaking of the evolution of API ecosystems and what it means for different industries and specific enterprise strategies. C

References

1. Basole, R.C. Visual business ecosystem intelligence: Lessons from the field. *IEEE Computer Graphics and Applications 5* (May 2014), 26–34.
2. Basole, R.C. et al. Understanding business ecosystem dynamics: A data-driven approach. *ACM Transactions on Management Information Systems (TMIS) 6*, 2 (June 2015), 6.
3. Curtis, J. Thomson Reuters opens sources data APIs for developers. *ITPRO* (July 1, 2015); http://bit.ly/1MRWuys
4. DuVander, A. Which APIs are handling billions of requests per day. *ProgrammableWeb* (May 23, 2015).
5. Evans, P.C. and Basole, R.C. Decoding the API Economy with Visual Analytics; http://bit.ly/1Us6bWU
6. Freeman, L.C. A set of measures of centrality based on betweenness. *Sociometry 40*, 1 (Jan. 1977), 35–41.
7. Martin, S. et al. OpenOrd: An open-source toolbox for large graph layout. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series 7868*, 5 (Jan. 2011), 4.
8. Newcomb, D. Ford, GM open their dashboards to outside developers. *Wired* (Jan. 8, 2013).
9. Newman, M.E. Modularity and community structure in networks. In *Proceedings of the National Academy of Sciences 103*, 23 (2006), 8577–8582.
10. Samuelson, P. *Oracle v. Google*: Are APIs copyrightable? *Commun. ACM 55*, 11 (Nov. 2012).
11. Tilson, D., Lyytinen, K., and Sørensen, C. Research commentary-digital infrastructures: The missing IS research agenda. *Information Systems Research 21*, 4 (2010), 748–759.

Peter C. Evans (pevans@thecge.net) is Vice President, The Center for Global Enterprise.

Rahul C. Basole (basole@gatech.edu) is Associate Professor, School of Interactive Computing and Director, Tennenbaum Institute at Georgia Institute of Technology, Atlanta, GA.

DOI:10.1145/2856451

Carl Landwehr

Privacy and Security
Privacy Research Directions

What must we learn in order to support privacy requirements as technology advances?

Not since the early 1970s, when computing pioneer Willis Ware chaired the committee that produced the initial Fair Information Practice Principles,[10] has privacy been so much in the U.S. public eye. Edward Snowden's revelations, as well as a growing awareness that merely living our lives seems to generate an expanding "digital exhaust," have triggered many workshops and meetings.[1,5,11,12] An alphabet soup of advisory groups—PRG,[a] PCLOB,[b] PCAST[c]—have produced privacy-related reports.[2,6,7–9] The wheels are turning at NITRD[d] to produce a national strategy for privacy research, perhaps paralleling the federal strategy for cybersecurity research and development.[3] I have participated in a number of these and have developed my own view of privacy and privacy research. My U.S. perspective may differ from those from different backgrounds; privacy views vary with culture.

Some characterize privacy in terms of harms: to have suffered a loss of privacy that is actionable, there must be some way to characterize the harm an individual suffers as a result of the privacy breach. This practical view motivates many privacy concerns: the data revealed may cause the loss of a benefit or service. However, this view runs into trouble where the damage seems primarily psychological—I really do not want my neighbor to know if I have unconventional sexual practices or have had cosmetic surgery, and I may suffer psychologically or emotionally from such revelations, but it may be difficult to characterize the loss in a way that can be compensated. Further, it may be difficult to know that a harm has occurred—I may be deprived of an opportunity to be employed by the disclosure of private information through a breach I am unaware of.

Domain-specific privacy definitions and rules may be needed. I once preferred the uniform structure of privacy

[a] President's Review Group on Intelligence and Communications Technologies: http://1.usa.gov/1dY0nmm

[b] Privacy and Civil Liberties Oversight Board: http://bit.ly/1NSRto6

[c] President's Council of Advisors on Science and Technology: http://1.usa.gov/1NSRxnJ

[d] National Information Technology Research and Development program office: http://1.usa.gov/1QB4W97

IMAGE BY ROB PONGSAJAPAN/FLICKR

Privacy Observations

- **"Privacy" has many definitions. Perhaps most stringent is privacy as the right to control fully the flow of one's personal data. But only those willing to forego most modern communication, transportation, and payment systems could today approach such an objective.**

- **Even people who say "I have nothing to hide" still value privacy. What they usually mean is "I have nothing to hide from law enforcement or others with legal authority to examine my records." They do not mean, "I have nothing to hide from a parent, sibling, child, neighbor, blogger, or journalist."**

- **The 40-year-old Fair Information Practice Principles remain sensible, but the average citizen has felt their effect mostly in paperwork generated by the notice and consent provisions.**

- **The schemes used to inform people about privacy and gain their consent are ineffective. The increasing ubiquity of automated sensing makes them more so. At one recent workshop, no panelist or audience member was willing to defend current "notice and consent" mechanisms.**

- **Government and private surveillance mechanisms generate substantial quantities of data without engaging the subjects under surveillance. Consent or even notice seems infeasible.**

- **Mass surveillance pits expected gains in national security for society as a whole against privacy of individuals, an unequal comparison. Mass surveillance affects society. Endorsing auditing and oversight mechanisms to enforce "good behavior," from video monitoring to catch fraud in stores to Inspectors General for Federal agencies, suggests individuals—and by extension, society—behave differently when knowingly watched.**

- **Anonymization of data can help preserve privacy, but it can also limit the benefits gained from analyzing a dataset, while remaining vulnerable to serious attempts at re-identification.**

regulations of the European Union's Data Protection Directive to the U.S. patchwork of laws and regulations that separately cover health records, educational records, legal proceedings, business transactions, and so on. Now I am less sure. One of the definitions of privacy that continues to seem useful to me is that privacy is preserved when information remains within a specified context—financial information stays with my financial advisor or broker, religious information with my religious counselor, health information with my medical practice, educational information with my school, and so on.[4] Perhaps it is better to continue to use policies that take these contexts and the semantics of the information into account and to strive for, but not insist on, unification. In this case, it is also necessary to specify when information can be allowed to move between normally isolated contexts, for example to deal with an emergency.

A legal regime in which data "belongs" to an owner who then has complete dominion over it is too simple to accommodate future needs both for preservation of useful notions of privacy and for productive use of data. The fact that a private party engages in a transaction with a business or public service, be it a grocery purchase, telephone call, email message, or database query, should not necessarily entitle the business or service to unlimited use or publication of the data from that transaction. In fact, there are already many cases where there are competing interests in particular data and the custodian of data does not exercise complete control over it. In the future, it may make sense for data custodians to be bound by "responsible use" policies that depend on how the data was collected, the domain of data collected, and other factors. Constraints need to accompany data in a form that enables the recipient to continue to enforce them easily.

To gain the benefits of having large datasets to analyze (perhaps most apparent in healthcare, but in many other domains as well), anonymization and differential privacy will be of some, but limited, use. It will be essential for the subjects whose data is collected to trust that the custodians of their data will handle it properly, in accordance with whatever responsible use policies are established for it. Otherwise the subjects may withhold their data and the societal benefits will be lost. Because humans and systems are fallible, there will undoubtedly be some instances where sensitive data is lost or mishandled. To maintain public trust in the face of such incidents it will be important to assure data subjects that the custodians can be held to account: mechanisms must be provided that enable injured parties to detect misuse and obtain redress.

Potential Areas of Research

With those thoughts in mind, I offer some potential privacy-related research areas, in no particular order, and with some overlap.

Effective privacy definitions, and in particular, domain-specific definitions of privacy. HIPAA (internationally considered a strongly protective model) and other regulations already provide what might be considered domain-specific rules for data privacy in healthcare. These have received some research attention. Other domains, including law enforcement, finance, and intelligence, might benefit from efforts to characterize the data involved, how it should be handled within the domain of use, and under what conditions and controls it might be allowed to flow to different domains.

Effects of surveillance on social groups of different sizes. For example, has the chilling effect of surveillance on free expression been studied systematically? I am not a social scientist, so I may be unaware of research in this area, but research results, if they exist, must be aired, and if they do not exist, they deserve study.

Development of better languages for specifying privacy policies. Not a new area of research, perhaps, but effective solutions do not seem to be available. Languages are needed that enable specification of policies that can be enforced algorithmically and also that can be understood by the public.

Accountability mechanisms for privacy violations. Both detection of a privacy violation and the ability to trace the violation back to responsible individuals are important. Detecting violations will of course imply there is an expressed policy that is to be enforced. Tracing violations has implications for authentication and auditing. Some financial systems incorporate accountability mechanisms for purposes of fraud deterrence and detection, and some health records systems incorporate mechanisms to account for privacy violations, but these mechanisms will need to find homes in a much broader range of systems with various privacy policy enforcement needs. The ability to support provenance for both data and programs at scale is needed.

Techniques and mechanisms for tracking information flow. To me, the fundamental nature of a privacy violation is an improper information flow. Privacy policy needs to distinguish proper and improper flows and to enable authorization of exceptions. Capabilities for tracking the flow of information within programs have matured substantially in the past decade. Those capabilities need to be extended to systems of programs.

Techniques for binding policies to data and enabling distributed enforcement. If the data itself can carry the policy to be enforced along with it, each domain in which it appears can apply appropriate enforcement. One might imagine the data also collecting an audit trail as it moves from place to place. Cryptographic approaches may help.

Techniques for identifying and quantifying benefits of large-scale data analysis and costs of privacy harms. It is a tall order to model in advance the benefit one may gain from analyzing some large dataset, since one does not know what might be learned. On the other hand, the analysis is usually undertaken with some objective in mind, and it might be possible to quantify what is to be gained if that objective is realized. Similarly, some resources need to be devoted to anticipating privacy harms and what damages may occur if large datasets are abused. These kinds of trade-offs must be understood as well as possible at the time people are deciding whether or not to initiate new projects if there is to be any rigorous risk/benefit analysis in this sphere.

Conclusion

Privacy may be difficult to define and culturally dependent, but it nevertheless seems to be universally valued. Future computing systems must incorporate mechanisms for preserving whatever privacy policies people and societies decide to embrace, and research is needed to identify those mechanisms and how they may best be applied.

Privacy may be difficult to define and culturally dependent, but it nevertheless seems to be universally valued.

References

1. National Academy of Science Raymond and Beverly Sackler U.S.-U.K. Scientific Forum on Cybersecurity, Dec. 8-9, 2014, Washington, D.C.
2. Networking and Information Technology Research and Development (NITRD) Program. Report on Privacy Research Within NITRD. (Apr. 2014); http://1.usa.gov/1lUFakz
3. Networking and information Technology Research and Development (NITRD) Program. Trustworthy Cyberspace: Strategic Plan for the Federal Cybersecurity Research and Development Program. (Dec. 2011); http://1.usa.gov/1NgEUFN
4. Nissenbaum, H. Privacy as contextual integrity. *Washington Law Review 79* (2004), 119–158.
5. 2015 NSF Secure and Trustworthy Cyberspace PI meeting (Jan. 5–7, 2015), Washington, D.C.; http://bit.ly/1OXJxVV
6. President's Review Group on Communications and Intelligence Technologies. *Liberty and Security in a Changing World.* (Dec. 12, 2013); http://1.usa.gov/1cBct0k
7. President's Council of Advisors on Science and Technology, *Big Data and Privacy: A Technological Perspective.* (May 2014); http://1.usa.gov/1rTipM2
8. Privacy and Civil Liberties Oversight Board, *Report on the Surveillance Program Operated Pursuant to Section 702 of the Foreign Intelligence Surveillance Act*, July 2, 2014; http://bit.ly/1FJat9g
9. Privacy and Civil Liberties Oversight Board. *Report on the Telephone Records Program Conducted under Section 215 of the USA PATRIOT Act and on the Operations of the Foreign Intelligence Surveillance Court.* (Jan. 23, 2014); http://bit.ly/1SRiPke
10. *Records, Computers, and the Rights of Citizens: Report of the Secretary's Committee on Automated Personal Data Systems* (July 1973). Dept. of Health, Education and Welfare, DHEW(OS), 73–94; http://1.usa.gov/1RIZLom
11. Second Annual CATO Surveillance Conference (Oct. 21, 2015), Washington, D.C.; http://bit.ly/1MEsY05
12. U.S. Privacy and Civil Liberties Oversight Board (PCLOB) Workshop “Defining Privacy”, Nov. 12, 2014, Washington, D.C.; http://bit.ly/1ReOmgT

Carl Landwehr (carl.landwehr@gmail.com) is Lead Research Scientist the Cyber Security Policy and Research Institute (CSPRI) at George Washington University in Washington, D.C., and Visiting McDevitt Professor of Computer Science at LeMoyne College in Syracuse, NY.

Calendar of Events

February 14–17
TEI ‘16: 10th International Conference on Tangible, Embedded, and Embodied Interaction,
Eindhoven, the Netherlands,
Sponsored: ACM/SIG,
Contact: Saskia Bakker,
Email: s.bakker@tue.nl

February 21–23
FPGA’16: The 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays,
Monterey, CA,
Sponsored: ACM/SIG,
Contact: Deming Chen,
Email: dchen@illinois.edu

February 22–25
WSDM 2016: 9th ACM International Conference on Web Search and Data Mining,
San Francisco, CA,
Sponsored: ACM/SIG,
Contact: Paul N. Bennett,
Email: paul.n.bennett@microsoft.com

February 27–March 2
CSCW ‘16: Computer Supported Cooperative Work and Social Computing,
San Francisco, CA,
Sponsored: ACM/SIG,
Contact: Meredith Ringel Morris,
Email: merrie@microsoft.com

March

March 1
I3D ‘16: Symposium on Interactive 3D Graphics and Games,
Sponsored: ACM/SIG,
Contact: Chris Wyman,
Email: chris.wyman@ieee.org

March 2–5
SIGCSE ‘16: The 47th ACM Technical Symposium on Computing Science Education
Memphis, TN,
Sponsored: ACM/SIG,
Contact: Jodi L. Tims,
Email: jltims@bw.edu

March 7–10
IUI’16: 21st International Conference on Intelligent User Interfaces,
Sonoma, CA,
Sponsored: ACM/SIG,
Contact: John O’Donovan,
Email: jodmail@gmail.com

DOI:10.1145/2856455

Rick Adrion, Renee Fall, Barbara Ericson, and Mark Guzdial

Education

Broadening Access to Computing Education State by State

Influencing computer science education at the state level.

IMAGE BY ANDRIJ BORYS ASSOCIATES/SHUTTERSTOCK

ARKANSAS, SAN FRANCISCO, Chicago, and New York City have made headlines with their announcements to make computer science available in every school to every student. That is a challenging goal. Actually making computing education available to everyone involves many policy decisions. Changing schools in the U.S. is difficult because education is largely a state prerogative. Standards, teacher certification, curricula, and graduation requirements are determined within individual states and territories, not by the federal government. Many states delegate the decisions down to cities, counties, or individual districts or schools, making the decision making even more distributed. Charter schools, magnet schools, virtual schools, homeschooling, school choice, and funding disparities among districts further complicate assuring broad and equitable access to computing education.

Reaching everyone through formal education pathways. If we want to give everyone access to computing education, we need to begin to do it through formal education pathways, for example, primary or elementary school, middle school, high school, community colleges, and universities. Informal computing education is unlikely to reach everyone. The formal computing education pathways are our best chance to broaden participation in computing. Female students and underrepresented minorities are less likely to seek out afterschool computing clubs or summer computing camps—some will, but we will not reach everyone that way. Making computer science available in public school systems requires that states and districts create policies that address several questions: Where do you teach computing? Do you integrate computing into existing mathematics or science classes? Do you teach separate computer science classes? The answers are likely to be different at the elementary, middle school, and high school levels.

Who teaches computing? In most U.S. states, computer science is classified as a business or career-technical education (CTE) subject. That classification raises issues regarding how teachers are assigned or considered highly qualified to teach computing. In turn, that affects in-service and pre-service professional development. Will a school, district, or state education authority require a business or CTE teaching credential? Can other teachers (such as math, science, technology) teach computing courses? Do you only provide the business and CTE teachers computer science professional development and education, or do you try to reach a broader range of teachers?

How do we certify that a teacher knows how to teach computer science? Most states offer no teacher certification in computing. Without a credentialing process, schools will not be able to tell whether a teacher is qualified to teach computer science. Without the possibility of earning some kind of credential, teachers may be unwilling to go through additional professional development. Current computing teachers may find new requirements for certification limiting their opportunity to continue to teach computing courses. Without certification, colleges are unlikely to create pre-service curricula and students planning to teach are unlikely to demand them.

Why do students take computing? Most high school computer science (CS) classes in the U.S. are elective; so few students take them—often only white or Asian males. If you want more students in computer science classes, require computer science (which is challenging to implement) or have CS classes meet some requirement that students care about. Some states count CS classes as meeting mathematics or science or even world language requirements for high school graduation. Georgia had a dramatic rise in the number of students taking the Advanced Placement CS exam after the AP CS course started counting as a science credit toward high school graduation.

What are we trying to teach? States set standards about desired learning outcomes. Some states are creating computer science standards, while other states are including computer science in existing standards (for example, in science). How will curricula and assessments be aligned with new, revised, or existing standards? Will they be tested or otherwise required?

The formal computing education pathways are our best chance to broaden participation in computing.

Reduce higher-education friction. Public college and university systems are also under state control to varying degrees. Community and technical college computing programs tend to serve more diverse communities. Easing the community college to four-year college transition can contribute to increasing diversity and broadening access in college and university computer science departments. Two-year college systems and articulation mechanisms vary by state. If you want to get more community college students to successfully transfer to state universities, you solve that problem at the state level.

Building a Community of State CS Education Leaders

In 2012, two National Science Foundation Broadening Participation in Computing (BPC) Alliances merged (CAITE and "Georgia Computes!") to create the Expanding Computing Education Pathways (ECEP) Alliance.[a] The authors of this column are the leads on ECEP. We aimed to use the lessons learned in Massachusetts and Georgia and the expertise of leaders in computing education to help other states improve and broaden computing education.

ECEP began working with four states: Massachusetts, Georgia, California, and South Carolina. We soon realized we could not just "translate the lessons learned" from one state to another. States vary dramatically, for example, in terms of how much control the state department of education has versus individual school districts, how teacher credentialing works, and who decides whether a particular class "counts" toward high school graduation requirements. States like Georgia and South Carolina are more centralized, meaning the state department of education defines classes and high school graduation requirements. States like Massachusetts and California distribute control down to local districts. The process for creating new requirements in Georgia is unlikely to work in California.

In 2015, we added additional states to ECEP so that the cohort now includes 11 states and Puerto Rico. We host face-to-face and virtual meetings where leaders of CS education reform in these states talk to one another about the issues they are facing in their states. This community of state leaders is the most important resource we have to offer in ECEP.

The lessons learned from Massachusetts and Georgia are useful for states joining ECEP, but so are the lessons from the other states. We have been surprised at how much our state leaders draw ideas from each other. South Carolina leaders used a teacher survey that was invented in Massachusetts. Utah draws inspiration from a Texas teacher recruitment strategy. What our state leaders find most useful about ECEP is access to other state leaders who share similar goals, for example, to broaden participation in computing by changing education pathways in their states.

A Model for State-based Efforts

Based on the ECEP experiences with states making public policy changes to improve K–16 computing education, we have developed an initial set of steps that we recommend to any state planning an effort to broaden access to computing education for K–12 and higher education students. We require states to have taken the first three steps in this process before they can join our ECEP state cohort, but we also believe it applies to any state whether or not they hope to partner with ECEP.

Step 1: Identify a State Leader. A state should have one or more leaders who are willing to participate in the ECEP cohort. The current ECEP state leaders cover the spectrum of computing education stakeholders, including high school teachers and administra-

[a] See http://ecepalliance.org

We have been surprised at how much our state leaders draw ideas from each other.

tors, university faculty (from both CS and education), industry leaders, and staff from state departments of education. It is a more difficult requirement to meet than we expected. Some potential leaders are in state departments of education, and their participation is limited by department policies. Some states have a lot of people working in computing education, but no one who is willing to coordinate efforts across the state.

Step 2: Figure out where you are and how change happens. As we described earlier, states vary in many, but predictable, ways. There must be a process by which high school graduation requirements are determined. There must be some process for managing teacher certification. We do hear from potential state leaders who have no idea how education policy works in their state, or even whether they have CS classes being taught in their state. That is a baseline requirement: change cannot start until you know where your state is and how change happens in your state.

Step 3: Organize a cross-sector committee. The leaders who are most successful influencing computing education public policy join forces across sectors. In Georgia, we started out with a coalition that crossed universities, high schools, and the department of education. We really got the attention of the legislature and the governor when industry started pushing for change, too. South Carolina has a steering committee that crosses all these sectors. Some states have computing education organizations—California has ACCCESS, Texas has TACSE, and Massachusetts has MassCAN. State leaders should be working with people from education, industry, and government.

Step 4: Find initial funding. The first three steps are essential. The fourth step is necessary to make change, and ECEP provides some small financial help to its members. Improving and broadening computing education requires some significant funding, for example, for teacher professional development. There are smaller-ticket items that are useful early in the process.

- Several of the ECEP cohort states have written landscape reports describing the current state of computing education and setting priorities for change. California leaders called their landscape report *In Need of Repair*.[b] The landscape report speaks to education policy stakeholders, to describe why and where computing education needs to change in the state.
- A summit meeting on computing education is where the computing education leaders gather and invite in the stakeholders (for example, public policymakers in the state government, industry leads, district superintendents, and school principals) who need to hear about the landscape report. Summits galvanize the community and generate shared goals for making progress in improving and broadening participation in computing education.

Those of us leading the ECEP Alliance do not have a recipe for change that works in every context. We do see a set of steps in a process that is working in several states. We have learned we cannot always predict what states will most need in order to make progress or what pitfalls lie ahead along the path. We are finding that, together, our cohort of state leaders is helping each state figure that out. ■

[b] See http://www.exploringcs.org/wp-content/uploads/2010/09/InNeedofRepair.pdf

Rick Adrion (adrion@cs.umass.edu) is Professor Emeritus in the School of Computer Science at the University of Massachusetts Amherst.

Renee Fall (rfall@cs.umass.edu) is project manager for the Commonwealth Alliance for Information Technology Education in the College of Information and Computer Sciences at the University of Massachusetts Amherst.

Barbara Ericson (ericson@cc.gatech.edu) is Director of Computing Outreach and Senior Research Scientist at Georgia Institute of Technology, Atlanta, GA.

Mark Guzdial (guzdial@cc.gatech.edu) is a professor in the College of Computing at Georgia Institute of Technology, Atlanta, GA.

DOI:10.1145/2856457

George V. Neville-Neil

Kode Vicious
Code Hoarding

Committing to commits, and the beauty of summarizing graphs.

Dear KV,
Why are so many useful features of open source projects hidden under obscure configuration options that mean they will get little or no use? Is this just typically poor documentation and promotion, or is there something that makes these developers hide their code? It is not as if the code seems broken. When I turned these features on in some recent code I came across, the system remained stable under test and in production. I feel code should either be used or removed from the system. If the code is in a source-code repository, then it is not really lost, but it is also not cluttering the rest of the system.

Use It or Lose It

Dear Use It,
There are as many reasons for exposing or hiding code as there are coders in the firmament of programming. Put another way, there is more code hidden in source repos than are dreamt of in your ... well, you get the idea.

One of the most common forms of code hiding I have encountered when working with any code, not just open source, is the code that is committed but to which the developers are not fully committed themselves. Sometimes this is code that supports a feature demanded by sales or marketing, but which the developers either do not believe in or which they consider to be actively harmful to the system.

"I'd like a stop button."

"A stop button, if used, will destroy all of our data."

"That's OK, I want a stop button, and I want it to be *red*."

IMAGE FROM SHUTTERSTOCK.COM

These types of features are often buried in the system and can be turned on only after clicking through dialog boxes with increasingly dire warnings.

A more aggravating form of buried code comes from developers who are unwilling to commit to a feature, usually because of a lack of understanding of the code. It is not uncommon in long-running projects for developers to come and go and for newer developers to fear existing code they have not taken the time to understand or measure. A host of excuses will be deployed against enabling the code—because it is too slow, too old, or has too many bugs. The problem is often compounded by the often-stubborn nature of most developers, who, until they are hit with overwhelming evidence, refuse to believe anything that does not match their worldview. I cannot legally recommend actually hitting developers, but I will say that I find my hard-covered notebooks do tend to make a very nice thudding sound on conference-room tables.

If you are working on a project and come up against this type of intransigence, there are better and worse ways of dealing with it. Arguing piecemeal, or tit-for-tat, with a developer is time consuming and, like teaching a pig to dance, only aggravates the pig. One way to handle the topic is political, a word I realize is anathema to much of this audience. Make your case to other developers for building a consensus on the right way of handling a feature. This is, surprisingly, a worthwhile exercise that has longer-term benefits to a project. It turns out it is sometimes easier to convince someone of something when other people they trust are all saying the same thing.

Making measurements and gathering evidence in support of turning on a feature are also valid methods of dealing with developer intransigence. As a side benefit, any test code and results you generate can be used again later, as test code, when the code inevitably changes. If you put the evidence on nice, heavy paper, it can, as pointed out earlier, be used more directly to make your case.

KV

Dear KV,
I have a co-worker who has spent the past several months making large-scale changes to our software in his private branch. While some of these changes are simple bug fixes that get placed back into the main branch of development, a lot of it is performance-related work that he emails about every week or two. These email messages contain nearly endless pages of numbers that

are explained only briefly or not at all. I want to ask if he has ever heard of graphs but fear it might just make him think I want to review more of his long email messages. Is there a better way to explain to him that we need a summary rather than all the data?

Bars and Lines

Dear Bars,

It continues to amaze me that most people never had to write lab reports in high school science classes. I don't know about you, but that is where I learned to summarize data and to create graphs of the data from experiments carried out during the class. My favorite statement about the use of graphing comes from Robert Watson, who teaches a graduate class on operating systems research at the University of Cambridge: "Graphs serve a number of functions in lab reports—not least, visually presenting a large amount of data, allowing trends in that data to be identified easily (e.g., inflection points), but also to allow a visual comparison between different sets of results. As such, if we set up an experimental comparison between two sets of data (e.g., static vs. dynamic performance), putting them on the same graph will allow them to be easily compared."

That is an instruction to graduate students who are required to summarize their data in lab reports for his course. If you remove the words "lab reports" in the first sentence, you could simply try beginning there in explaining to your co-worker why no one is reading his email messages full of data.

Once you have convinced him that graphs are good, you will probably have to help make good graphs. Unlike when we made graphs in class, with paper, pencils, and straightedges, there is now a plethora of software available that will take raw data and graph it on your behalf. The problem is you still need to know which question you are trying to answer and whether the summary is helping or hindering your understanding of the underlying problem.

Graphing a set of numbers is not just a matter of dumping them into a program and seeing a pretty picture; it has to do with taking a set of data and making an easy-to-understand representation that can be shared among a group of people in order to make a judgment about a system. To make any sort of judgment, you need to know what you are trying to measure.

All the measurements you are trying to graph need to share a common trait; otherwise, you risk comparing apples and oranges. For example, a repeated measure of network bandwidth, over time, is a measurement of the number of bits per unit of time (often seconds) seen during some sampling period, such as "every five minutes." As long as all the measurements to be compared share those attributes, all is well, but if the sampling period between two sets were different—for example, if one is hourly and another is every five minutes—that would be a false comparison.

When graphing data for comparison it is important to ensure your axes line up. Generating five graphs with software that automatically resizes the graph to encompass all the data gives results that cannot be compared. First find the maximal *x*- and *y*-ranges of all your data, and use these maxima to dictate the length and markings of all of the graphs so they can be compared visually. If one set of data has significant outliers, it can make the entire set of graphs useless. As an example, a time-series graph of network latencies that is mostly measured in microseconds will be rendered unusable by a single measurement in milliseconds, as all that you will see is a slightly thickened line following the *y*-, or time, axis, and a single spike somewhere in the middle of the graph. Graphs that do not handle outliers correctly are completely useless.

Most graphing software does a poor job of labeling data, and yet the labeling is an important and intrinsic part of the story. What does that thin blue line really indicate? It is going up and to the right. Is that good? Or is that the number of unclosed bug reports? Picking reasonable colors and labels for your data may sound like too much work, but much like commenting your code, it will pay off in the end. If you cannot remember which dots are from which dataset and therefore what the graph even means, then the graph is useless.

Finally, a quick note on summarizing data. An important distinction that is often overlooked or misunderstood is that between the mean and the median. The mean is the simple average of a set of numbers and is the most commonly used—and misused—type of data summary. The problem with using the mean to summarize a set of data is that the data must be normal for the mean to have any meaning. When I say "normal," I do not mean to imply it had a happy childhood. Normal data is evenly distributed along a bell curve, which does not describe many sets of data. If data is heavily skewed, or has significant outliers, then the median is preferred. The median is more difficult to calculate by hand, but I hear these computer things that are all the rage can calculate it for you.

While there are large open source and commercial packages that will graph your data, I prefer to start simply, and, therefore, I commend to you ministat (https://www.freebsd.org/cgi/man.cgi?query=ministat), a program written by Poul-Henning Kamp that is included in the FreeBSD operating system. It generates simple text graphs based on columns of data and will do all the work of telling you about means, medians, and statistical significance. If, at some point, your data has to be communicated to those who are from another world, as you indicated in your question, you can then plot it with R (https://www.r-project.org), but that's a story for another time.

KV

Related articles on queue.acm.org

Commitment Issues
Kode Vicious
http://queue.acm.org/detail.cfm?id=1721964

Unlocking Concurrency
Ali-Reza Adl-Tabatabai, Christos Kozyrakis, and Bratin Saha
http://queue.acm.org/detail.cfm?id=1189288

Software Needs Seatbelts and Airbags
Emery D. Berger
http://queue.acm.org/detail.cfm?id=2333133

George V. Neville-Neil (kv@acm.org) is the proprietor of Neville-Neil Consulting and co-chair of the *ACM Queue* editorial board. He works on networking and operating systems code for fun and profit, teaches courses on various programming-related subjects, and encourages your comments, quips, and code snips pertaining to his *Communications* column.

DOI:10.1145/2811254

Satish Chandra, Suresh Thummalapenta, and Saurabh Sinha

Viewpoint
Lessons from the Tech Transfer Trenches

Moving from the research realm to real-world business application.

As researchers employed by a company, we wear two hats. One of our roles is to participate in the research community. The other is to channel some of our research toward "business impact" on the company we work for. In this Viewpoint, we present our experiences in taking our research project on test automation to business impact. Notwithstanding differences between organizations, we hope our colleagues in other research institutions will find some of these lessons useful in their own attempts toward business impact.

The work we describe here was done in the context of IBM's service delivery organization. Since this context may be unfamiliar to many readers, we first explain it briefly. Software businesses fall roughly into two categories: those (for example, Microsoft) that manufacture and sell software—essentially licenses to pre-packaged software—to other businesses and consumers, and those (for example, Accenture) that sell software development as a service to other businesses; some companies engage in both kinds of businesses. Both product and services businesses employ lots of software engineers, and both serve very large markets. Product companies differentiate their offerings based on the features in their products. By contrast, services companies differentiate their offerings based on the cost and quality of service. As such, prowess in software engineering is directly connected to their success.

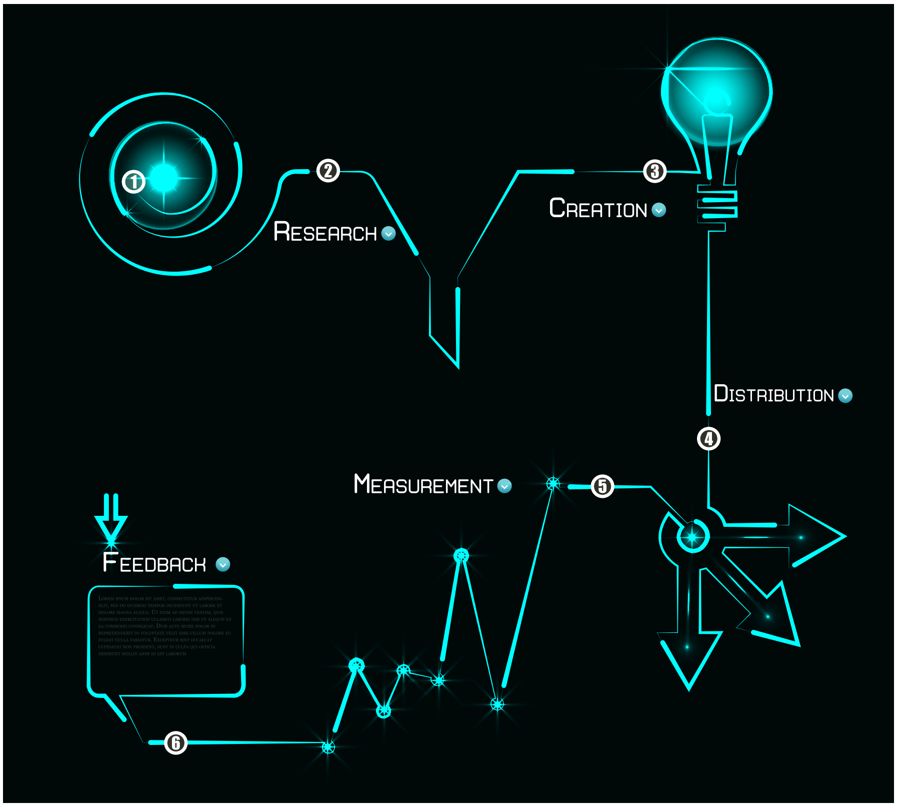


IMAGE FROM SHUTTERSTOCK.COM

The topic of our research project was regression testing for Web applications. For commercial Web applications, such as an e-commerce portal for a bank, there are thousands of test cases, to be run against a large number of browser and platform variants. Moreover, the application itself is updated frequently. Companies that own these Web applications prefer not to invest in in-house staff to carry out this testing, and so this work is often outsourced to service providers. Large service providers, including IBM, offer a variety of testing services, including regression testing.

Given the scale of the problem and the limited time available in a regression testing cycle, comprehensive manual test execution is generally infeasible. This is where test automation comes in. The idea is to write programs,

Over time, we realized trying to change the ways of existing projects might be a fruitless initiative.

a.k.a. test scripts, which drive these Web applications programmatically by taking control over a browser and simulating user actions. Unfortunately, there are costs to automation. One of them is the cost of creating these test scripts in the first place. The second is an often hidden cost, which is to keep these scripts working in the face of small UI changes. If not done properly, the cost of continual test maintenance can negate the benefits of automation.

In our research, we proposed a new approach to test automation. Since test cases are initially available as "manual" tests, which are written as test steps in plain English, our idea was to generate a program almost automatically based on lightweight natural language processing and local exploration, as is common in approaches to program synthesis. This semi-automation decreases the cost of entry into test automation, because this approach requires less programming expertise than would be needed otherwise. Our approach also addresses the issue of script fragility: it represents the generated scripts in a DOM-independent[a] manner. We called our tool ATA, for Automating Test Automation.[3]

Given a research prototype, and given our context—IBM services delivery organization—we wanted to see if there was an opportunity of business impact by deploying the ATA in client accounts. The services delivery organization in IBM is basically set up as separate account teams, each servicing a certain client. We decided to approach one-by-one the account teams in which we knew test automation was one of the deliverables.

Our research affiliation was effective in opening the doors. In most cases, we were able to quickly arrange for a demo to the account team. These demos worked well, and in many cases, we were able to persuade the team to allow us to do a pilot engagement with them. This was a good outcome, as it was clear that the approach we presented was at least of preliminary interest to the practitioners of test automation; after all, the pilot implied a time commitment on behalf of the practitioners too.

This brings us to our first lesson: for tech transfer, *the technology has to be of direct applicability to the business* (see the accompanying table for a summary of lessons learned).

The actual conducting of pilots turned out to be a much more difficult undertaking than we had anticipated. The goal of a pilot is to show the technology works in a real-world context, that is, in the context of a client account. This immediately exposed all sorts of assumptions we had baked into our research prototype. We had to extend the prototype to accommodate issues such as poorly written manual tests, missing test steps, exceptional flows, verification steps, and so on. The automatic exploration option in ATA did not work as well as we had hoped, and besides, users did not like ATA automatically crawling through their apps. Each account with which we piloted exposed a fresh set of shortcomings, and we had to fix ATA to handle all of these issues. Since the account teams we worked with were all located in Bangalore, we could not risk getting any negative comments on ATA leaking out in this more-or-less close knit community!

After months of hard work, we felt we had shown ATA works well in real situations. Yet, no team signed up to adopt ATA in their day-to-day work. This was a disappointment to us. Delivery managers were being, we felt, overly conservative in the way they executed their projects. On the other hand, we were asking for a lot: we were asking their teams to change the way they carried out test automation. After all, ATA was not a tool whose output you could take or ignore and continue as usual; ATA was how you would get your work done. If ATA did not work out, a lot of time would have been wasted and the

[a] DOM is the document object model, a structure used by all browsers to represent a Web page.

Summary of lessons for tech transfer.

Relevance	Researchers should talk to their colleagues on the development side frequently to identify opportunities where a research insight can address a current problem.
Cost-benefit trade-offs	Researchers should be prepared to propose and implement measurements on both costs and benefits that could be collected in an initial trial period.
Supporting early users	Early adopters deserve extra consideration for the risk they take, and for the proof points they will generate.
Organizational dynamics	Decisions on technology adoption may not be based purely on users' perception of the technology in question.
"Last mile"	Researchers who desire their work to be adopted in production should be prepared to walk the long road from a research prototype to a production-ready tool.

project would fall behind schedule. Delivery managers are foremost responsible for predictable and consistent delivery, and were understandably circumspect about adopting ATA in their teams. Finally, their clients were satisfied with the existing level of productivity, and there was no incentive to change the status quo.

This brings us to the second important lesson: although we had established the technical feasibility of using ATA in real projects, we had not made a case that the *benefits outweighed the costs and the risks involved*. Just because a research tool is available for free does not mean people will adopt it. People wanted to see a prior deployment as a comfort factor in being able to defend adopting ATA, and we had none to show. Moreover, we had no data indicating the actual productivity improvements when using ATA.

This chicken-and-egg problem found its resolution due to a lucky coincidence. We came in contact with a team located in the next building over from us, in charge of test automation for an internal website. This team was trying to get through automation of about 7,000 tests, and they were falling behind. Since they were desperate, and were not under the confines of a client contract, they decided to try out ATA. This allowed us to collect some citable data.[2] The actual data is not important here, and possibly had caveats, but it corroborated the claims of higher productivity as well as script resilience. We tried to offer this team as good "customer service" as we possibly could, which came in handy later when we asked them to be our reference for others.

The obvious but crucial third lesson is *your users, particularly the early adopters, are precious and should be treated as such*, because their referral is crucial in opening more doors.

Over time, we realized trying to change the ways of existing projects might be a fruitless initiative. It might be better to approach the sales side of the business, and get ATA worked into the deal right from the start. This turned out to be a good idea. Sales people liked flaunting the unique technology in test automation that IBM Research had to offer. On our part, we enjoyed getting to talk to higher-level decision makers from client side—these would often be executive-level people in CIO teams of major corporations—as opposed to just delivery managers on the IBM side. Once salespeople promised the use of ATA to the clients, the delivery managers had no choice but to comply. The result: better traction!

The fourth lesson then is that *tech transfer is subject to organizational dynamics*, and sometimes a top-down approach might be more appropriate than a bottom-up push.

Getting client teams interested turned out to be only part of the battle. There was significant work involved in customizing ATA to suit a client's needs. Since the automation tool is only one part of the overall workflow, we needed to ensure ATA interoperates with any third-party quality management infrastructure (such as Quality Center[1]) that the client uses in their organization. We also found ourselves under a lot of pressure due to the fact people could be vocal about their wish list from ATA, where they would have quietly put up with the limitations of off-the-shelf software! Notwithstanding the unique capabilities of ATA that differentiated it from competitor product offerings, users were not averse to comparing ATA with these other tools in terms of feature completeness. Addressing this partly involved managing user expectations of a tool that was, in essence, a research prototype rather than a product.

One of the recurring issues in client acceptance was that of tool usability. ATA was, by design, a tool for non- or semi-expert users. As such, we had to pay significant attention to making the tool behave well under all sorts of usage, sometimes even comically inept ones. Failure to anticipate such tool abuses resulted in escalations, and with it the risk of creating a bad image for the technology.

The final lesson then is the *"last mile" is a deeply flawed metaphor* when used in the context of tech transfer of software tools. In reality, a research prototype is just the first mile; everything after that is the work needed to make the technology work in real-world scenarios, usable by the target audience, and perhaps most importantly, to establish a positive value proposition for it. This requires patience and a long-term commitment on the part of researchers who wish to carry out a successful tech transfer.

References

1. Quality Center Enterprise, HP; http://www8.hp.com/us/en/software-solutions/quality-center-quality-management/
2. Thummalapenta, S., Devaki, P., Sinha, S., Chandra, S., Gnanasundaram, S., Nagaraj, D., and Sathishkumar, S. Efficient and change-resilient test automation: An industry case study. In *Proceedings of the International Conference on Software Engineering* (Software Engineering in Practice), 2013.
3. Thummalapenta, S., Sinha, S., Singhania, N., and Chandra, S. Automating test automation. In *Proceedings of the International Conference on Software Engineering*, 2012.

Satish Chandra (schandra@acm.org) is Senior Principal Engineer at Samsung Research America, Mountain View, CA.

Suresh Thummalapenta (suthumma@microsoft.com) is a member of the Tools for Software Engineering department at Microsoft Corporation, Redmond, WA.

Saurabh Sinha (sinhas@us.ibm.com) is a member of the Programming Technologies department at the IBM T.J. Watson Research Center, Yorktown Heights, NY.

This Viewpoint is based on an invited talk Satish Chandra presented at the 22nd ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE 2014). The work described here was carried out at IBM Research in Bangalore, India.

DOI:10.1145/2809777

Herbert Lin

Viewpoint
Having a Conversation about Bulk Surveillance

Considering a controversial subject that extends far beyond the collection of phone metadata.

Bulk collection of signals intelligence (bulk surveillance, for short) is a controversial topic. The most well known program to collect signals intelligence "in bulk" is the bulk collection of telephone metadata authorized by the Foreign Intelligence Surveillance Court under Section 215 of the Patriot Act. Opponents of the program contended the program is an unwarranted invasion of privacy, a general search of exactly the kind prohibited by the 4th Amendment to the U.S. Constitution. Supporters of the program asserted it was and is a critical tool in the nation's counterterrorist arsenal.

On May 8, 2015, the U.S. Court of Appeals for the Second Circuit held the language of this section could not plausibly be interpreted as authorizing bulk surveillance, though it did not rule on whether the program would be constitutional even with statutory justification. On June 1, Section 215 authority for this program expired, and on June 2, a new program was enacted into law under the USA Freedom Act requiring the metadata be held by phone companies. The National Security Agency lost the ability to query this metadata broadly based on prior arrangements from the Foreign Intelligence Surveillance Court, but continued to have access to specific numbers with an appropriate order issued by the Court.

Protesters at a rally against mass surveillance in Washington, D.C.

PHOTO BY RENA SCHILD/SHUTTERSTOCK.COM

The new program was hailed by civil liberties advocates as a step forward, but it is pretty clear it is only the first step in a broader debate over policy regarding bulk collection of signals intelligence. Regardless of where one stands on the issue, a reasoned discussion has to start with clarity about terms such as "bulk," "collection," and "signals intelligence." (To the best of my knowledge, the word "of" has not engendered much controversy.)

Broadly speaking, signals intelligence refers to information contained in electronic signals used by foreign targets of interest.[a] Thus, the debate over bulk surveillance properly extends far

[a] The National Security Agency definition of signals intelligence is "intelligence derived from electronic signals and systems used by foreign targets, such as communications systems, radars, and weapons systems." See http://www.nsa.gov/sigint.

beyond the phone metadata program.

Bulk collection is defined in presidential policy directive (PPD-28) as "the authorized collection of large quantities of signals intelligence (SIGINT) data which, due to technical or operational considerations, is acquired without the use of discriminants (e.g., specific identifiers, selection terms, etc.)"[b] But the directive does not define "discriminant," and so there is no precise definition of bulk (or targeted) collection.

Under this definition, all signals intelligence associated with communications in, for example, Syria, would be regarded as "targeted," simply because the selector "Syria" was used to separate Syrian traffic from other traffic. And signals intelligence associated with a communications channel linking only two individuals would be regarded as "bulk," simply because all (two) individuals in using that channel were being monitored.

What is the commonsense meaning of the term "bulk" collection? A recent National Research Council (NRC) study on Presidential Policy Directive 28,[c] in which I participated as a staffer, argued that "if a significant portion of the data collected is not associated with current targets, it is bulk collection; otherwise, it is targeted." The study went on to note "there is no precise definition of bulk collection, but rather a continuum, with no bright line separating bulk from targeted [collection]." The report acknowledges the term "significant" is itself imprecise, but in my view "significant" is at least amenable to quantification, and in principle one could argue as to whether "significant" should mean 10%, 90%, or some number in between.

A third issue involves the term "collection." In the context of modern electronic communications, at least two conceptually different definitions are possible. Under one definition (call it Definition A), collection has occurred when the electronic signal of a communication is first captured. Under a second definition (Definition B) that is particularly important in an environment in which different communications streams are interleaved on the same physical channel, collection has occurred only when the signal has been processed to determine whether it is relevant to the purpose of the collection. In this environment, every signal must be examined to know which ones are irrelevant.

The proper measure is not whether a program is necessary but rather whether it is helpful.

The NRC study elaborated Definition B as encompassing three steps: extraction of data into discrete data items from a communications channel, filtering the extracted data for items of interest (as might be indicated, for example, by the use of a discriminant; if all data items are of interest, no discriminant is used); and storage of these items in a database controlled by the cognizant U.S. government authority. (Under this definition, the new program does not call for government collection of phone metadata.)

Concerns about bulk surveillance raise objections that can be lumped into three general categories—its legal propriety, its efficacy in the overall counterterrorism effort, and its policy implications.

Regarding legal propriety, first consider domestic U.S. law. Foreigners do not have protection against surveillance in any form, except that foreigners in the U.S. are presumed to be U.S. persons unless there is specific evidence to the contrary. Executive order (specifically EO 12333) governs the handling of communications involving U.S. persons related to a legitimate foreign intelligence investigation, and a number of analysts assert the scope of information gathered regarding such communications is vast, including both content and metadata for a wide variety of communications modalities, including phone, email, chat rooms, instant messages and so on.[d] Within scope, for example, are communications between two U.S. persons that happen to be routed outside U.S. borders. However, Congressional oversight over activities conducted under EO 12333 is minimal.

As for international law, Article 17 of the International Covenant on Civil and Political Rights states that "No one shall be subjected to arbitrary or unlawful interference with his privacy, … or correspondence," and some have argued that U.S. bulk surveillance violates this provision.[e] However, this argument ignores Article 2(1) of the Covenant, which calls for parties to the Covenant "to respect and to ensure *to all individuals within its territory and subject to its jurisdiction* the rights recognized in the present Covenant" (emphasis added). Thus, the U.S., as well as most analysts, have taken the view the Covenant imposes duties on a nation only with respect to activities involving individuals within its territory and subject to its jurisdiction—and does not impose duties on that nation with respect to activities involving activities outside its territory.

Regarding the efficacy of bulk surveillance, some opponents have argued the bulk collection of phone metadata authorized under Section 215 has been of minimal value because there has not been a single terrorist plot that would have happened but for information derived from the program.[f] That is, the Section 215 program has not been *necessary* for thwarting any terrorist plot. But as I have argued elsewhere,[g] the proper measure is not whether a program is necessary but rather whether it is helpful. For most intelligence analysts, redundant information is helpful in corroborating what is already known and increases confidence in the accuracy of a judgment, as, for example, in identifying potential targets of interest or in ruling out targets.

Along these lines, the PCLOB noted that "any particular technique or legal authority can [only rarely] be identified as the key component without which a terrorist plot would

[b] http://1.usa.gov/1MUm5Yz
[c] http://bit.ly/15fzKbx
[d] http://wapo.st/Ug0kLS; http://bit.ly/1Y69CEt
[e] http://bit.ly/1IY3Gqi
[f] See, for example, http://bit.ly/1SSq8ea
[g] http://bit.ly/1U9dNza

have succeeded. Intelligence-gathering tools can provide value in more indirect ways, by helping to advance investigations and focus efforts in ways that are sometimes more difficult to measure."[h] Although the PCLOB went on to find no evidence the Section 215 program has made any significant contribution to counterterrorism efforts to date, this conclusion cannot be taken as an indictment of all possible bulk surveillance programs, each of which would have to be examined on its own merits for the benefits that it had provided or could be expected to provide in the future.

Others have argued that bulk surveillance results in information overload that makes it more difficult for analysts to find the information they do need.[i] That is, they argue a needle is more difficult to find in a big haystack than in a smaller one. Implicit in this argument is the claim the needle *does* exist in the haystack, and thus smarter analysis will be more helpful than adding more hay (information). But if the needle is not in the haystack, only adding more hay has even a chance of resulting in a successful needle discovery—and this is true despite the undeniable fact the additional data may place a greater burden on analysts and may still fail in the end to provide the necessary data. And until the needle is found, it is difficult to decide what information will turn out to be unnecessary before the analysis is complete.

Bulk surveillance is also useful for understanding events that have occurred in the past.[j] It can easily happen that information collected on Day X alerts analysts to the importance of a certain event A that occurred before Day X. Under such circumstances, it would only have been a matter of luck that any targeted surveillance operating before Day X would provide information about A, because A was not known before Day X to be important.

Finally, many policy implications of bulk surveillance remain to be addressed. How and to what extent, if any, should safeguarding the privacy of foreigners be relevant to U.S. collection of intelligence for national security purposes? Through PPD-28, President Obama granted foreigners certain privacy rights regarding information gathered on them through bulk surveillance. Should this step be rolled back, be the first step in treating foreigners and Americans alike, or be the last step along this path?

Many policy implications of bulk surveillance remain to be addressed.

A second issue is the scope of bulk surveillance. As noted, Section 215 authority has been used to justify bulk surveillance on domestic telephone metadata. But in principle, bulk surveillance could apply to communications modalities apart from telephone and to all kinds of data (indeed, the line between data and metadata—perhaps well-defined in an era of plain old telephone service—may well be blurry with other modalities). How far are we willing to go along these lines?

Lastly, how should the U.S. balance the intelligence value of capabilities provided by bulk surveillance against its costs? As noted earlier, bulk surveillance does have some value for the intelligence community. But as the PCLOB noted, "an intelligence-gathering tool with significant ramifications for privacy and civil liberties cannot be regarded as justified merely because it provides *some* value in protecting the nation from terrorism" (emphasis in the original). Nor, I might add, for other purposes as well. If any such tool is to be used to protect the public, the value it provides must be sufficient to outweigh its costs, which including financial, operational, and opportunity costs as well as costs to privacy and civil liberties.

The Section 215 debate pitted a tool (bulk surveillance over domestic phone metadata) that may be very helpful in preventing some serious terrorism incident against one that may be used to harass legal protestors of government policy—but there is no evidence that either has happened. In the absence of evidence, how should value and cost be determined?

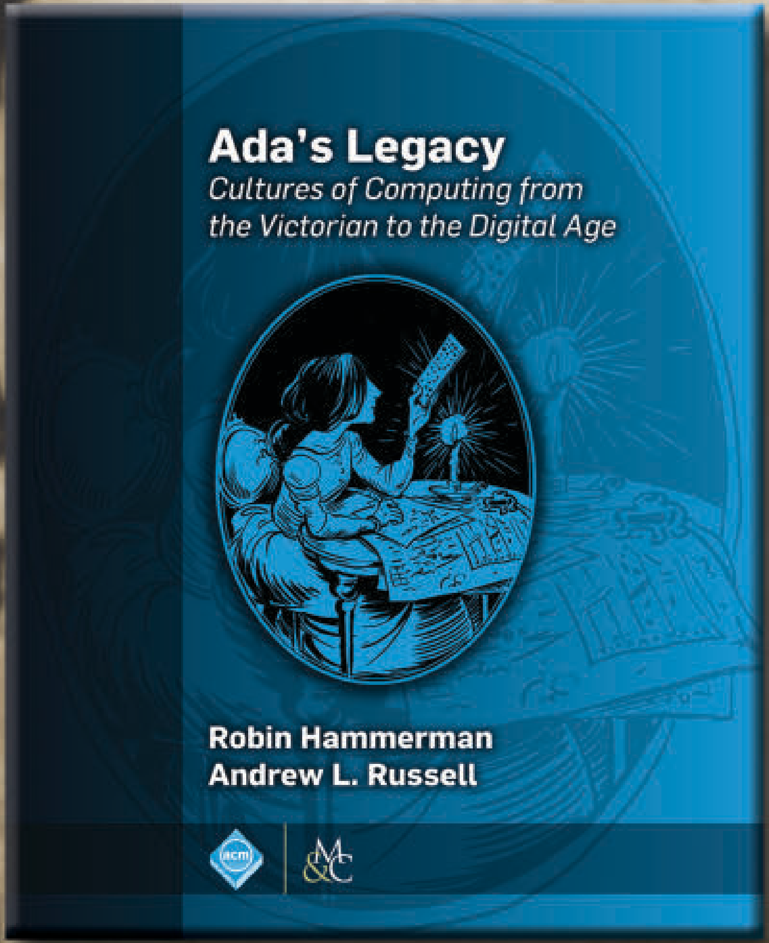
Many factors enter into any such determination, but perhaps the most important factor is the reality that the determination is a strong function of the circumstances extant at any given time. As much as some would like it to be otherwise, a serious national security incident inevitably results in greater concerns for security and lesser concerns about privacy and civil liberties. Tools that policy makers see as providing marginal value and entailing high costs before the incident may well be seen as providing higher value and entailing lower costs afterward.

Does such a shift in perspective ever result in overreaction? Certainly. In the light of history, the internment of U.S. citizens of Japanese origin after the Pearl Harbor attack is widely acknowledged as being an overreaction. And the passage of the USA Freedom Act may indicate the beginnings of a similar realization, although the Paris attacks of November 2015 and the San Bernardino shootings of December 2015 cast such a realization in a different light. Time will tell how the U.S. Congress decides to act on all of these matters, and the voices of computing professionals can help inform their future actions.

Herbert Lin (herblin@stanford.edu) is senior research scholar for cyber policy and security at the Stanford Center for International Security and Policy and research fellow at Stanford's Hoover Institution.

Although this Viewpoint is informed by the author's work on the NRC report mentioned, any differences between statements in this Viewpoint and the NRC report should be attributed solely to the author.

[h] http://bit.ly/1SRiPke
[i] http://zd.net/1GzGnRs
[j] Chapter 4 of the NRC report describes a variety of applications for bulk surveillance.


Ada's Legacy
Cultures of Computing from the Victorian to the Digital Age
Robin Hammerman
Andrew L. Russell

DOI:10.1145/2844544

Article development led by acmqueue
queue.acm.org

Big data makes common schemas even more necessary.

BY R.V. GUHA, DAN BRICKLEY, AND STEVE MACBETH

Schema.org: Evolution of Structured Data on the Web

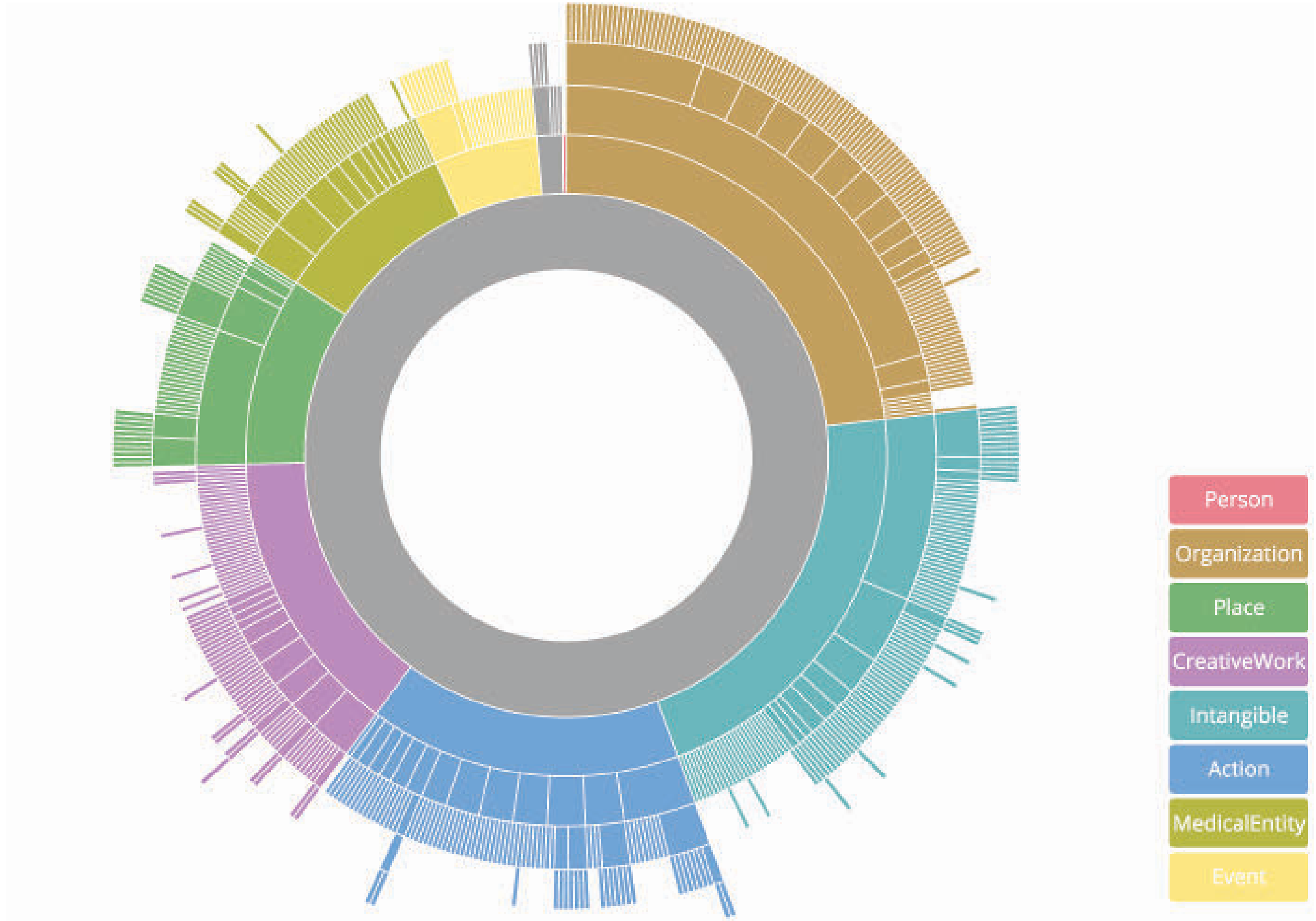
SEPARATION BETWEEN CONTENT and presentation has always been one of the important design aspects of the Web. Historically, however, even though most websites were driven off structured databases, they published their content purely in HTML. Services such as Web search, price comparison, reservation engines, among others that operated on this content had access only to HTML. Applications requiring access to the structured data underlying these Web pages had to build custom extractors to convert plain HTML into structured data. These efforts were often laborious and the scrapers were fragile and error prone, breaking every time a site changed its layout.

Recent proliferation of devices with widely varying form factors has dramatically increased the number of different presentation formats that websites must target. At the same time, a number of new personal assistant applications such as Google App and Microsoft's Cortana have started providing sites with new channels for reaching their users. Further, mature Web applications such as Web search are increasingly seeking to use the structured content, if any, to power richer and more interactive experiences. These developments have finally made it vital for both Web and application developers to be able to exchange their structured data in an interoperable fashion.

This article traces the history of efforts to enable Web-scale exchange of structured data and reports on Schema.org, a set of vocabularies based on existing standard syntax, in widespread use today by both publishers and consumers of structured data on the Web. Examples illustrate how easy it is to publish this data and some of the ways in which applications use this data to deliver value to both users and publishers of the data.

Early on it became clear that domain-independent standards for structured data would be very useful. One approach—XML—attempted to standardize the syntax. While XML was initially thought of as the future of browser-based HTML, it has found more utility for structured data, with more traditional data-interoperability scenarios.

Another approach—MCF[18] (Meta Content Framework)—introduced ideas from knowledge representation (frames and semantic nets) to the Web and proposed going further by using a common data model—namely, a directed labeled graph. Its vision was to create a single graph (or knowledge base) about a wide range of entities, different parts of which would come from different sites. An early diagram of this vision is shown in Figure 1, in which information about Tori Amos is pulled together from different sites of



An interactive version of the Starburst visualization (http://blog.schema.org/) allows for exploring Schema.org's hierarchy.

IMAGE BY FABIO VALSECCHI

that era into a single coherent graph.

The hope at that time was to enable many different applications to work easily with data from many different sites. Over time, the vision grew to cover all kinds of intelligent processing of data on the Web. A 2001 *Scientific American* article by Tim Berners-Lee et al. on the Semantic Web was probably the most ambitious and optimistic view of this program.[5]

Between 1997 and 2004 various standards (RDF, RDFS, and OWL) were developed for the syntax and data model. A number of vocabularies were proposed for specific verticals, some of which were widely adopted. One of these was RSS (Rich Site Summary), which allowed users to customize home pages such as Netscape's Netcenter and Yahoo's My Yahoo with their favorite news sources. Another was vCard/hCard (such as, IMC's vCard standard, expressed in HTML using microformat via the CSS class attribute), which was used to exchange contact information between contact managers, email programs, and so on. These were later joined by hCalendar, a format for calendar exchange, again a microformats HTML re-expression of an existing IETF (Internet Engineering Task Force) standard, iCalendar. FOAF (Friend of a Friend) predated these efforts but saw its usage for social-network data decline as that industry matured.[11] It has found a niche in the RDF (Resource Description Framework) Linked Data community as a commonly reused schema.[6]

In each of these cases where structured data was being published, one class of widely used application consumed it. Since the goal was to create a graph with wide coverage, well beyond narrow verticals, the challenge was to find a widely used application that had broad coverage. This application turned out to be text search.

The intense competition in Web search led companies to look beyond the ranking of results to improve search results. One technique used first by Yahoo and then Google was to augment the snippet associated with each search result with structured data from the results page.

They focused on a small number of verticals (eventually around 10, such as recipes, events, among others), each with a prescribed vocabulary, reusing existing vocabularies such as hCard and FOAF when appropriate. For each, they augmented the snippet with some structured data so as to optimize the user's and webmaster's experience. This approach led to much greater adoption, and soon a few hundred thousand sites were marking up their pages with structured data markup. The program had a substantial drawback, however. The vocabularies for the different verticals were completely independent, leading to substantial duplication and confusion. It was clear that extending this to hundreds or thousands of verticals/classes was impossible. To make things worse, different search engines recommended different vocabularies.

Because of the resulting confusion, most webmasters simply did not add any markup, and the markup they did add was often incorrectly formatted. This abundance of incorrect formatting required consumers of markup to build complex parsers that were able to handle improperly formed syntax and vocabulary. These complex parsers turned out to be just as brittle as the original systems used to extract structured data from HTML and thus did not result in the expected advances.

Schema.org

In 2011, the major search engines Bing, Google, and Yahoo (later joined by Yandex) created Schema.org to improve this situation. The goal was to provide a single schema across a wide range of topics that included people, places, events, products, offers, and so on. A single integrated schema covered these topics. The idea was to present webmasters with a single vocabulary. Different search engines might use the markup differently, but webmasters had to do the work only once and would reap the benefits across multiple consumers of the markup.

Figure 1. Example of a knowledge base sourced from multiple sites.

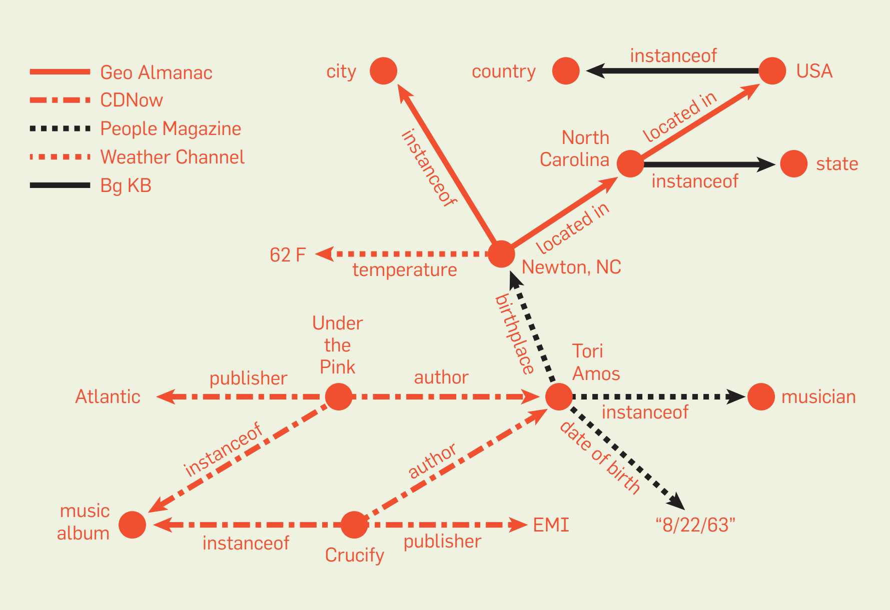


Figure 2. Restaurant reservation email markup (microdata syntax).

```
<p itemscope
itemtype=”http://schema.org/FoodEstablishmentReservation”>
     Your reservation for <span itemprop=”partySize”>3</span>
     at Local Edition is
     <link itemprop=”reservationStatus”
     href=”http://schema.org/Confirmed”/>confirmed</link> for
     <timeitemprop=”startTime” datetime=”2015-05-02T18:30:00Z”>May
2nd,2015 at 6:30 PM</time>.
     The reservation is held under:
<span itemscope itemtype=”http://schema.org/Person”>
     <span itemprop=”givenName”>Dan Brickley</span>.
     </span>
Serve yourself when you arrive.
<span itemscope itemtype=”http://schema.org/Restaurant”>
     <meta itemprop=”telephone” content=”tel:+1-202-555-0125” />
     To get there:<br />
     <span itemprop=”name”> Local Edition </span><br />
     <span itemprop=”address” itemscope itemtype=”http://schema.org/Posta-
lAddress”>
           <span itemprop=”streetAddress”>2370 South Market Street, San
Francisco, USA.</span>
     </span>
</span>
</p>
```

Schema.org was launched with 297 classes and 187 relations, which over the past five years have grown to 638 classes and 965 relations. The classes are organized into a hierarchy, where each class may have one or more superclasses (though most have only one). Relations are polymorphic in the sense they have one or more domains and one or more ranges. The class hierarchy is meant more as an organizational tool to help browse the vocabulary than as a representation of common sense, à la Cyc.

The first application to use this markup was Google's Rich Snippets, which switched over to Schema.org vocabulary in 2011. Over the past four years, a number of different applications across many different companies have started using Schema.org vocabulary. Some of the more prominent among these include the following:

- In addition to per-link Rich Snippets, annotations in Schema.org are used as a data source for the Knowledge Graph, providing background information about well-known entities (for example, logo, contact, and social information).
- Schema.org-based structured data markup is now being used in places such as email messages.

For example, emailmessages confirming reservations (restaurant, hotel, airline, and so on), purchase receipts, have embedded Schema.org markup with details of the transaction. This approach makes it possible for email assistant tools to extract the structured data and make it available through mobile notifications, maps, and calendars. Google's Gmail and Search products use this data to provide notifications and reminders (Figure 2). For example, a dinner booking made on Opentable.com will trigger a reminder for leaving for the restaurant, based on the location of the restaurant, the user, traffic conditions, and so on.

- Microsoft's Cortana (for Windows 10 and Windows phones) makes use of Schema.org from email messages, as shown in Figure 3.
- Yandex uses many parts of Schema.org, including recipes, autos, reviews, organizations, services, and directories. Its earlier use of FOAF

(corresponding to the popularity of the LiveJournal social network in Russia) demonstrated the need for pragmatic vocabulary extensions that support consumer-facing product features.

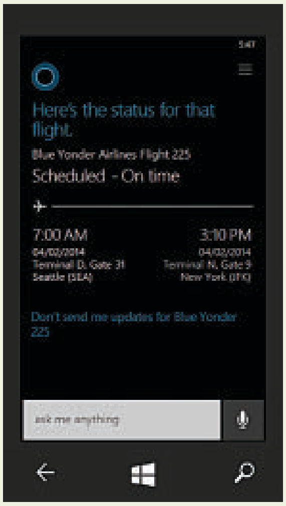
- Pinterest uses Schema.org to provide rich pins for recipe, movie, article, product, or place items.
- Apple's iOS 9 (Searchlight/Siri) uses Schema.org for search features including aggregate ratings, offers, products, prices, interaction counts, organizations, images, phone numbers, and potential website search actions. Apple also uses Schema.org within RSS for news markup.

Adoption Statistics

The key measure of success is, of course, the level of adoption by webmasters. A sample of 10 billion pages from a combination of the Google index and Web Data Commons provides some key metrics. In this sample 31.3% of pages have Schema.org markup, up from 22% one year ago. On average, each page containing this markup makes references to six entities, making 26 logical assertions among them. Figure 4a lists well-known sites within some of the major verticals covered by Schema.org, showing both the wide range of topics covered and the adoption by the most popular sites in each of these topics. Figures 4b and 4c list some of the most frequently used types and relations. Extrapolating from the numbers in this sample, we estimate at least 12 million sites use Schema.org markup. The important point to note is structured data markup is now of the same order of magnitude as the Web itself.

Although this article does not present a full analysis and comparison, we should emphasize various other formats are also widespread on the Web. In particular, OGP (Open Graph Protocol) and microformat approaches can be found on approximately as many sites as Schema.org, but given their much smaller vocabularies, they appear on fewer than half as many pages and contain fewer than a quarter as many logical assertions. At this point, Schema.org is the only broad vocabulary used by more than one-quarter of the pages found in the major search indices.

Figure 3. Flight reservation email markup (JSON-LD syntax) and its use in Microsoft's Cortana.



Here's the status for that flight.

Blue Yonder Airlines Flight 225

Scheduled - On time

7:00 AM	3:10 PM
04/02/2014	04/02/2014
Terminal D, Gate 31	Terminal N, Gate 9
Seattle (SEA)	New York (JFK)

Don't send me updates for Blue Yonder 225

```
{ “@context”: “http://schema.org/”,
  “@type”: “FlightReservation”,
  “reservationNumber”: “QWERT0123456789”,
  “reservationStatus”:
“http://schema.org/Confirmed”,
  “underName”:{
    “@type”: “Person”,
    “name”: “Estella Gallagher”
  },
  “reservationFor”: {
    “@type”: “Flight”,
    “flightNumber”: “123”,
    “departureAirport”: {
      “@type”: “Airport”,
      “name”: “ Seattle-Tacoma International
Airport”,
      “iataCode”: “SEA”
    },
    “arrivalAirport”: {
      “@type”: “Airport”,
      “name”: “ John F Kennedy International
Airport”,
      “iataCode”: “JFK”
     },
 “departureTime”: “2014-04-02T10:32:00Z”,
    “arrivalTime”: “2014-04-02T11:45:00Z”,
    “airline”: {
      “@type”: “Airline”,
      “name”: “Blue Yonder Airlines”,
      “iataCode”: “BY”
    }
  }
}
```

Figure 4. (a) Major sites that have published Schema.org, (b) Most frequently used types (from public Web), (c) Most frequently used properties (as of July 2015).

(a)

Category	Sites
News	nytimes.com, guardian.com, bbc.co.uk
Movies	imdb.com, rottentomatoes.com, movies.com
Jobs / Careers	careerjet.com, monster.com, indeed.com
People	linkedin.com, pinterest.com, familysearch.org, archives.com
Products	ebay.com, alibaba.com, sears.com, cafepress.com, sulit.com, fotolia.com
Video	youtube.com, dailymotion.com, frequency.com, vinebox.com
Medical	cvs.com, drugs.com
Local	yelp.com, allmenus.com, urbanspoon.com
Events	wherevent.com, meetup.com, zillow.com, eventful.com
Music	last.fm, myspace.com, soundcloud.com

(b)

WebSite, SearchAction, WebPage, Product, ImageObject, Person, Offer, BlogPosting, Organization, Article, PostalAddress, Blog, LocalBusiness, AggregateRating, WPFooter, SiteNavigationElement, WPHeader, WPSideBar, CreativeWork, Review, EntryPoint, ViewAction, Place, Rating, ItemList, Event, ListItem, VideoObject, GeoCoordinates, Thing, SocialMediaPosting, UserComments, ProfilePage, Restaurant, Brand, OpeningHoursSpecification, CollectionPage, Recipe, QuantitativeValue, RealEstateAgent, NewsArticle, ItemPage, JobPosting, MusicGroup, ImageGallery, MusicRecording, WPAdBlock, Store

(c)

name, url, description, image, target, query-input, potentialAction, datePublished, author, articleBody, null, price, offers, contentURL, address, telephone, addressLocality, priceCurrency, availability, streetAddress, headline, postalCode, thumbnailUrl, addressRegion, ratingValue, mainContentOfPage, blogPost, aggregateRating, text, logo, sku, postId, blogId, image_url, bestRating, inLanguage, reviewCount, breadcrumb, email, urlTemplate, keywords, ratingCount, addressCountry, reviewRating, itemListElement, sameAs, openingHours, position, location, worstRating, startDate

A key driver of this level of adoption is the extensive support from third-party tools such as Drupal and Wordpress extensions. In verticals (such as events), support from vertical-specific content-management systems (such as Bandsintown and Ticketmaster) has had a substantial impact. A similar phenomenon was observed with the adoption of RSS, where the number of RSS feeds increased dramatically as soon as tools such as Blogger started outputting RSS automatically.

The success of Schema.org is attributable in large part to the search engines and tools rallying behind it. Not every standard pushed by big companies has succeeded, however. Some of the reason for Schema.org's success lies with the design decisions underlying it.

Design Decisions

The driving factor in the design of Schema.org was to make it easy for webmasters to publish their data. In general, the design decisions place more of the burden on consumers of the markup. This section addresses some of the more significant design decisions.

Syntax. From the beginning, Schema.org has tried to find a balance between pragmatically accepting several syntaxes versus making a clear and simple recommendation to webmasters. Over time it became clear multiple syntaxes would be the best approach. Among these are RDFa (Resource Description Framework in Attributes) and JSON-LD (JavaScript Object Notation for Linked Data), and publishers have their own reasons for preferring one over another.

In fact, in order to deal with the complexity of RDFa 1.0, Schema.org promoted a newer syntax, Microdata that was developed as part of HTML5. Design choices for Microdata were made through rigorous usability testing on webmasters. Since then, prompted in part by Microdata, revisions to RDFa have made it less complex, particularly for publishers.

Different syntaxes are appropriate for different tools and authoring models. For example, Schema.org recently endorsed JSON-LD, where the structured data is represented as a set of JavaScript-style objects. This works well for sites that are generated using client-side JavaScript as well as in personalized email where the data structures can be significantly more verbose. There are a small number of content-management systems for events (such as concerts) that provide widgets that are embedded into other sites. JSON-LD allows these embedded widgets to carry structured data in Schema.org. In contrast, Microdata and RDFa often work better for sites generated using server-side templates.

It can sometimes help to idealize this situation as a trade-off between machine-friendly and human-friendly formats, although in practice the relationship is subtler. Formats such as RDF and XML were designed primarily with machine consumption in mind, whereas microformats have a stated bias toward humans first. Schema.org is exploring the middle ground, where some machine-consumption convenience is traded for publisher usability.

Polymorphism. Many frame-based KR (knowledge representation) systems, including RDF Schema and OWL (Web Ontology Language) have a single domain and range for each relation. This, unfortunately, leads to many unintuitive classes whose only role is to be the domain or range of some relation. This also makes it much more difficult to reuse existing relations without significantly changing the class hierarchy. The decision to allow multiple domains and ranges seems to have significantly ameliorated the problem. For example, though there are various types (Events, Reservations, Offers) in Schema.org whose instance can take a startDate property, the polymorphism has allowed us to get away with not having a common supertype (such as TemporallyCommencable-Activity) in which to group these.

Entity references. Many models such as Linked Data have globally unique URIs for every entity as a core architectural principle.[4] Unfortunately, coordinating entity references with other sites for the tens of thousands of entities about which a site may have information is much too difficult for most sites. Instead, Schema.org insists on unique URIs for only the very small number of terms provided by Schema.org. Publishers are encouraged to add as much extra description to each entity as possible so that consumers of the data can use this description to do entity reconciliation. While this puts a substantial additional burden on applications consuming data from multiple websites, it eases the burden on webmasters significantly. In the example shown in Figure 1, instead of requiring common URIs for the entities (for example, Tori Amos; Newton, NC; and Crucify), of which there are many hundreds of millions (with any particular site using potentially hundreds of thousands), webmasters must use standard vocabulary only for terms such as *country*, *musician*, *date of birth*, and so on of which there are only a few thousand (with any particular site using at most a few dozen). Schema.org does, however, also provide a sameAs property that can be used to associate entities with well-known pages (home pages, Wikipedia, and so on) to aid in reconciliation, but this has not found much adoption.

Incremental complexity. Often, making the representation too simplistic would make it hard to build some of the more sophisticated applications. In such cases, we start with something simple, which is easy for webmasters to implement, but has enough data to build a motivating application. Typically, once the simple applications are built and the vocabulary gets a minimal level of adoption, the application builders and webmasters demand a more expressive vocabulary—one that might have been deemed too complex had we started off with it.

At this point, it is possible to add the complexity of a more expressive vocabulary. Often this amounts to the relatively simple matter of adding a few more descriptive properties or subtypes. For example, adding new types of actions or events is a powerful way of extending Schema.org's expressivity. In many situations, however, closer examination reveals subtle differences in conceptualization. For example, creative works have many different frameworks for analyzing seemingly simple concepts, such as *book*, into typed, interrelated entities (for example, in the library world, functional requirements for bibliographic records, or FRBR); or with e-commerce *offers*, some systems distinguish manufacturer warranties from vendor warranties. In such situations there is rarely a right answer. The Schema.org approach is to be led by practicalities—the data fields available in the wider Web and the information requirements of applications that can motivate large-scale publication. Schema.org definitions are never changed in pursuit of the perfect model, but rather in response to feedback from publishers and consumers.

Schema.org's incremental complexity approach can be seen in the interplay among evolving areas of the schema. The project has tried to find a balance between two extremes: uncoordinated addition of schemas with overlapping scopes versus overly heavy coordination of all topics. As an example of an area where we have stepped back from forced coordination, both creative works (books, among others) and e-commerce (product descriptions) wrestle with the challenge of describing versions and instances of various kinds of mass-produced items. In professional bibliographies, it is important to describe items at various levels (for example, a particular author-signed copy of a particular paperback versus the work itself, or the characteristics of that edition such as publisher details). Surprisingly similar distinctions must be made in e-commerce when describing nonbibliographic items such as laser printers. Although it was intellectually appealing to seek schemas that capture a "grand theory of mass produced items and their common properties," Schema.org instead took the pragmatic route and adopted different modeling idioms for bibliography[12] and e-commerce.[8]

It was a pleasant surprise, by contrast, to find unexpected common ground between those same fields when it was pointed out that Schema.org's concept of an *offer* could be applied in not-for-profit fields beyond e-commerce, such as library lending. A few community-proposed pragmatic adjustments to our definitions were needed to clarify that offers are often made without expectation of payment. This is typical of our approach, which is to publish schemas early in the full knowledge they will need improving, rather than to attempt to perfect everything prior to launch. As with many aspects of Schema.org, this is also a balancing act: given strong incentives from consumers, terms can go from nothing to being used on millions of sites within a matter of months. This provides a natural corrective force to the desire to continue tweaking definitions; it is impractical (and perhaps impolite) to change schema definitions too much once they have started to gain adoption.

The driving factor in the design of Schema.org was to make it easy for webmasters to publish their data.

Cleanup. Every once in a while, we have gotten carried away and have introduced vocabulary that never gets meaningful usage. While it is easy to let such terms lie around, it is better to clean them out. Thus far, this has happened only with large vocabularies that did not have a strong motivating application.

Extensions

Given the variety of structured data underlying the Web, Schema.org can at best hope to provide the core for the most common topics. Even for a relatively common topic such as automobiles, potentially hundreds of attributes are required to capture the details of a car's specifications as found on a manufacturer's website. Schema.org's strategy has been to have a small core vocabulary for each such topic and rely on extensions to cover the tail of the specification.

From the beginning there have been two broad classes of extensions: those that are created by the Schema.org community with the goal of getting absorbed into the core, and those that are simply deployed "in the wild" without any central coordination. In 2015, the extension mechanism was enhanced to support both of these ideas better. First, the notion of *hosted* extensions was introduced; these are terms tightly integrated into Schema.org's core but treated as additional (in some sense optional) layers. Such terms still require coordination discussion with the broader community to ensure consistent naming and to identify appropriate integration points. The layering mechanism, however, is designed to allow greater decentralization to expert and specialist communities.

Second came the notion of *external* extensions. These are independently managed vocabularies that have been designed with particular reference to Schema.org's core vocabulary with the expectation of building upon, rather than duplicating, that core. External extensions may range from tiny vocabularies that are product/service-specific (for example, for a particular company's consumption), geographically specific (for example, U.S.-Healthcare), all the way to large schemas that are on a scale similar to Schema.org.

We have benefited from Schema.org's cross-domain data model. It has allowed a form of loosely coupled collaboration in which topic experts can collaborate in dedicated fora (for example, sports, health, bibliography), while doing so within a predictable framework for integrating their work with other areas of Schema.org.

The more significant additions have come from external groups that have specific interests and expertise in an area. Initially, such collaborations were in a project-to-project style, but more recently they have been conducted through individual engagement via W3C's Community Group mechanism and the collaboration platform provided by GitHub.

The earliest collaboration was with the IPTC's rNews initiative, whose contributions led to a number of term additions (for example, NewsArticle) and improvements to support the description of news. Other early additions include healthcare-related schemas, e-commerce via the inclusion of the GoodRelations project, as well as LRMI (Learning Resources Metadata Initiative), a collaboration with Creative Commons and the Association of Educational Publishers.

The case of TV and radio markup illustrates a typical flow, as well as the evolution of our collaborative tooling.[9] Schema.org began with some rough terminology for describing television content. Discussions at W3C identified several ways in which it could be improved, bringing it more closely in line with industry conventions and international terminology, as well as adding the ability to describe radio content. As became increasingly common, experts from the wider community (BBC, EBU, and others) took the lead in developing these refinements (at the time via W3C's wikis and shared file systems), which in turn inspired efforts to improve our collaboration framework. The subsequent migration to open source tooling hosted on GitHub in 2014 has made it possible to iterate more rapidly, as can be seen from the project's release log, which shows how the wider community's attention to detail is being reflected in fine-grained improvements to schema details.[10]

Schema.org does not mandate exactly how members of the wider community should share and debate ideas—beyond a general preference for public fora and civil discussion. Some groups prefer wikis and IRC (Internet Relay Chat); others prefer Office-style document collaborative authoring, telephones, and face-to-face meetings. Ultimately, all such efforts need to funnel into the project's public GitHub repository. A substantial number of contributors report problems or share proposals via the issue tracker. A smaller number of contributors, who wish to get involved with more of the technical details, contribute specific changes to schemas, examples, and documentation.

Related Efforts

Since 2006, the "Linked Data" slogan has served to redirect the W3C RDF community's emphasis from Semantic Web ontology and rule languages toward open-data activism and practical data sharing. Linked data began as an informal note from Tim Berners-Lee that critiqued the (MCF-inspired) FOAF approach of using reference by description instead of "URIs everywhere:"[3]

"This linking system was very successful, forming a growing social network, and dominating, in 2006, the linked data available on the Web. However, the system has the snag that it does not give URIs to people, and so basic links to them cannot be made."

Linked-data advocacy has successfully elicited significant amounts of RDF-expressed open data from a variety of public-sector and open-data sources (for example, in libraries,[14] the life sciences,[16] and government.[15] A strong emphasis on identifier reconciliation, complex best practice rules (including advanced use of HTTP), and use of an arbitrary number of partially overlapping schemas, however, have limited the growth of linked-data practices beyond fields employing professional information managers. Linked RDF data publication practices have not been adopted in the Web at large.

Schema.org's approach shares a lot with the linked-data community: it uses the same underlying data model and schema language,[17] and syntaxes (for example, JSON-LD and RDFa), and shares many of the same goals. Schema.org also shares the linked-data community's skepticism toward the premature formalism (rule systems, description logics, and so on) found in much of the academic work that is carried out under the Semantic Web banner. While Schema.org also avoids assuming that such rule-based processing will be commonplace, it differs from typical linked-data guidelines in its assumption that various other kinds of cleanup, reconciliation, and post-processing will usually be needed before structured data from the Web can be exploited in applications.

Linked data aims higher and has consequently brought to the Web a much smaller number of data sources whose quality is often nevertheless very high. This opens up many opportunities for combining the two approaches—for example, professionally published linked data can often authoritatively describe the entities mentioned in Schema.org descriptions from the wider mainstream Web.

Using unconstrained combinations of identifying URIs and unconstrained combinations of independent schemas, linked data can be seen as occupying one design extreme. A trend toward Google Knowledge Graphs can be viewed at the other extreme. This terminology was introduced in 2012 by Google, which presented the idea of a Knowledge Graph as a unified graph data set that can be used in search and related applications. In popular commentary, Google's (initially Freebase-based) Knowledge Graph is often conflated with the specifics of its visual presentation in Google's search results—typically as a simple factual panel. The terminology is seeing some wider adoption.

The general idea builds upon common elements shared with linked data and Schema.org: a graph data model of typed entities with named properties. The Knowledge Graph approach, at least in its Google manifestation, is distinguished in particular by a strong emphasis on up-front entity reconciliation, requiring curation discipline to ensure new data is carefully integrated and linked to existing records. Schema.org's approach can be seen as less noisy and decentralized than linked data, but more so than Knowledge Graphs. Because of the shared underlying approach, structured data expressed as Schema.org is a natural source of information for integration into Knowledge Graphs. Google documents some ways of doing so.[7]

Lessons

Here are some of the most important lessons we have learned thus far, some of which might be applicable to other standards efforts on the Web. Most are completely obvious but, interestingly, have been ignored on many occasions.

1. *Make it easy for publishers/developers to participate.* More generally, when there is an asymmetry in the number of publishers and the number of consumers, put the complexity with the smaller number. They have to be able to continue using their existing tools and workflows.

2. *No one reads long specifications.* Most developers tend to copy and edit examples. So, the documentation is more like a set of recipes and less like a specification.

3. *Complexity has to be added incrementally, over time.* Today, the average Web page is rather complex, with HTML, CSS, JavaScript. It started out being very simple, however, and the complexity was added mostly on an as-needed basis. Each layer of complexity in a platform/standard can be added only after adoption of more basic layers.

Conclusion

The idea of the Web infrastructure requiring structured data mechanisms to describe entities and relationships in the real world has been around for as long as the Web itself.[1,2,13] The idea of describing the world using networks of typed relationships was well known even in the 1970s, and the use of logical statements about the world has a history predating computing. What is surprising is just how difficult it was for such seemingly obvious ideas to find their way into the Web as an information platform. The history of Schema.org suggests that rather than seeking directly to create "languages for intelligent agents," addressing vastly simpler scenarios from Web search has turned out to be the best practical route toward structured data for artificial personal assistants.

Over the past four years, Schema.org has evolved in many ways, both organizationally and in terms of the actual schemas. It started with a couple of individuals who created an informal consortium of the three initial sponsor companies. In the first year, these sponsor companies made most decisions behind closed doors. It incrementally opened up, first moving most discussions to W3C public forums, and then to a model where all discussions and decision making are done in the open, with a steering committee that includes members from the sponsor companies, academia, and the W3C.

Four years after its launch, Schema.org is entering its next phase, with more of the vocabulary development taking place in a more distributed fashion. A number of extensions, for topics ranging from automobiles to product details, are already under way. In such a model, Schema.org itself is just the core, providing a unifying vocabulary and congregation forum as necessary.

The increased interest in big data makes the need for common schemas even more relevant. As data scientists are exploring the value of data-driven analysis, the need to pull together data from different sources and hence the need for shared vocabularies is increasing. We are hopeful that Schema.org will contribute to this.

Acknowledgments. Schema.org would not be what it is today without the collaborative efforts of the teams from Google, Microsoft, Yahoo and Yandex. It would also be unrecognizable without the contributions made by members of the wider community who have come together via W3C.

Related articles on queue.acm.org

Proving the Correctness of Nonblocking Data Structures
Mathieu Desnoyers
http://queue.acm.org/detail.cfm?id=2490873

Managing Semi-Structured Data
Daniela Florescu
http://queue.acm.org/detail.cfm?id=1103832

The Five-minute Rule: 20 Years Later and How Flash Memory Changes the Rules
Goetz Graefe
http://queue.acm.org/detail.cfm?id=1413264

References

1. Berners-Lee, T. Information management: a proposal; http://www.w3.org/History/1989/proposal.html.
2. Berners-Lee, T. W3 future directions, 1994; http://www.w3.org/Talks/WWW94Tim/.
3. Berners-Lee, T. Linked Data, 2006; http://www.w3.org/DesignIssues/LinkedData.html.
4. Berners-Lee, T. Is your linked open data 5 star? 2010; http://www.w3.org/DesignIssues/LinkedData#fivestar.
5. Berners-Lee, T., Hendler, J. and Lassila, O. The semantic web. *Scientific American* (May 2001), 29–37; http://www.scientificamerican.com/article/the-semantic-web/.
6. Friend of a Friend vocabulary (foaf); http://lov.okfn.org/dataset/lov/vocabs/foaf.
7. Google Developers. Customizing your Knowledge Graph, 2015; https://developers.google.com/structured-data/customize/overview.
8. Guha, R.V. Good Relations and Schema.org. Schema Blog; http://blog.schema.org/2012/11/good-relations-and-schemaorg.html.
9. Raimond, Y. Schema.org for TV and radio markup. Schema Blog; http://blog.schema.org/2013/12/schemaorg-for-tv-and-radio-markup.html.
10. Schema.org. Release log; http://schema.org/docs/releases.html.
11. Schofield, J. Let's be Friendsters. *The Guardian* (Feb. 19, 2004); http://www.theguardian.com/technology/2004/feb/19/newmedia.media.
12. Wallis, R., Scott, D. Schema.org support for bibliographic relationships and periodicals. Schema Blog; http://blog.schema.org/2014/09/schemaorg-support-for-bibliographic_2.html.
13. W3C. Describing and linking Web resources. Unpublished note, 1996; http://www.w3.org/Architecture/NOTE-link.html.
14. W3C. Library Linked Data Incubator Group Final Report, 2011; http://w3.org/2005/Incubator/lld/XGR-lld-20111025/.
15. W3C. Linked Data Cookbook; http://www.w3.org/2011/gld/wiki/Linked_Data_Cookbook.
16. W3C. Health Care and Life Science Linked Data Guide, 2012; http://www.w3.org/2001/sw/hcls/notes/hcls-rdf-guide/.
17. W3C. RDF Schema 1.1, 2014; http://www.w3.org/TR/rdf-schema/
18. W3C. MCF Using XML, R.V.Guha, T.Bray, 1997; http://w3.org/TR/NOTE-MCF-XML

R.V. Guha is a Google Fellow and a vice president in research at Google. He is the creator of Web standards such as RSS and Schema.org. He is also responsible for products such as Google Custom Search. He was a co-founder of Epinions.com and Alpiri and co-leader of the Cyc project.

Dan Brickley works at Google on the Schema.org initiative and structured-data standards. He is best known for his work on Web standards in the W3C community, where he helped create the Semantic Web project and many of its defining technologies. Previous work included metadata projects around TV, agriculture, DLs, and education.

Steve Macbeth is partner architect in the application and service group at Microsoft, where he is responsible for designing and building solutions at the intersection of mobile, cloud, and intelligent systems. Previously, he was a senior leader in the Bing Core Search, general manager and co-founder of the Search Technology Center Asia, and founder and CTO of Riptide Technologies and pcsupport.com.

DOI:10.1145/2844108

Article development led by acmqueue
queue.acm.org

A practitioner's guide to increasing confidence in system correctness.

BY CAITIE MCCAFFREY

The Verification of a Distributed System

LESLIE LAMPORT, KNOWN for his seminal work in distributed systems, famously said, "A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable." Given this bleak outlook and the large set of possible failures, how do you even begin to verify and validate the distributed systems you build are doing the right thing?

Distributed systems are difficult to build and test for two main reasons: partial failure and asynchrony. Asynchrony is the nondeterminism of ordering and timing within a system; essentially, there is no now.[10] Partial failure is the idea components can fail along the way, resulting in incomplete results or data.

These two realities of distributed systems must be addressed to create a correct system. Solving these problems often leads to solutions with a high degree of complexity. This in turn leads to an increase in the probability of human error in either design, implementation, or operation. In addition, the interaction of asynchrony and partial failure leads to an extremely large state space of failure conditions. Because of the complexity of distributed systems and the large state space of failure conditions, testing and verifying these systems are critically important. Without explicitly forcing a system to fail, it is unreasonable to have any confidence it will operate correctly in failure modes.

Formal verification is a set of methods that prove properties about a system. Once these methods have been used, a system gets a gold star for being provably correct.

Formal specification languages. Individual systems and protocols can be verified using formal methods such as TLA+ and Coq.[16] These are formal specification languages that allow users to design, model, document, and verify the correctness of concurrent systems. Every system under test requires the definition of the program, an environment including what kinds of failures can happen, and a correctness specification that defines the guarantees the system provides. With these tools a system can be declared provably correct.

Amazon Web Services (AWS) successfully used TLA+ to verify more than 10 core pieces of its infrastructure, including Simple Storage Service (S3).[8] This resulted in the discovery of several subtle bugs in the system design and allowed AWS to make aggressive performance optimizations without having to worry about breaking existing behavior.

The typical complaint about formal specification languages is that learning them is difficult and writing the specifications is tedious and time consuming. It is true that while they can uncover

IMAGE FROM SHUTTERSTOCK.COM

onerous bugs, there is a steep learning curve and a large time investment.

Model checking. Another formal method—model checking—can also determine if a system is provably correct. Model checkers use state-space exploration systematically to enumerate paths for a system. Once all paths have been executed, a system can be said to be correct. Examples of model checkers include Spin,[11] MoDIST,[14] TLC,[7] and MaceMC.[6]

Given the multitude of inputs and failure modes a system can experience, however, running an exhaustive model checker is incredibly time and resource consuming.

Composite. Some of the objections to and costs of formal verification methods could be overcome if provably correct components composed with one another to create provably correct systems. However, this is not the case. Often components are verified under different failure models, and these obviously do not compose. In addition, the correctness specifications for each component do not compose together to create a correctness specification for the resulting system.[1] To prove the resulting system is provably correct, a new correctness specification must be created and the tests must be rerun.

Creating a new correctness specification for each combination of components does not scale well, especially in the microservice-oriented architectures that have recently become popular. Such architectures consist of anywhere from tens to thousands of distinct services, and writing correctness specifications at this scale is not tenable.

Verification in the Wild

Given that formal verification is expensive and does not produce components that can be composed into provably correct systems without additional work, one may despair and claim the problem seems hopeless, distributed systems are awful, and we cannot have nice things.

However, all is not lost! You *can* employ testing methods that greatly increase your confidence the systems you build are correct. While these methods do not provide the gold star of verified provable correctness, they do provide a silver star of "seems pretty legit."

Monitoring is often cited as a means for verifying and testing distributed systems. Monitoring includes metrics, logs, distributed tracing systems such as Dapper[12] and Zipkin,[2] and alerts. While monitoring the system and detecting errors is an important part of running any successful service, and necessary for debugging failures, it is a wholly reactive approach for validating distributed systems; bugs can be found only once the code has made it into production and is affecting customers. All of these tools provide visibility into what your system is currently doing versus what it has done in the past. Monitoring allows you only to observe and should not be the sole means of verifying a distributed system.

Canarying new code is an increasingly popular way of "verifying" the code works. It uses a deployment pattern in which new code is gradually introduced into production clusters. Instead of replacing all the nodes in the service with the new code, a few nodes are upgraded to the new version. The metrics and/or output from the canary nodes are compared with the nodes running the old version. If they are deemed equivalent or better, more nodes can be upgraded to the canary version. If the canary nodes behave differently or are faulty, they are rolled back to the old version.

Canarying is very powerful and greatly limits the risk of deploying new code to live clusters. It is limited in the guarantees it can provide, however. If a canary test passes, the only guarantee you have is the canary version performs at least as well as the old version at this moment in time. If the service is not under peak load or a network partition does not occur during the canary test, then no information about how the canary performs compared with the old version is obtained for these scenarios.

Canary tests are most valuable for validating the new version works as expected in the common case and no regressions in this path have occurred in the service, but it is not sufficient for validating the system's correctness, fault tolerance, and redundancy.

All is not lost! You *can* employ testing methods that greatly increase your confidence the systems you build are correct.

Unit and integration tests. Engineers have long included unit and integration tests in their testing repertoires. Often, however, these tests are skipped or not focused on in distributed systems because of the commonly held beliefs that failures are difficult to produce offline and that creating a production-like environment for testing is complicated and expensive.

In a 2014 study, Yuan et al.[15] argue this conventional wisdom is untrue. Notably, the study shows that:

- Three or fewer nodes are sufficient to reproduce most failures;
- Testing error-handling code could have prevented the majority of catastrophic failures; and,
- Incorrect error handling of nonfatal errors is the cause of most catastrophic failures.

Unit tests can use mock-ups to prove intrasystem dependencies and verify the interactions of various components. In addition, integration tests can reuse these same tests without the mock-ups to verify they run correctly in an actual environment.

The bare minimum should be employing unit and integration tests that focus on error handling, unreachable nodes, configuration changes, and cluster membership changes. Yuan et al. argue this testing can be done at low cost and it greatly improves the reliability of a distributed system.

Random model checkers. Libraries such as QuickCheck[9] aim to provide property-based testing. QuickCheck allows users to specify properties about a program or system. It then generates a configurable amount of random input and tests the system against that input. If the properties hold for all inputs, the system passes the test; otherwise, a counterexample is returned. While QuickCheck cannot declare a system provably correct, it helps increase confidence that a system is correct by exploring a large portion of its state space. QuickCheck is not designed explicitly for testing distributed systems, but it can be used to generate input into distributed systems, as shown by Basho, which used it to discover and fix bugs in its distributed database, Riak.[13]

Fault-injection testing causes or introduces a fault in the system. In a distributed system a fault could be a dropped message, a network partition, or even the loss of an entire data center. Fault-injection testing forces these failures to occur and allows engineers to observe and measure how the system under test behaves. If failures do not occur, this does not guarantee a system is correct since the entire state space of failures has not been exercised.

Game days. In October 2014, Stripe uncovered a bug in Redis by running a fault-injection test. The simple test of running kill -9 on the primary node in the Redis cluster resulted in all data in that cluster being lost. Stripe referred to its process of running controlled fault-injection tests in production as *game day exercises*.[4]

Jepsen. Kyle Kingsbury has written a fault-injection tool called Jepsen[3] that simulates network partitions in the system under test. After the simulation, the operations and results are analyzed to see whether data loss occurred and whether claimed consistency guarantees were upheld. Jepsen has proved to be a valuable tool, uncovering bugs in many popular systems such as MongoDB, Kafka, ElasticSearch, etcd, and Cassandra.

Netflix Simian Army. The Netflix Simian Army[5] is a suite of fault-injection tools. The original tool, called Chaos Monkey, randomly terminates instances running in production, thereby injecting single-node failures into the system. Latency Monkey injects network lag, which can look like delayed messages or an entire service being unavailable. Finally, Chaos Gorilla simulates an entire Amazon availability zone going down.

As noted in Yuan et al.,[15] most of the catastrophic errors in distributed systems were reproducible with three or fewer nodes. This finding demonstrates that fault-injection tests do not even need to be executed in production and affect customers in order to be valuable and discover bugs.

Once again, passing fault-injection tests does not guarantee a system is correct, but these tests do greatly increase confidence the systems will behave correctly under failure scenarios. As Netflix puts it, "With the ever-growing Netflix Simian Army by our side, constantly testing our resilience to all sorts of failures, we feel much more confident about our ability to deal with the inevitable failures that we'll encounter in production and to minimize or eliminate their impact to our subscribers."[5]

Lineage-driven fault injection. Like building them, testing distributed systems is an incredibly challenging problem and an area of ongoing research. One example of current research is lineage-driven fault injection, described by Peter Alvaro et al.[1] Instead of exhaustively exploring the failure space as a model checker would, a lineage-driven fault injector reasons about successful outcomes and what failures could occur that would change this. This greatly reduces the state space of failures that must be tested to prove a system is correct.

Conclusion

Formal methods can be used to verify a single component is provably correct, but composition of correct components does not necessarily yield a correct system; additional verification is needed to prove the composition is correct. Formal methods are still valuable and worth the time investment for foundational pieces of infrastructure and fault-tolerant protocols. Formal methods should continue to find greater use outside of academic settings.

Verification in industry generally consists of unit tests, monitoring, and canaries. While this provides some confidence in the system's correctness, it is not sufficient. More exhaustive unit and integration tests should be written. Tools such as random model checkers should be used to test a large subset of the state space. In addition, forcing a system to fail via fault injection should be more widely used. Even simple tests such as running kill -9 on a primary node have found catastrophic bugs.

Efficiently testing distributed systems is not a solved problem, but by combining formal verification, model checking, fault injection, unit tests, canaries, and more, you can obtain higher confidence in system correctness.

Acknowledgments. Thank you to those who provided feedback, including Peter Alvaro, Kyle Kingsbury, Chris Meiklejohn, Flavio Percoco, Alex Rasmussen, Ines Sombra, Nathan Taylor, and Alvaro Videla.

Related articles on queue.acm.org

Monitoring and Control of Large Systems with MonALISA
Iosif Legrand et al.
http://queue.acm.org/detail.cfm?id=1577839

There's Just No Getting around It: You're Building a Distributed System
Mark Cavage
http://queue.acm.org/detail.cfm?id=2482856

Testing a Distributed System
Philip Maddox
http://queue.acm.org/detail.cfm?id=2800697

References

1. Alvaro, P., Rosen, J. and Hellerstein, J.M. Lineage-driven fault injection, 2015; http://www.cs.berkeley.edu/~palvaro/molly.pdf.
2. Aniszczyk, C. Distributed systems tracing with Zipkin; https://blog.twitter.com/2012/distributed-systems-tracing-with-zipkin.
3. Aphyr. Jepsen; https://aphyr.com/tags/Jepsen.
4. Hedlund, M. Game day exercises at Stripe: learning from "kill -9", 2014; https://stripe.com/blog/game-day-exercises-at-stripe.
5. Izrailevsky, Y. and Tseitlin, A. The Netflix Simian Army; http://techblog.netflix.com/2011/07/netflix-simian-army.html.
6. Killian, C., Anderson, J.W., Jhala, R., Vahdat, A. Life, death, and the critical transition: finding liveness bugs in system code, 2006; http://www.macesystems.org/papers/MaceMC_TR.pdf.
7. Lamport, L. and Yu, Y. TLC—the TLA+ model checker, 2011; http://research.microsoft.com/en-us/um/people/lamport/tla/tlc.html.
8. Newcomb, C., Rath, T., Zhang, F., Munteanu, B., Brooker, M. and Deardeuff, M. How Amazon Web Services uses formal methods. 2015. *Commun. ACM 58*, 4 (Apr. 2015), 68–73; http://cacm.acm.org/magazines/2015/4/184701-how-amazon-web-services-uses-formal-methods/fulltext.
9. QuickCheck; https://hackage.haskell.org/package/QuickCheck.
10. Sheehy, J. There is no now. *ACM Queue 13*, 33 (2015); https://queue.acm.org/detail.cfm?id=2745385.
11. Spin; http://spinroot.com/spin/whatispin.html.
12. Sigelman, B.H., Barroso, L.S., Burrows, M., Stephenson, P., Plakal, M., Beaver, D., Jaspan, S. and Shanbhag, C. Dapper, a large-scale distributed systems tracing infrastructure, 2010; http://research.google.com/pubs/pub36356.html.
13. Thompson, A. QuickChecking poolboy for fun and profit, 2012; http://basho.com/posts/technical/quickchecking-poolboy-for-fun-and-profit/.
14. Yang, J. et al. MODIST: transparent model checking of unmodified distributed systems, 2009; https://www.usenix.org/legacy/events/nsdi09/tech/full_papers/yang/yang_html/index.html.
15. Yuan, D. et al. Simple testing can prevent most critical failures: an analysis of production failures in distributed data-intensive systems, 2014; https://www.usenix.org/conference/osdi14/technical-sessions/presentation/yuan.
16. Wilcox, J.R. et al. Verdi: A framework for implementing and formally verifying distributed systems. In *Proceedings of the ACM SIGPLAN 2015 Conference on Programming Language Design and Implementation*: 357–368; https://homes.cs.washington.edu/~mernst/pubs/verify-distsystem-pldi2015-abstract.html.

Caitie McCaffrey (CaitieM.com; @Caitie) is the tech lead for observability at Twitter. Prior to that she spent the majority of her career building services and systems that power the entertainment industry at 343 Industries, Microsoft Game Studios, and HBO. She has worked on several video games including Gears of War 2 and 3 and Halo 4 and 5.

DOI:10.1145/2844110

Article development led by acmqueue
queue.acm.org

A view from computational journalism.

BY NICHOLAS DIAKOPOULOS

Accountability in Algorithmic Decision Making

EVERY FISCAL QUARTER, automated writing algorithms churn out thousands of corporate earnings articles for the Associated Press based on little more than structured data. Companies such as Automated Insights, which produces the articles for the AP, and Narrative Science can now write straight news articles in almost any domain that has clean and well-structured data: finance, sure, but also sports, weather, and education, among others. The articles are not cardboard either; they have variability, tone, and style, and in some cases readers even have difficulty distinguishing the machine-produced articles from human-written ones.[4]

It is difficult to argue with the scale, speed, and labor-saving cost advantage that such systems afford. But the trade-off for media organizations appears to be nuance and accuracy. A quick search on Google for "'generated by Automated Insights' correction'" yields results for thousands of articles that were automatically written, published, and then had to have corrections issued. The errors range from relatively innocuous ones about where a company is based, to more substantial wrong word choices—*missing* instead of *beating* earnings expectations, for example. Were any of these market-moving errors? Was the root cause bad data, a faulty inference, or sloppy engineering? What is the right way to post corrections?

Algorithmic curation of content is also behind some of the most important and influential news-dissemination platforms that virtually all of us use. A recent Pew study found Facebook is a source of news about government and politics for 61% of millennials,[18] yet a majority of the public is not aware the Facebook newsfeed is algorithmically curated.[11] This becomes a lot more problematic when you consider Facebook can affect voter turnout in elections based merely on the amount of hard news promoted in an individual's news feed.[24] This bit of information, together with recent research showing biased search results can shift the voting preferences of undecided voters,[10] points to the need to start asking questions about the degree to which such curation and ranking systems can affect democratic processes.

These are just a few examples of algorithms influencing our media and information exposure. But the impact of automated decision making is being felt throughout virtually all strands of industry and government, whether it be fraud-detection systems for municipalities managing limited resources, a formula that grades and ranks teacher performance, or the many ways in which dynamic product pricing is done by Amazon, Airbnb, or Uber.[8] It is time to think seriously about how the algorithmically informed decisions now driving large swaths of society should be accountable to the public. In the face of important or expensive errors, discrimination, unfair denials of public services, or censorship, when and how should algorithms be reined in?

Computer science and engineering

IMAGE BY BAHRI ALTAY

professionals have a role to play here. While autonomous decision making is the essence of algorithmic power, the human influences in algorithms are many: criteria choices, optimization functions, training data, and the semantics of categories, to name just a few. Often a human operator is involved in a final decision only to have been influenced by the algorithm's nudging and suggestions along the way.

Algorithmic Decision Making

It is helpful first to get the lay of the land in terms of the different types of atomic decisions that algorithms make. These include processes that prioritize, classify, associate, and filter.

Prioritizing is something we do on a daily basis to cope with the information onslaught. As beings with limited time and attention, we cannot ignore the need to economize. Algorithms prioritize information in a way that emphasizes or brings attention to certain things at the expense of others; by definition prioritization is about discrimination. As a result, there may be ramifications to individuals or other entities that should be considered during design. Search engines are canonical examples, but there are many other consequential rankings—for everything from the quality of schools and hospitals, to the riskiness of illegal immigrants on watch lists.[14] The criteria used in a ranking, how they are defined and datafied, and their weighting are essential design decisions that deserve careful consideration and scrutiny.

Classification decisions mark a particular entity as belonging to a given class by considering key characteristics of that entity. Class membership can then drive all kinds of downstream decisions. The opportunities for bias, uncertainty, or outright mistakes are plentiful in automated classification. The training data that is the basis for supervised machine-learning algorithms is an important consideration, given the human biases that may be lurking there. Recently published research by Shilad Sen and collaborators underscores the need to consider the cultural community from which training data is collected.[23] Data crowdsourced from Mechanical Turk may be useful for widely shared and agreed-upon knowledge but introduces discrepancies in other cases. The bottom line, as they write, is this: "When collecting a gold standard researchers and practitioners must consider the audience of the gold standard, the system or algorithm that uses it, and the type of knowledge."

In developing classification algorithms, designers must also consider the accuracy of the classifications: the false positives and false negatives, and the consequences to stakeholders of either of those types of errors. For example, in Boston a man classified as having a fraudulent driver's license (a false positive in this case) was unable to work until the matter was resolved. Classification algorithms can be tuned to make fewer of either type of mistake, but as one goes down the other goes up. Tuning can grant privilege to different stakeholders and outcomes in a decision, implying that designers make an essential value judgment when balancing error rates.[16]

Association decisions revolve around creating relationships between entities. The semantics of those relationships can vary from the generic "related to" or "similar to" to distinct domain-specific meanings. These associations lead to connotations in their human interpretation. For example, when a man in Germany searched for his name on Google and the autocomplete suggestions connected him to "scientology" and "fraud," this was both meaningful and unsettling to the man, leading to a defamation lawsuit that he ultimately won.[6] Collaborative filtering is a popular class of algorithm that defines an association neighborhood (a cluster, really) around an entity and uses those close ties to suggest or recommend other items.[13] The quantification bugbear torments associations just as it does rankings and classifications. The criteria that are defined and measured, and the similarity metrics that dictate how closely two entities match, are engineering choices that can have implications for the accuracy of an association, both objectively and in terms of how that association is interpreted by other people.

One issue with the church of big data is its overriding faith in correlation as king. Correlations certainly do create statistical associations between data dimensions. But despite the popular adage, "Correlation does not equal causation," people often misinterpret correlational associations as causal. The man whose name was associated with fraud on Google may or may not be the cause of that association, but we certainly read it that way. This all indicates a challenge in communicating associations and the need to distinguish correlative vs. causal associations.

Filtering decisions involve including or excluding information according to various rules or criteria. Often this emerges at the user-interface level in, for example, news-reading applications such as Facebook or Flipboard. Since there is practically always some troll or miscreant willing to soil the sandbox, moderation and filtering are crucial elements when publishing social media. Online comments are sometimes filtered algorithmically to determine whether or not they are antisocial and therefore unworthy of public consumption. Of course, the danger here is in going too far—into censorship. Censorship decisions that may be false positives should be carefully considered, especially in cultures where freedom of speech is deeply ingrained.

Ultimately, when considering the various decisions and predictions that an algorithm may make, particularly ones that can affect people, but also those that affect property, you must consider the errors and potential for discrimination and censorship that can arise. Just read ACM's ethics policies.[1,2]

The ACM Code of Ethics for software engineering lists eight principles that are aspirations for professional behavior. First and foremost is that software engineers should act in the *public interest*: to be accountable and responsible for their work, to moderate private interests with public good, to ensure safety and privacy, to avoid deception, and to consider the disadvantaged. The general moral imperatives of ACM include "avoid harm to others," "be fair and take action not to discriminate," and "respect the privacy of others."

Let that sink in.

Have you ever programmed an algorithm that could violate any of these mandates? It may not have been intentional, but there are side effects you might have noticed if you had done more thorough benchmarking or considered the human contexts in which the output of your algorithmic creations would be used. Is it possible that you used a protected trait such as race, ethnicity, religion, nationality, gender, sexuality, disability, marital status, or age in an inappropriate way? The point is these ethical ideals need to be incorporated throughout the engineering process, for people to be constantly reconsidering: What are the consequences of the unlikely false positive, or the impacts of how criteria are measured and defined in training datasets? Helen Nissenbaum was far ahead of the curve when, almost two decades ago, she recommended the development of explicit standards of care including rigorous engineering guidelines that would consider these issues.[21]

Government vs. Private Sector Accountability

The mandate for accountable algorithms (and for the accountability of the people behind them) for government usage is a bit different from that for the private sector. In the case of the modern democratic state, citizens elect a government that provides social goods and exercises its power and control in a way that is moderated through norms and regulation. The government is legitimate only to the extent it is accountable to the citizenry. But algorithms are largely unregulated now, and they are indeed exercising power over individuals or policies in a way that in some cases (for example, hidden government watch lists) lacks any accountability whatsoever. A recent academic review of Social Security Administration models used to predict life expectancy and solvency found systematic underestimation, implying that funds were on firmer ground than warranted.[15] We, the governed, should find it unacceptable there is no transparency or even systematic benchmarking and evaluation of these forecasts, given the important policy decisions they feed.

Corporations, on the other hand, do not have the same mandate for public accountability, though they may sometimes be impelled to act through social pressure (for example, boycotts). Perhaps more compelling is the capitalist argument that higher data quality and thus better inference will lead to more satisfied customers. The most clear-cut way to do this is to design processes that adjudicate and facilitate the correction of false positives by end users. Allowing users to inspect, dispute, and correct inaccurate labels in the data would improve overall data quality for machine-learning applications.

Transparency can be a mechanism that facilitates accountability, one that we should demand from government and exhort from industry. Corporations often limit their transparency out of fear of losing a competitive advantage from a trade secret or of exposing their systems to gaming and manipulation. Complete source-code transparency of algorithms, however, is overkill in many if not most cases. Instead, the disclosure of certain key pieces of information, including aggregate results and benchmarks, would be far more effective in communicating algorithmic performance to the public.

When automobile manufacturers disclose crash-test results, they do not tell you the details of how they engineered the vehicle. When local municipalities publish restaurant inspection scores, they do not disclose a restaurant's unique recipes. The point is there are models for transparency that can effectively audit and disclose information of interest to the public without conflicting with intellectual property and trade secrets. In some cases fear of manipulation and gaming of disclosed criteria are unfounded. For example, criteria not based on user behaviors offers no mechanism for gaming from individuals who have no direct control over those attributes. In some cases gaming or manipulation of an algorithm might even be a good thing. For example, if credit-rating agencies disclosed the criteria they used to score individuals, wouldn't it be a great thing if everyone gamed their score? They would have to act financially responsibly in that game.

For government, the Freedom of Information Act (FOIA) and similar laws in the U.S. and many other jurisdictions compel disclosure of government records and data when requested, though there are, of course, exceptions, such as when a government integrates a third-party system protected by trade secrets. There has been at least one successful use of an FOIA request to compel the disclosure of government source code.[19] In another FOIA case decided in 1992, the Federal Highway Administration resisted disclosing the algorithm it used to compute safety ratings for carriers, but ultimately lost in court to a plaintiff that successfully argued the government must disclose the weighting of factors used in that calculation.[9]

Thus, FOIA is of some use in dealing with the government use of algorithms, but one of the issues with current FOIA law in the U.S. is it does not require agencies to create documents that do not already exist. Hypothetically, a government algorithm could compute a variable in memory that corresponds to some protected class such as race and use that variable in some other downstream decision. As long as that variable in memory was never directly stored in a document, FOIA would be unable to compel its disclosure. Audit trails could help mitigate this issue by recording stepwise correlations and inferences made during the prediction process. Guidelines should be developed for when government use of an algorithm should trigger an audit trail.[3]

Allowing users to inspect, dispute, and correct inaccurate labels in the data would improve overall data quality for machine-learning applications.

It may be time to reconsider FOIA regulation along the lines of what I propose be called the Freedom of Information Processing Act (FOIPA). FOIPA would sidestep the issues associated with disclosing formulas or source code and instead allow the public to submit benchmark datasets the government would be required to process through its algorithm and then provide the results. This would allow interested parties, including journalists or policy experts, to run assessments that prod the government algorithm, benchmark errors, and look for cases of discrimination or censorship. For example, you could take two rows of data that varied on just one piece of sensitive information like race and examine the outcome to determine if unjustified discrimination occurred.

An Algorithmic Transparency Standard

So far we have covered a lot of ground: from the decisions that algorithms make, to the stakes of errors, and the ethics of responsibly engineering these systems. But you may still be asking yourself this overriding question: What can we and should we be disclosing about our algorithms?

To help answer that question I led the organization of a workshop on Algorithmic Transparency in the Media at Columbia University's Tow Center for Digital Journalism in spring 2015. About 50 people from the news media and academia convened to discuss how to work toward ideas that support a robust policy of news and information stewardship via algorithms. We discussed case studies on "Automatically Generated News Content," "Simulation, Prediction, and Modeling in Storytelling," and "Algorithmically Enhanced Curation," and brainstormed dimensions of the various algorithms in play that might be disclosed publicly.

Based on the wide array of ideas generated at the workshop, we came up with five broad categories of information that we might consider disclosing: human involvement, data, the model, inferencing, and algorithmic presence.

Human involvement. At a high level, transparency around human involvement might involve explaining the goal, purpose, and intent of the algorithm, including editorial goals and the human editorial process or social-context crucible from which the algorithm was cast. Who at your company has direct control over the algorithm? Who has oversight and is accountable? Ultimately we want to identify the authors, or the designers, or the team that created and are behind this thing. In any collective action it will be difficult to disaggregate and assign credit to exactly who did what (or might be responsible for a particular error),[21] yet disclosure of specific human involvement would bring about social influences that both reward individuals' reputations and reduce the risk of free riding. Involved individuals might feel a greater sense of public responsibility and pressure if their names are on the line.

Data. There are many opportunities to be transparent about the *data* that drives algorithms in various ways. One avenue for transparency here is to communicate the quality of the data, including its accuracy, completeness, and uncertainty, as well as its timeliness (since validity may change over time), representativeness of a sample for a specific population, and assumptions or other limitations. Other dimensions of data processing can also be made transparent: how was it defined, collected, transformed, vetted, and edited (either automatically or by human hands)? How are various data labels gathered, and do they reflect a more objective or subjective process? Some disclosure could be made about whether the data was private or public, and if it incorporated dimensions that if disclosed would have personal privacy implications. If personalization is in play, then what types of personal information are being used and what is the collected or inferred profile of the individual driving the personalization?

The model itself, as well as the modeling process, could also be made transparent to some extent. Of high importance is knowing what the model actually uses as input: What features or variables are used in the algorithm? Often those features are weighted: What are those weights? If training data was used in some machine-learning process, then you would characterize the data used for that along all of the potential dimensions described here. Some software-modeling tools have different assumptions or limitations: What were the tools used to do the modeling?

The main challenge moving forward is to determine appropriate mechanisms for disclosure that are beneficial but do not kill usability.

Of course, this all ties back into human involvement, so we want to know the rationale for weightings and the design process for considering alternative models or model comparisons. What are the assumptions (statistical or otherwise) behind the model, and where did those assumptions arise? And if some aspect of the model was not exposed in the front end, why was that?

Inferencing. The inferences made by an algorithm, such as classifications or predictions, often leave questions about the accuracy or potential for error. Algorithm creators might consider benchmarking against standard datasets and with standard measures of accuracy to disclose some key statistics. What is the margin of error? What is the accuracy rate, and how many false positives versus false negatives are there? What kinds of steps are taken to remediate known errors? Are errors a result of human involvement, data inputs, or the algorithm itself? Classifiers often produce a confidence value, and this, too, could be disclosed in aggregate to show the average range of those confidence values as a measure of uncertainty in the outcomes.

Algorithmic presence. Finally, we might disclose if and when an algorithm is being employed at all, particularly if personalization is in use, but also just to be made aware of, for example, whether A/B testing is being used. Other questions of visibility relate to surfacing information about which elements of a curated experience have been filtered away. In the case of Facebook, for example, what are you not seeing, and, conversely, what are you posting (for example, in a news feed) that other people are not seeing.

Technical systems are fluid, so any attempt at disclosure has to consider the dynamism of algorithms that may be continually learning from new data. The engineering culture must become ingrained with the idea of continual assessment. Perhaps new multidisciplinary roles relating to algorithmic risk modeling or transparency modeling need to be created so these questions receive dedicated and sustained attention.

In the case of information disclosure that would make an entity look bad or be substantially damaging to its public image, I am enough of a pragmatist not to expect voluntary compliance with any ethical mandate. Entities use information disclosure to engage in strategic impression management. Instead, we might look toward regulations that compel information disclosure or at least routine audits around key algorithmically influenced decisions, such as credit scoring.[3] The dimensions of information disclosure articulated here could also feed such regulatory designs.

In other cases, a more adversarial approach may be necessary for investigating black-box algorithms. In the journalism domain, I refer to this as algorithmic accountability reporting,[5] and it involves sampling algorithms along key dimensions to examine the input-output relationship and investigate and characterize an algorithm's influence, mistakes, or biases. This is an extension of traditional investigative accountability journalism, which for many years has had the goal of exposing malfeasance and misuse of power in government and other institutions.

To provide a flavor of this type of reporting, I started investigating in early 2015 the much-publicized Uber surge-pricing algorithm.[8] The ride-sharing app uses dynamic pricing to "encourage more drivers to go online" and try to match supply and demand. It is an easy line to buy and appeals to basic economic theories. My investigation, based on an analysis of a month's worth of pricing data in Washington, D.C., indicated that instead of motivating a fresh supply of drivers to get on the road, surge pricing instead redistributes drivers already on the road. This is important because it means the supply of drivers will shift toward neighborhoods offering higher surge prices, leaving other neighborhoods undersupplied and with longer waiting times for a car. Uber cars are rival goods, and the analysis raises questions about which neighborhoods end up with better or worse service quality. Higher prices and better service for some means worse service for others.

Challenges Ahead

There is still much research to be done to understand when and how best to act responsibly and be transparent about the algorithms we build. Deciding what to disclose is just a start; the communication vehicle also needs to be explored. Human-computer interaction as well as machine learning and software engineering have roles to play here.

This article has broadly articulated classes of information that might be disclosed about algorithms: the human element, data, the model, inferences, and the algorithmic presence. Practically speaking, however, each algorithm is a bit different and must be understood in context to determine what can be disclosed. This is both a technical process as well as a human-centered one. We must develop a process of information-disclosure modeling that includes thinking about how the public would use any particular bit of information disclosed.

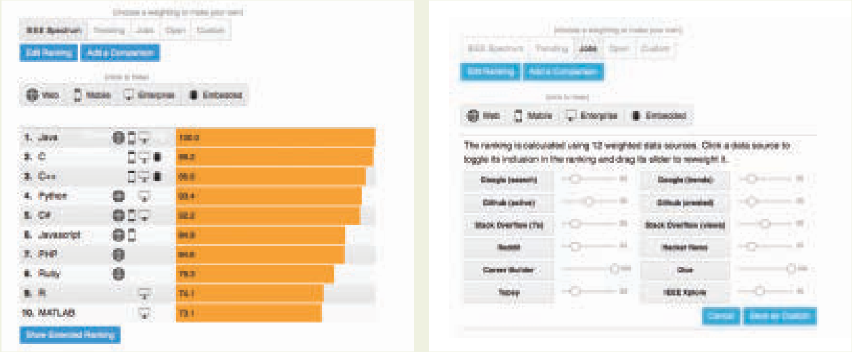
Providing transparency and explanations of algorithmic outputs might serve a number of goals, including scrutability, trust, effectiveness, persuasiveness, efficiency, and satisfaction. Ultimately, we need to do user modeling and think through a series of questions such as what we want to accomplish with each bit of disclosed information, and what behavior we are trying to affect. What are the decisions the public would make based on that information? What information could be disclosed that would make those decisions more effective, or mitigate risks? How might a user respond to this information?[25]

Another dimension to the human-interface challenge is to design effective user experiences for transparency information. Recent research has shown algorithmic transparency information can lead to a better outcome but comes at the expense of enjoyable and reassuring usage.[22] The main challenge moving forward is to determine appropriate mechanisms for disclosure that are beneficial but do not kill usability. Additionally, some users may simply not care, while others are deeply interested, raising the design challenge of accommodating the needs of many publics, while not polluting the user experience with a surfeit of information for the uninterested. Of course, algorithmic transparency need not be directly integrated into the user experience. For example, corporations or governments might issue algorithmic transparency reports on a quarterly or yearly basis that would disclose aspects of the five dimensions discussed previously.

One approach gaining some attention in the research community is to develop machine-learning methods that can be explained in ways that humans can readily understand. For example, the Bayesian Rule List (BRL) technique learns a series of human-readable rules that when chained together offer a human-readable explanation of the classifier.[17] Other methods are being developed in natural language generation (NLG) to output text that explains why or how a decision was reached. Imagine if your favorite machine-learning library, say scikit-learn, could explain in a sentence why a particular input case was classified the way it was. That would be useful for debugging, if nothing else.

On the other hand, we might consider integrated presentation strategies that leverage data visualization to succinctly communicate the workings of an algorithm. For example, early research has shown that salient visual explanations such as histograms can be effective for communicating recommendation explanations.[13] In a collaboration with *IEEE Spectrum*, I built a data-driven app ranking top programming languages that feeds off of 12 different weighted data inputs to arrive at a ranking.[7] Instead of making it a static, fixed ranking, however, like, say, the annual *U.S. News & World Report* College Rankings, we defined several different weightings. So, for example, you could quickly rank languages weighted toward job listings or open source projects, and you could create your own custom ranking by deciding which data inputs were important to you and reweight them accordingly (see the accompanying figure). You could also visually compare your rankings to do a sensitivity analy-

IEEE top programming languages ranking and reweighting interfaces.



1. Java 100.0
2. C 99.2
3. C++ 95.5
4. Python 93.4
5. C# 92.2
6. Javascript 84.9
7. PHP 84.5
8. Ruby 78.3
9. R 74.1
10. MATLAB 73.1

The ranking is calculated using 12 weighted data sources. Click a data source to toggle its inclusion in the ranking and drag its slider to reweight it.

sis and see how a change in a factor would impact the resulting output ranking. Based on 1,285 tweets that people shared about the app, we found about one in six indicated people were reweighting the ranking in various ways. While it is too early to claim victory in designing dynamic and transparent ranking interfaces, this is at least a step in the direction I envision for interactive modeling.

There are technical challenges here, too. In particular, concerns often arise over manipulation and gaming that may be enabled by disclosing information about how systems work. A certain amount of threat modeling may be necessary if transparency is required. If a particular piece of information were made available about an algorithm, how might that be gamed, manipulated, or circumvented? Who would stand to gain or lose? Manipulation-resistant algorithms also need to be designed and implemented. Feature sets that are robust and difficult to game need to be developed.

The software engineering of algorithms also needs to consider architectures that support transparency and feedback about algorithmic state so they can be effectively steered by people.[20] Algorithm implementations should support callbacks or other logging mechanisms that can be used to report information to a client module. This is essential systems work that would form the basis for outputting audit trails.

Finally, we must work on machine-learning and data-mining solutions that directly take into account provisions for fairness and anti-discrimination. For example, recent research has explored algorithmic approaches that can identify and correct for disparate impact in classifiers by statistically transforming the input data set so that prediction of protected attributes is not possible.[12] Additional research is needed in this space as different types of models and data types may demand different technical approaches and adaptations.

Conclusion

Society must grapple with the ways in which algorithms are being used in government and industry so that adequate mechanisms for accountability are built into these systems. The ideas presented here about acting ethically and responsibly when empowering algorithms to make decisions are important to absorb into your practice. There is much research still to be done to understand the appropriate dimensions and modalities for algorithmic transparency, how to enable interactive modeling, how journalism should evolve, and how to make machine learning and software engineering sensitive to, and effective in, addressing these issues. C

Related articles on queue.acm.org

Online Algorithms in High-frequency Trading
Jacob Loveless, Sasha Stoikov, and Rolf Waeber
http://queue.acm.org/detail.cfm?id=2534976

AI Gets a Brain
Jeff Barr and Luis Felipe Cabrera
http://queue.acm.org/detail.cfm?id=1142067

Other People's Data
Stephen Petschulat
http://queue.acm.org/detail.cfm?id=1655240

References

1. ACM. Software Engineering Code of Ethics and Professional Practice, 2015; https://www.acm.org/about/se-code#full.
2. ACM Code of Ethics and Professional Conduct. 1992; https://www.acm.org/about/code-of-ethics.
3. Citron, D. and Pasquale, F. The scored society: due process for automated predictions. *Washington Law Review 89* (2014).
4. Clerwall, C. Enter the robot journalist. *Journalism Practice 8*, 5 (2014): 519–531.
5. Diakopoulos, N. Algorithmic accountability: Journalistic investigation of computational power structures. *Digital Journalism 3*, 3 (2015): 398–415.
6. Diakopoulos, N. Algorithmic defamation: the case of the shameless autocomplete. Tow Center for Digital Journalism, 2014.
7. Diakopoulos, N., et al. Data-driven rankings: the design and development of the IEEE Top Programming Languages news app. *Proceedings of the Symposium on Computation + Journalism*, 2014.
8. Diakopoulos, N. How Uber surge pricing really works. *Washington Post Wonkblog* (Apr. 17, 2015).
9. *Don Ray Drive-A-Way Co. v. Skinner*, 785 F. Supp. 198 (D.D.C. 1992); http://law.justia.com/cases/federal/district-courts/FSupp/785/198/2144490/.
10. Epstein, R. and Robertson, R.E. The search engine manipulation effect (SEME) and its possible impact on the outcomes of elections. In *Proceedings of the National Academy of Sciences 112*, 33 (2015).
11. Eslami, M. et al. 'I always assumed that I wasn't really that close to [her]:' Reasoning about invisible algorithms in the news feed. In *Proceedings of the 33rd Annual ACM SIGCHI Conference on Human Factors in Computing Systems*, 2015.
12. Feldman, M., et al. Certifying and removing disparate impact. In *Proceedings of the 21st ACM International Conference on Knowledge Discovery and Data Mining*, 2015, 259–268.
13. Herlocker, J.L. et al. Explaining collaborative filtering recommendations. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, 2000, 241–250.
14. Kalhan, A. Immigration policing and federalism through the lens of technology, surveillance, and privacy. *Ohio State Law Journal 74* (2013).
15. Kashin, K. et al. Systematic bias and nontransparency in US Social Security Administration forecasts. *Journal of Economic Perspectives 29*, 2 (2015).
16. Kraemer, F. et al. Is there an ethics of algorithms? *Ethics and Information Technology 13*, 3 (2010), 251–260.
17. Letham, B. et al. Building interpretable classifiers with rules using Bayesian analysis. *Annals of Applied Statistics*, 2015.
18. Mitchell, A. et al. Millennials and Political News. Pew Research Center, Journalism and Media (June 1, 2015); http://www.journalism.org/2015/06/01/millennials-political-news/.
19. Muckrock. Source code of HEAT SAFETY TOOL, 2011; https://www.muckrock.com/foi/united-states-of-america-10/source-code-of-heat-safety-tool-766/.
20. Mühlbacher, T. et al. Opening the black box: strategies for increased user involvement in existing algorithm implementations. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014) 1643–1652.
21. Nissenbaum, H. Accountability in a computerized society. *Science and Engineering Ethics 2*, 1 (1996): 25–42.
22. Schaffer, J. et al. Getting the message?: a study of explanation interfaces for microblog data analysis. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, 2015, 345–356.
23. Sen, S. et al. Turkers, Scholars, 'Arafat' and 'Peace:' Cultural communities and algorithmic gold standards. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work and Social Computing*, 2015. 826–838.
24. Sifry, M. Facebook wants you to vote on Tuesday. Here's how it messed with your feed in 2012. *Mother Jones* (Oct. 31, 2014); http://www.motherjones.com/politics/2014/10/can-voting-facebook-button-improve-voter-turnout.
25. Tintarev, N. and Masthoff, J. A survey of explanations in recommender systems. *Proceedings of the International Conference on Data Engineering*, 2007, 801–810.

Nicholas Diakopoulos is an assistant professor at the University of Maryland, College Park, Philip Merrill College of Journalism, with courtesy appointments in the College of Information Studies and Department of Computer Science. He is also a fellow at the Tow Center for Digital Journalism at Columbia University

DOI:10.1145/2812802

This publicly available curated dataset of almost 100 million photos and videos is free and legal for all.

BY BART THOMEE, BENJAMIN ELIZALDE, DAVID A. SHAMMA, KARL NI, GERALD FRIEDLAND, DOUGLAS POLAND, DAMIAN BORTH, AND LI-JIA LI

YFCC100M: The New Data in Multimedia Research

THE PHOTOGRAPH AND our understanding of photography transitioned from a world of unprocessed rolls of C-41 sitting in a refrigerator 50 years ago to sharing photos on the 1.5-inch screen of a point-and-shoot camera 10 years ago. Today, the photograph is again something different. The way we take photos has fundamentally changed from what it was. We can view, share, and interact with photos on the device that took them. We can edit, tag, or "filter" photos directly on the camera at the same time we take the photo. Photos can be automatically pushed to various online sharing services, and the distinction between photos and videos has lessened. Beyond this, and more important there are now lots of them. As of 2013, to Facebook alone more than 250 billion photos had been uploaded and on average received more than 350 million new photos each day,[6] while YouTube reported in July 2015 that 300 hours of video were uploaded every minute.[22] A back-of-the-envelope calculation estimated 10% of all photos in the world were taken in the last 12 months, as of more than four years ago.[8]

Today, a large number of shared digital media objects have been uploaded to services like Flickr and Instagram, which, along with their metadata and social ecosystem, form a vibrant environment for finding solutions to many research questions at scale. Photos and videos provide a wealth of information covering entertainment, travel, personal records, and various other aspects of life as they were when taken. Viewed collectively, they represent knowledge beyond what is captured in any individual snapshot and provide information on trends, evidence of phenomena or events, social context, and societal dynamics. Consequently, media collections are useful for qualitative and quantitative empirical research in many domains. However, scientific endeavors in fields like social computing and computer vision have generally relied on independently collected multimedia datasets, complicating research and synergy. Needed is a more substantial dataset for researchers, engineers, and scientists around the globe.

» key insights

- In the same way freely licensed works have advanced user-generated content rights, we propose a common dataset to advance scientific discovery and enhance entrepreneurship across academia and industry.
- We introduce a dataset with the volume and complexity to address questions across many fields—from computer vision to social computing to artificial intelligence and sensemaking.
- The core photos and videos in the dataset reveal a surprising amount of detail about how people experience and interact with the world and with each other; the multimedia commons extends the dataset annotations in a community-driven manner.

IMAGE BY ANDRIJ BORYS ASSOCIATES, USING PHOTOS FROM YFCC100M DATASET BY TREY RATCLIFF, SKOEBER, DAVID KRACHT, GEIR TØNNESSEN, (M), CAMIL TULCAN, PAUL BICA, LOTUS CARROLL, LYLE VINCENT, JOSÉ EUGENIO GÓMEZ RODRÍGUEZ, MORBCN, AND THOMAS LEUTHARD. ALL PHOTOS © FLICKR

To address the call for scale, openness, and diversity in academic datasets, we take the opportunity in this article to present a new multimedia dataset containing 100 million media objects we developed over the past two years and explain the rationale behind its creation. We discuss its implications for science, research, engineering, and development and demonstrate its usefulness toward tackling a range of problems in multiple domains. The release of the dataset represents an opportunity to advance research, giving rise to new challenges and addressing existing ones.

Sharing Datasets

Datasets are critical for research and exploration,[16] as data is required to perform experiments, validate hypotheses, analyze designs, and build applications. Over the years, multimedia datasets have been put together for research and development; Table 1 summarizes the most popular multimedia datasets over time. However, most of them cannot truly be called multimedia, as they contain only a single type of media, rather than a mixture of modalities (such as photos, videos, and audio). Datasets range from one-off instances created exclusively to support the work presented in a single paper or demo, or "short-term datasets," to those created with multiple related or separate endeavors in mind, or "long-term datasets." A notable problem is the collected data is often not made publicly available. While this restriction is sometimes out of necessity due to the proprietary or sensitive nature of the data, it is not always the case.

The topic of sharing data for replication and growth has arisen several times over the past 30 years alone[2,7,18] and has been brought into discussion through ACM's SIGCHI.[20] This "sharing discussion" reveals many of the underlying complexities of sharing, with regard to both the data (such as what exactly is considered data) and the sharing point of view (such as incentives and disincentives for doing so); for example, one might be reluctant to share data freely, as it has a value from the often substantial amount of time, effort, and money invested in collecting it. Another barrier arises when data is harvested for research under a general umbrella of "academic fair use" without regard to its licensing terms. Beyond the corporate legal issues, such academic fair use may violate the copyright of the owner of the data that in many user-generated content sites like Flickr stays with the creator. The Creative Commons (CC), a nonprofit organization founded in 2001, seeks to build a rich public domain of "some rights reserved" media, sometimes referred to as the "copyleft movement." The licenses allow media owners to communicate how they would like their media to be rights reserved; for example, an owner can indicate a photo may be used for only noncommercial purposes or someone is allowed to remix it or turn it into a collage. Depending how the licensing options are chosen, CC licenses can be applied that are more restrictive (such as CC Attribution-NonCommercial-NoDerivs/CC-BY-NC-ND license) or less restrictive (such as CC Attribution-ShareAlike/CC-BY-SA) in nature. A public dataset with clearly marked licenses that do not overly impose restrictions on how the data is used (such as those offered by CC) would therefore be suitable for use in both academia and industry.

We underscore the importance of sharing—perhaps even its principal argument—is it ensures data equality for research. While the availability of data alone may not necessarily be sufficient for the exact reproduction of scientific results (since the original experimental conditions would also have to be replicated as closely as possible, which may not always be possible), research should start with publicly sharable and legally usable data that is flexible and rich enough to promote advancement, rather than with data that serves only as a one-time collection for a specific task and that cannot be shared. Shared datasets can play a singular role in achieving research growth and facilitating synergy within the research community otherwise difficult to achieve.

YFCC100M Dataset

We created the Yahoo Flickr Creative Commons 100 Million Dataset[a] (YFCC100M) in 2014 as part of the Yahoo Webscope program, which is a reference library of interesting and scientifically useful datasets. The YFCC100M is the largest public multimedia collection ever released, with a total of 100 million media objects, of which approximately 99.2 million are photos and 0.8 million are videos, all uploaded to Flickr between 2004 and 2014 and published under a CC commercial or noncommercial license. The dataset is distributed through Amazon Web Services as a 12.5GB compressed archive containing only metadata. However, as with many datasets, the YFCC100M is constantly evolving; over time, we have released and will continue to release various expansion packs containing data not yet in the collection; for instance, the actual photos and videos, as well as several visual and aural features extracted from the data, have already been uploaded to the cloud,[b] ensuring the dataset remains accessible and intact for years to come. The YFCC100M dataset overcomes many of the issues affecting existing multimedia datasets in terms of modalities, metadata, licensing, and, principally, volume.

Metadata. Each media object included in the dataset is represented by its Flickr identifier, the user who created it, the camera that took it, the time it was taken and uploaded, the location where it was taken (if available), and the CC license under which it was published. The title, description, and tags are also available, as are direct links to its page and content on Flickr. Social features, comments, favorites, and followers/following data are not included, as such metadata changes from day to day. This information is, however, easily obtained by querying the Flickr API.[c] We are working toward the release of the Exif metadata of the photos and videos as an expansion pack.

Tags. There are 68,552,616 photos and 418,507 videos in the dataset users have annotated with tags, or keywords. The tags make for a rich, diverse set of entities related to people (`baby`, `family`), animals (`cat`, `dog`), locations (`park`, `beach`), travel (`nature`, `city`), and more. A total of 3,343,487 photos and 7,281 videos carry machine tags—labels automatically generated and added by camera, computer, application, or other automated system.

Timespan. Although the YFCC100M dataset contains media uploaded between the inception of Flickr in 2004 and creation of the dataset in 2014, the actual timespan during which they were captured is much longer. Some scans of books and newspapers have even been backdated to the early 19th century when originally published. However, note camera clocks are not always set to the correct date and time, and some photos and videos erroneously report they were captured in the distant past or future; Figure 1 plots the moments of capture and upload of photos and videos during the period 2000–2014, or 99.6% of the media objects in the dataset.

Locations. There are 48,366,323 photos and 103,506 videos in the dataset that have been annotated with a geographic coordinate, either manually by the user or automatically through GPS. The cities in which more than 10,000 unique users captured media are Hong Kong, London, New York, Paris, San Francisco, and Tokyo. Overall, the dataset spans 249 different territories (such as countries and islands) and includes photos and videos taken in international waters and international airspace (see Figure 2).

Cameras. Table 2 lists the top 25 cameras used to take the photos and videos in the dataset as overwhelmingly digital single lens reflex (DSLR) models, with the exception of the Apple iPhone. Considering the most popular cameras in the Flickr community are primarily various iPhone models[d] this bias in the data is likely due to CC licenses attracting a certain subcommu-

[a] Dataset available at https://bit.ly/yfcc100md

[b] Photos, videos, and features available at http://www.multimediacommons.org/

[c] https://www.flickr.com/services/api/

[d] https://www.flickr.com/cameras/

Table 1. Popular multimedia datasets used by the research community. When various versions of a particular collection are available, we generally include only the most recent one. PASCAL, TRECVID, MediaEval, and ImageCLEF are recurring annual benchmarks that consist of one or more challenges, each with its own dataset; here, we report the total number of media objects aggregated over all datasets that are part of the most recent edition of each benchmark.

Year	Dataset	Type	Image	Video	Audio	License	Accessibility	Content
1966	Brodatz	texture	<1K	-	-	©	⬇	
1996	COIL-100	object	7K	-	-	ⓘ	⬇	★
1996	Corel	stock	60K	-	-	©	🛒 🚚	★
2000	FERET	face	14K	-	-	©	⬇ ✎	★
2005	Yale Face B+	face	16K	-	-	©	⬇	★
2005	Ponce	texture	1K	-	-	©	⬇	★
2007	Caltech-256	object	30K	-	-	ⓘ	⬇	★
2007	Oxford	buildings	5K	-	-	©	⬇	★
2008	CMU Multi-PIE	face	750K	-	-	©	🚚 ✎	★
2008	Tiny Images	web	80M	-	-	©	⬇	⯪ A 📊
2008	MIRFLICKR-25K	Flickr	25K	-	-	🅭	⬇	★ 📷 🕘 📊 👤 🏷
2009	NUS-WIDE	Flickr	270K	-	-	©	⬇ 🔗	★ 📷 🕘 📊 🌍 🏷
2009	ImageNet	web	14M	-	-	©	⬇ ✎ ➜	★ A 📊 □
2010	SUN	web	131K	-	-	©	⬇	★ A 📊
2010	MIRFLICKR-1M	Flickr	1M	-	-	🅭	⬇	⯪ 📷 🕘 📊 👤 🏷
2012	PASCAL	Flickr	23K	-	-	©	⬇ ✎ ➜ 🏆 *	★ □ ▱
2013	MS Clickture	web	40M	-	-	©	🚚 ✎ ➜ 🔗 **	☰
2014	Sports-1M	sports	-	1M	-	🅭	⬇ 🔗 ***	★
2014	MS COCO	Flickr	330K	-	-	🅭	⬇	★ A 🕘 ▱ >_ ⚗
2014	YFCC100M	Flickr	99M	800K	-	🅭	⬇ ✎ ➜ 🔗 ****	⯪ 📷 🕘 📊 👤 🌍 🏷 ⚗
2015	TRECVID	mixed	-	220K	-	©	⬇ ✎ ➜ 🏆	★ >_ ◌
2015	MediaEval	mixed	6M	51K	1,4K	©	⬇ ✎ ➜ 🏆 🔗 *****	⯪ 📊 □ >_ ◌
2015	ImageCLEF	mixed	500K	-	-	©	⬇ ✎ ➜ 🏆	⯪ A 📊 ◌

The icons represent the following:

© Some or all content in dataset is copyrighted.

🅭 All content in dataset has a Creative Commons license.

ⓘ Content in dataset can be freely used on condition of citing the dataset paper.

⬇ Dataset has to be downloaded.

🛒 Dataset has to be purchased.

🚚 Dataset is delivered by mail.

✎ Dataset can only be obtained by accepting a license agreement.

➜ Dataset can only be obtained after creating an account.

🏆 Dataset can only be obtained by participating in a benchmark competition.

🔗 Dataset contains URLs to the content instead of the content itself.

★ / ⯪ Dataset is fully/partially annotated with class labels.

A Dataset contains content found by querying search engines with dictionary words.

📊 Dataset contains generated features.

>_ Dataset contains subtitles, transcripts, or captions describing the content.

☰ Dataset contains search engine click log data.

👤 Dataset contains user information.

📷 Dataset contains camera information.

🕘 Dataset contains timestamps.

🌍 Dataset contains locations.

🏷 Dataset contains tags.

□ Dataset contains object bounding boxes.

▱ Dataset contains object segmentations.

⚗ Dataset is still evolving.

◌ Dataset changes from year to year.

* The PASCAL training and development data can be freely downloaded, but the test data requires registration.

** Reduced-resolution images are included in the dataset, while full-resolution images must be downloaded separately.

*** The Sports-1M dataset has a CC license, though the videos it links to are hosted on YouTube and copyrighted.

**** The photos and videos have been uploaded to the cloud; like the metadata, the photo and video data can be mounted as a read-only network drive or downloaded.

***** Most MediaEval challenges include the media objects in their dataset, though some provide only URLs; in previous editions data had to be purchased and delivered by postal mail in order to participate in certain challenges.

nity of photographers that differs from the overall Flickr user base.

Licenses. The licenses themselves vary by CC type, with approximately 31.8% of the dataset marked appropriate for commercial use and 17.3% assigned the most liberal license requiring attribution for only the photographer who took the photo (see Table 3).

Content. The YFCC100M dataset includes a diverse collection of complex real-world scenes, ranging from 200,000 street-life-blogged photos by photographer Andy Nystrom (see Figure 3a) to snapshots of daily life, holidays, and events (see Figure 3b). To understand more about the visual content represented in the dataset, we used a deep-learning approach to detect a variety of concepts (such as people, animals, objects, food, events, architecture, and scenery). Specifically, we applied an off-the-shelf deep convolutional neural network[13] with seven hidden layers, five convolutional layers, and two fully connected layers. We employed the penultimate layer of the convolutional neural network output as the image-feature representation for training the visual-concept classifiers. We used Caffe[11] to train 1,570 classifiers, each a binary support vector machine, using 15 million photos we selected from the entire Flickr corpus; positive examples were crowd-labeled or handpicked by us based on targeted search/group results, while we drew negative examples from a general pool. We tuned the classifiers such that they achieved at least 90% precision on a held-out test set; Table 4 lists the top 25 detected concepts in both photos and videos (using the first frame). We see a diverse collection of visual concepts being detected, from outdoor to indoor images, sports to art, and nature to architecture. As we view the detected visual concepts as valuable to the research community, we released them as one of our expansion packs in July 2015.

Flickr makes little distinction between photos and videos, though videos do play a role in Flickr and in the YFCC100M dataset. While photos encode their content primarily through visual means, videos also do so through audio and motion. Only 5% of the videos in the YFCC100M dataset lack an audio track. From a manual examination of more than 120 randomly selected geotagged videos with audio, we found most of the audio tracks to be diverse; 60% of the videos were home-video style with little am-

Figure 1. Number of captured and uploaded media objects per month in the YFCC100M dataset, 2000–2014; the number of uploads closely follows the number of captures, with the number of more-recent uploads exceeding the number of captures as older media is uploaded.

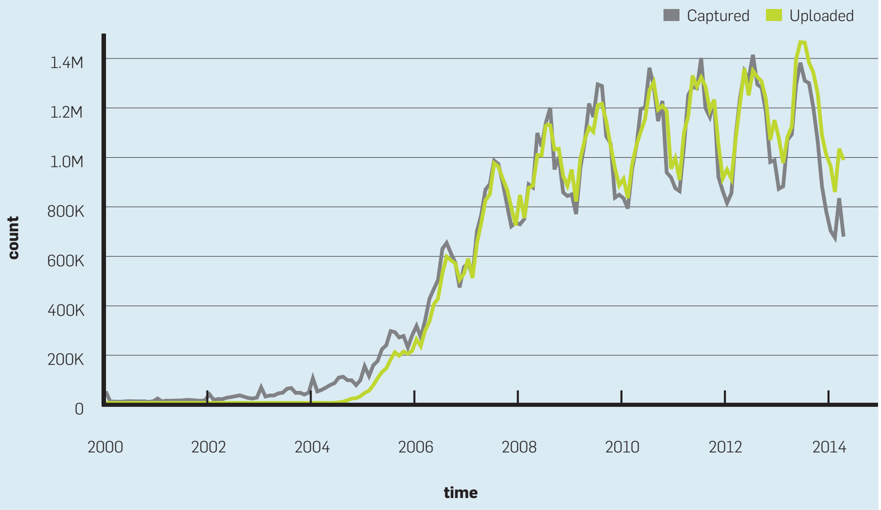


Figure 2. Global coverage of a sample of one million photos from the YFCC100M dataset; *One Million Creative Commons Geotagged Photos* by David A. Shamma 🅭 🅯 ⊜ (https://flic.kr/p/o1Ao2o).

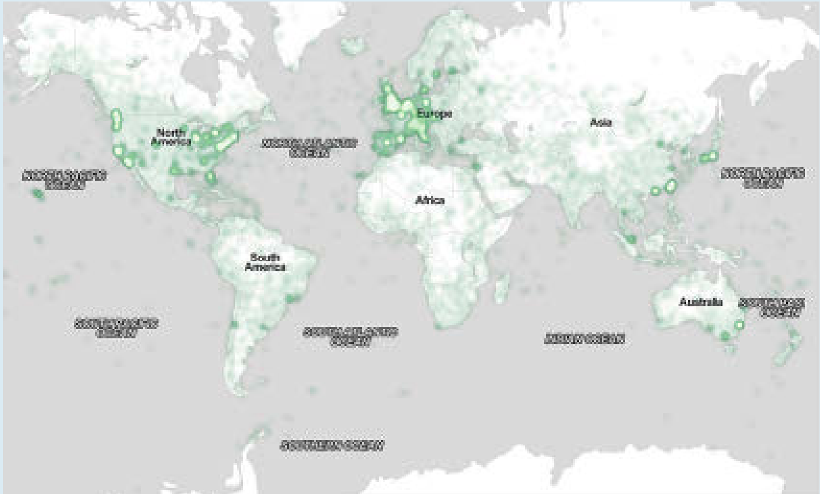


Table 2. Top 25 cameras and photo counts in the YFCC100M dataset; we merged the entries for the Canon models in the various markets, European (such as EOS 650D), American (such as EOS Rebel T4i), and Asian (such as EOS Kiss X6i).

Make	Model	Photos
Canon	EOS 400D	2,539,571
Canon	EOS 350D	2,140,722
Nikon	D90	1,998,637
Canon	EOS 5D Mark II	1,896,219
Nikon	D80	1,719,045
Canon	EOS 7D	1,526,158
Canon	EOS 450D	1,509,334
Nikon	D40	1,358,791
Canon	EOS 40D	1,334,891
Canon	EOS 550D	1,175,229
Nikon	D7000	1,068,591
Nikon	D300	1,053,745
Nikon	D50	1,032,019
Canon	EOS 500D	1,031,044
Nikon	D700	942,806
Apple	iPhone 4	922,675
Nikon	D200	919,688
Canon	EOS 20D	843,133
Canon	EOS 50D	831,570
Canon	EOS 30D	820,838
Canon	EOS 60D	772,700
Apple	iPhone 4S	761,231
Apple	iPhone	743,735
Nikon	D70	742,591
Canon	EOS 5D	699,381

bient noise; 47% had heavy ambient noise (such as people chatting in the background, traffic sounds, and wind blowing into the microphone); 25% of the sampled videos contained music, played in the background of the recorded scene or inserted during editing; 60% of the videos did not contain any human speech at all, while for the 40% that did contain human speech, 64% included multiple subjects and crowds in the background speaking to one another, often at the same time. The vocabulary of approximately 280,000 distinct user tags used as video annotations indeed shows tags describing audio content (`music`, `concert`, `festival`) and motion content (`timelapse`, `dance`, `animation`) were more frequently applied to videos than to photos. When comparing the videos in the dataset to those from YouTube, 2007–2012, we found YouTube videos are on average longer (Flickr: 39 seconds, YouTube: 214 seconds). This is likely due to the initial handling of videos on Flickr where their length until May 2013 was restricted to a maximum of 90 seconds; recent videos uploaded to Flickr tend to be longer.

Representativeness. In creating the dataset, we did not perform any special filtering other than to exclude photos and videos that had been marked as "screenshot" or "other" by the Flickr user. We did, however, include as many videos as possible, as videos represent a small percentage of media uploaded to Flickr, and a random selection would have led to relatively few videos being selected. We further included as many photos as possible associated with a geographic coordinate to encourage spatiotemporal research. These photos and videos together form approximately half of the dataset; the rest is CC photos we randomly selected from the entire pool of photos on Flickr.

To investigate whether the YFCC100M dataset includes a representative sample of real-world photography, we collected an additional random sample of 100 million public Flickr photos and videos, irrespective of their license, uploaded during the

Table 3. A breakdown of the 100 million photos and videos by their kind of 🅭 Creative Commons license, 🅯 attribution, ⊜ no derivatives, 🄎 share alike, and 🄏 noncommercial.

License	Photos	Videos
🅭🅯	17,210,144	137,503
🅭🅯🄎	9,408,154	72,116
🅭🅯⊜	4,910,766	37,542
🅭🅯🄏	12,674,885	102,288
🅭🅯🄏🄎	28,776,835	235,319
🅭🅯🄏⊜	26,225,780	208,668
Total	99,206,564	793,436

Table 4. The top 25 of 1,570 visually detected concepts in the YFCC100M dataset; photos and videos are counted by how often they include visual concepts.

Concept	Photos	Videos
Outdoor	44,290,738	266,441
Indoor	14,013,888	127,387
People	11,326,711	56,664
Nature	9,905,587	47,703
Architecture	6,062,789	11,289
Landscape	5,121,604	28,222
Monochrome	4,477,368	18,243
Sport	4,354,325	25,129
Building	4,174,579	7,693
Vehicle	3,869,095	13,737
Plant	3,591,128	11,815
Black and White	2,585,474	10,351
Animal	2,317,462	9,236
Groupshot	2,271,390	4,392
Sky	2,232,121	11,488
Water	2,089,110	15,426
Text	2,074,831	5,623
Road	1,796,742	12,808
Blue	1,658,929	10,273
Tree	1,641,696	6,808
Hill	1,448,925	6,075
Shore	1,439,950	8,602
Car	1,441,876	4,067
Head	1,386,667	8,984
Art	1,391,386	2,248

Figure 3. Two photos of real-world scenes from photographers in the YFCC100M dataset: (a) *IMG_9793: Streetcar (Toronto Transit)* by Andy Nystrom 🅭🅯🄏⊜ (https://flic.kr/p/jciMdz) and (b) *Celebrating our 6th wedding anniversary in Villa Mary* by Rita and Tomek 🅭🅯🄏🄎 (https://flic.kr/p/fCXEJi).

same time period as those included in the dataset. We then compared the relative frequency with which content and metadata are present in the YFCC100M dataset and in the random sample. We found the average difference in relative frequencies between two corresponding visual concepts, cameras, timestamps (year and month), and locations (countries) was only 0.02%, with an overall standard deviation of 0.1%. The maximum difference we observed was 3.5%, due to more videos in the YFCC100M having been captured in the U.S. than in the random sample (46.2% vs. 42.7%). While we found little individual difference between the relative frequency of use of any two corresponding cameras in the YFCC100M dataset and in the random sample, at most 0.5%, we did find the earlier mentioned tendency toward more professional DSLR cameras in the dataset rather than regular point-and-shoot cameras. This tendency notwithstanding, the dataset appears to exhibit similar characteristics as photos and videos in the entire Flickr corpus.

Features and annotations. Computing features for 100 million media objects is time consuming and computationally expensive. Not everyone has access to a distributed computing cluster, and performing even light processing of all the photos and videos on a single desktop machine could take several days. From our experience organizing benchmarks on image annotation and location estimation we noted accompanying the datasets with pre-computed features reduced the burden on the participating teams, allowing them to focus on solving the task at hand rather than on processing the data. As mentioned earlier, we are currently computing a variety of visual, aural, textual, and motion features for the dataset and have already released several of them. The visual features span the gamut of global (such as Gist), local (such as SIFT), and texture (such as Gabor) descriptors; the aural features include power spectrum (such as MFCC) and frequency (such as Kaldi) descriptors; the textual features refer to closed captions extracted from the videos; and the motion features include dense trajectories and shot boundaries. These features, as computed descriptors of the photos and videos, will be licensed without restriction under the CC0 (🅭🄍) license. Real-world data lacks well-formed annotations, thus highlighting the sense-making of the dataset itself as an area for investigation. Annotations (such as bounding boxes, segmentations of objects and faces, and image captions) are not yet available for the YFCC100M, though generating and releasing them is on our roadmap.

Ecosystem. The YFCC100M dataset has already given rise to an ecosystem of diverse challenges and benchmarks, similar to how ImageNet, PASCAL, and TRECVID have been used by the multimedia research community; for example, the MediaEval Placing Task,[3] an annual benchmark in which participants develop algorithms for estimating the geographic location where a photo or video was taken, is currently based on our dataset. To support research in multimedia event detection the YLI-MED corpus[1] was introduced September 2014 and consists of 50,000 handpicked videos from the YFCC100M that belong to events similar to those defined in the TRECVID MED challenge. Approximately 2,000 videos were categorized as depicting one of 10 target events and 48,000 as belonging to none of these events. Each video was further annotated with additional attributes like language spoken and whether it includes a musical score. The annotations also include degree of annotator agreement and average annotator confidence scores for the event categorization of each video. The authors said the main motivation for the creation of the YLI-MED corpus was to provide an open set without the license restrictions imposed on the original TRECVID MED dataset, while possibly also serving as, say, additional annotated data to improve the performance of current event detectors. Other venues incorporating the YFCC100M dataset are the ACM Multimedia 2015 Grand Challenge on Event Detection and Summarization and the ACM Multimedia 2015 MMCommons Workshop; the latter aims to establish a research community around annotating all 100 million photos and videos in the YFCC100M. The utility of the dataset is expected to grow as more features and annotations are produced and shared, whether by us or by others.

Strengths and limitations. Note the following strengths (⊕) and limitations (⊖) of the YFCC100M dataset.

⊕ *Design.* The YFCC100M dataset differs in design from most other multimedia collections. Its photos, videos, and metadata have been curated by us to be comprehensive and representative of real-world photography, expansive and expandable in coverage, free and legal to use, and intended to consolidate and supplant many of the existing datasets currently in use. We emphasize it does not challenge collections that are different and unique (such as PASCAL, TRECVID, ImageNet, and COCO); we instead aspire to make it the preferred choice for researchers, developers, and engineers with small and large multimedia needs that may be readily satisfied by the dataset, rather than having them needlessly collect their own data.

⊕ *Equality.* The YFCC100M dataset ensures data equality for research to facilitate reproduction, verification, and extension of scientific experiments and results.

⊕ *Volume.* Spanning 100 million media objects, the YFCC100M dataset is the largest public multimedia collection ever released.

⊕ *Modalities.* Unlike most existing collections, the YFCC100M dataset includes both photos and videos, making it a truly multimodal multimedia collection.

⊕ *Metadata.* Each media item is represented by a substantial amount of metadata, including some (such as machine tags, geotags, timestamps, and cameras) often absent from existing datasets. While social metadata is not included due to its ever-changing nature, it is readily obtained by querying the Flickr API.

⊕ *Licensing.* The vast majority of available datasets includes media for which licenses do not allow their use without explicit permission from the rightsholder. While fair-use exceptions may be invoked, they are, depending on the nature of use, generally not applicable to research and development performed by industry and/or for commercial gain; for example, a university spin-off offering a mobile product-recognition application that displays matching ImageNet images for each detected product would violate not only the ImageNet license agreement but also very likely copyright law. The YFCC100M dataset prevents potential violations by providing rules on how the dataset should be used to comply with licensing, attribution, and copyright.

⊖ *Annotations.* The YFCC100M dataset reflects the data as it is in the wild; there are lots of photos and videos, but they are currently associated with limited metadata and annotations. Note the dataset may not and/or cannot offer every kind of content, metadata, and annotation in existing collections (such as object segmentations, as in COCO, and broadcast videos, as in TRECVID), although our efforts and those that spring from the ecosystem being constructed around it will offer more depth and richness to the existing content, as well as new material, making it more complete and useful over time. While a lack of annotations represents a limitation of the dataset, it is also a challenge. With 100 million media objects, there are enough metadata labels for training and prediction of some attributes (such as geographic coordinates) and opportunities to create new methods for labeling and annotation through explicit crowdsourced work or through more modern techniques involving social community behaviors. In addition, given the plurality of existing Flickr datasets and the size of our dataset, some overlap is to be expected, such that existing annotations directly transfer over to the dataset. Of special note is COCO, of which one-third (approximately 100,000 images) is included in the YFCC100M. Over time we will also release the intersections with known datasets as expansion packs. We envision the intersection with existing datasets will allow researchers to expand on what is known and actively researched.

Guidelines and recommendations. Although we consider volume a strength of the YFCC100M dataset, it can also pose a weakness when insufficient computational power, memory, and/or storage is available. The compressed metadata of all 100 million media objects requires 12.5GB hard disk space and, at the default pixel resolution used on Flickr, the photos take up approximately 13.5TB and the videos 3.0TB. While the entire dataset—metadata and/or content—can be processed in minutes to hours on a distributed computing cluster, it might take a few hours to days on a single machine. It can still be used for experiments by focusing on only a subset of the data. Also, different fields of research, engineering, and science have different data requirements and evaluation needs, and all 100 million media objects in the YFCC100M dataset are not likely to be needed for each and every study. Note it is uncommon in the computer science literature for a paper to describe in enough detail how the dataset the authors used in their evaluations was created, effectively preventing others from fully replicating or comparing against the achieved results. One clear future challenge is how to ensure subsets of the dataset used in experiments can be reproduced accurately. To this end, we suggest researchers forego arbitrary selections from the YFCC100M dataset when forming a subset for use in their evaluations but rather use a principled approach that can be described succinctly. Such selection logic should examine one or both of two aspects of the dataset: the photos and videos in it are already randomized, and it consists of 10 consecutively numbered files. As such, a selection logic could be as simple as "We used the videos in the first four metadata files for training, those in the following three files for development, and those in the last three for testing" or in more complicated form as "From all photos taken in the U.S., we selected the first five million and performed tenfold cross-validation." Alternatively, the created subset can be made available for download described in terms of a set of object identifiers that index into the dataset. As an example, the organizers of the MediaEval Placing Task made the visual and aural features they extracted from the content available for download, in addition to the training and test sets. We envision the research community as likewise following this way of using and sharing the dataset.

Although we consider volume a strength of the YFCC100M dataset, it can also pose a weakness when insufficient computational power, memory, and/or storage is available.

Future directions. The YFCC100M dataset enables large-scale unsupervised learning, semi-supervised learning, and learning with noisy data. Beyond this, the dataset offers the opportunity to advance research, give rise to new challenges, and address existing problems; note the following challenges in which the YFCC100M dataset might play a leading role.

Artificial intelligence and vision. Large datasets have played a critical role in advancing computer vision research. ImageNet[5] has led the way toward advanced machine learning techniques like deep learning.[13] Computer vision as a field now seeks to do visual recognition by learning and benchmarking from large-scale data. The YFCC100M dataset provides new opportunities in this direction by developing new approaches that harness more than just pixels; for instance, new semantic connections can be made by inferring them through relating groups of visually co-occurring tags in images depicting similar scenes, where such efforts hitherto were hampered by lack of a sufficiently large dataset. Our expansion pack containing the detected visual concepts in photos and videos can help. However, visual recognition goes beyond image classification toward obtaining a deeper understanding of an image. Object localization, object interaction, face recognition, and image annotation are all important cornerstone challenges that will lead the way to retelling the story of an image—what is happening in the image and why it was taken. With a rich, diverse collection of image types, Flickr provides the groundwork for total scene understanding[14] in computer vision and artificial intelligence, a crucial task that can be expanded through the YFCC100M dataset, and even more once additional annotations are released.

Spatiotemporal computing. Beyond pixels, we emphasize time and location information as key components for research that aims to understand and summarize events. Starting with location, the geotagged photos and videos (approximately half of the dataset) provide a comprehensive snapshot of photographic activity in space and time. In the past, geotagged media was used to address a variety of research endeavors (such as location estimation,[9] event detection,[15] finding canonical views of places,[4] and visual reconstruction of the world[17]). Even styles and habits of understanding have been used to reverse-lookup authors online.[10] Geotagged media in the YFCC100M dataset can help push the boundaries in these research areas.

More data brings new discoveries and insights, even as it makes searching, organizing, and presenting the data and findings more difficult. The cameraphone has enabled people to capture more photos and videos than they can effectively organize. One challenge for the future is thus devising algorithms able to automatically and dynamically create albums for use on personal computers, cloud storage, or mobile devices, where desired media and moments of importance are easily surfaced based on simple queries. Harnessing the spatiotemporal context at capture time and query time will thus take a central role.

The challenge of automatically creating albums speaks toward social computing efforts aimed at understanding events in unstructured data through the reification of photos with space and time. While GPS-enabled devices are capable of embedding the precise time, location, and orientation of capture in a photo's metadata, this information (including seconds, hours, and sometimes even months) is often unavailable or out of sync. In addition, people frequently forget to adjust the camera clock to the correct time zone when traveling. Such issues pose problems for the accuracy of any kind of spatiotemporal data analysis, and new challenges in computational photography thus include devising algorithms that either fix or are resilient against erroneous information.

Digital culture and preservation. What we know to be user-generated content has grown from simple video uploads and bulletin board systems; life online has come to reflect culture. These large online collections tell a larger story about the world around us, from consumer reviews[21] on how people engage with the spaces around them to 500 years of scanned book photos and illustrations[12] that describe how concepts and objects have been visually depicted over time. Beyond archived collections, the pho-

While the entire dataset—metadata and/or content—can be processed in minutes to hours on a distributed computing cluster, it might take a few hours to days to do it on a single machine.

tostreams of individuals represent multiple facets of recorded visual information, from remembering moments and storytelling to social communication and self-identity.[19] How to preserve digital culture is a grand challenge of sensemaking and understanding digital archives from nonhomogeneous sources. Photographers and curators alike have contributed to the larger collection of Creative Commons images, yet little is know of how such archives will be navigated and retrieved or how new information can be discovered therein. The YFCC100M dataset offers avenues of computer science research in multimedia, information retrieval, and data visualization, in addition to the larger questions of how to preserve digital libraries.

Conclusion

Data is a core component of research and development in all scientific fields. In the field of multimedia, datasets are usually created for a single purpose and, as such, lack reusability. Moreover, datasets generally are not or may not be freely shared with others and, as such, also lack reproducibility, transparency, and accountability. That is why we released one of the largest datasets ever created, with 100 million media objects, published under a Creative Commons license. We curated the YFCC100M dataset to be comprehensive and representative of real-world photography, expansive and expandable in coverage, free and legal to use, and intentionally consolidate and supplant many existing collections. The YFCC100M dataset encourages improvement and validation of research methods, reduces the effort to acquire data, and stimulates innovation and potential new data uses. We have further provided rules on how the dataset should be used to comply with licensing, attribution, and copyright and offered guidelines on how to maximize compatibility and promote reproducibility of experiments with existing and future work.

Acknowledgments

We thank Jordan Gimbel and Kim Capps-Tanaka at Yahoo, Pierre Garrigues, Simon Osindero, and the rest of the Flickr Vision & Search team, Carmen Carrano and Roger Pearce at Lawrence Livermore National Laboratory, and Julia Bernd, Jaeyoung Choi, Luke Gottlieb, and Adam Janin at the International Computer Science Institute (ICSI). We are further thankful to ICSI for making its data publicly available in collaboration with Amazon. Portions of this work were performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 and supported by the National Science Foundation by ICSI under Award Number 1251276.

References

1. Bernd, J., Borth, D., Elizalde, B., Friedland, G., Gallagher, H., Gottlieb, L.R., Janin, A., Karabashlieva, S., Takahashi, J., and Won, J. The YLI-MED corpus: Characteristics, procedures, and plans. Computing Research Repository Division of arXiv abs/1503.04250 (Mar. 2015).
2. Borgman, C.L. The conundrum of sharing research data. *Journal of the American Society for Information Science and Technology 63*, 6 (Apr. 2012), 1059–1078.
3. Choi, J., Thomee, B., Friedland, G., Cao, L., Ni, K., Borth, D., Elizalde, B., Gottlieb, L., Carrano, C., Pearce, R., and Poland, D. The placing task: A large-scale geo-estimation challenge for social-media videos and images. In *Proceedings of the Third ACM International Workshop on Geotagging and Its Applications in Multimedia* (Orlando, FL, Nov. 3–7). ACM Press, New York, 2014, 27–31.
4. Crandall, D. J., Backstrom, L., Huttenlocher, D., and Kleinberg, J. Mapping the world's photos. In *Proceedings of the 18th IW3C2 International Conference on the World Wide Web* (Madrid, Spain, Apr. 20–24). ACM Press, New York, 2009, 761–770.
5. Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Miami, FL, June 20–25). IEEE Press, New York, 2009. 248–255.
6. Facebook, Ericsson, and Qualcomm. *A Focus on Efficiency.* Technical Report, Internet.org, 2013; https://web.archive.org/web/20150402101302/http://internet.org/efficiencypaper
7. Fienberg, S.E., Martin, M.E., and Straf, M.L. Eds. (National Research Council). *Sharing Research Data.* National Academy Press, Washington, D.C., 1985; http://www.nap.edu/catalog/2033/sharing-research-data
8. Good, J. How many photos have ever been taken?. *Internet Archive Wayback Machine*, Sept. 2011; https://web.archive.org/web/20150203215607/http://blog.1000memories.com/94-number-of-photos-ever-taken-digital-and-analog-in-shoebox
9. Hays, J. and Efros, A.A. IM2GPS: Estimating geographic information from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Anchorage, AK, June 23–28). IEEE Press, New York, 2008.
10. Hecht, B., Hong, L., Suh, B., and Chi, E. H. Tweets from Justin Bieber's heart: The dynamics of the location field in user profiles. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, Canada, May 7–12). ACM Press, New York, 2011, 237–246.
11. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R. B., Guadarrama, S., and Darrell, T. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM International Conference on Multimedia* (Orlando, FL, Nov. 3–7). ACM Press, New York, 2014, 675–678.
12. Kremerskothen, K. Welcome the Internet archive to the commons. Flickr, San Francisco, CA, Aug. 2014; https://blog.flickr.net/2014/08/29/welcome-the-internet-archive-to-the-commons/
13. Krizhevsky, A., Sutskever, I., and Hinton, G.E. ImageNet classification with deep convolutional neural networks. In *Proceedings of Advances in Neural Information Processing Systems* (Lake Tahoe, CA, Dec 3–8). Curran Associates, Red Hook, NY, 2012, 1097–1105.
14. Li, L., Socher, R., and Fei-Fei, L. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Miami, FL, June 20–25). IEEE Press, New York, 2009, 2036–2043.
15. Rattenbury, T., Good, N., and Naaman, M. Towards automatic extraction of event and place semantics from Flickr tags. In *Proceedings of the 30th ACM International Conference on Research and Development in Information Retrieval* (Amsterdam, the Netherlands, July 23–27). ACM Press, New York, 2007, 103–110.
16. Renear, A.H., Sacchi, S., and Wickett, K.M. Definitions of dataset in the scientific and technical literature. In *Proceedings of the 73rd Annual Meeting of the American Society for Information Science and Technology* (Pittsburgh, PA, Oct. 22–27). Association for Information Science and Technology, Silver Spring, MD, 2010, article 81.
17. Snavely, N., Seitz, S., and Szeliski, R. Photo tourism: Exploring photo collections in 3D. *ACM Transactions on Graphics 25*, 3 (July 2006), 835–846.
18. Swan, A. and Brown, S. *To Share or Not to Share: Publication and Quality Assurance of Research Data Outputs.* Technical Report. Research Information Network, London, U.K., 2008.
19. Van Dijck, J. Digital photography: Communication, identity, memory. *Visual Communication 7*, 1 (Feb. 2008), 57–76.
20. Wilson, M.L., Chi, E.H., Reeves, S., and Coyle, D. RepliCHI: The workshop II. In *Proceedings of the International Conference on Human Factors in Computing Systems, Extended Abstracts* (Toronto, Canada, Apr. 26–May 1). ACM Press, New York, 2014, 33–36.
21. Yelp. Yelp Dataset Challenge. Yelp, San Francisco, CA; http://yelp.com/dataset_challenge/
22. YouTube. YouTube press statistics. YouTube, San Bruno, CA; http://youtube.com/yt/press/statistics.html

Bart Thomee (bthomee@yahoo-inc.com) is a senior research scientist in the HCI Research Group at Yahoo Labs and Flickr in San Francisco, CA.

David A. Shamma (aymans@acm.org) is director of the HCI Research Group at Yahoo Labs and Flickr in San Francisco, CA.

Gerald Friedland (fractor@icsi.berkeley.edu) is director of the Audio and Multimedia Lab at the International Computer Science Institute in Berkeley, CA.

Benjamin Elizalde (bmartin1@andrew.cmu.edu) is a Ph.D. student at Carnegie Mellon University in Mountain View, CA; this work was done while he was at the International Computer Science Institute in Berkeley, CA.

Karl Ni (kni@iqt.org) is a program lead and senior data scientist at In-Q-Tel's Lab41 in Menlo Park, CA; this work was done while he was at Lawrence Livermore National Laboratory in Livermore, CA.

Douglas Poland (poland1@llnl.gov) is a principal investigator at the Lawrence Livermore National Laboratory in Livermore, CA.

Damian Borth (damian.borth@dfki.de) is head of the Multimedia Analysis & Data Mining Group at the German Research Center for Artificial Intelligence in Kaiserslautern, Germany; this work was done while he was at the International Computer Science Institute in Berkeley, CA.

Li-Jia Li (lijiali.vision@gmail.com) is head of research at Snapchat, Venice, CA; this work was done while she was at Yahoo Labs, San Francisco, CA.

Watch the authors discuss their work in this exclusive *Communications* video. http://cacm.acm.org/videos/yfcc100m-the-new-data-in-multimedia-research

DOI:10.1145/2869958

It turns out riding across America is more than a handy metaphor for building system software.

BY MICHAEL STONEBRAKER

The Land Sharks Are on the Squawk Box

KENNEBAGO, ME, SUMMER 1993. The "Land Sharks" are on the squawk box, Illustra (the company commercializing Postgres) is down to fumes, and I am on a conference call with the investors to try to get more money. The only problem is I am in Maine at my brother's fishing cabin for a family event while the investors are on a speakerphone (the squawk box) in California. There are eight of us in cramped quarters, and I am camped out in the bathroom trying to negotiate a deal. The conversation is depressingly familiar. They say more-money-lower-price; I say less-money-higher-price. We ultimately reach a verbal handshake, and Illustra will live to fight another day.

Negotiating with the sharks is always depressing. They are superb at driving a hard bargain; after all, that is what they do all day. I feel like a babe in the woods by comparison.

This article interleaves two stories (see Figure 1). The first is a cross-country bike ride my wife Beth and I took during the summer of 1988; the second is the design, construction, and commercialization of Postgres, which occurred over a 12-year period, from the mid-1980s to the mid-1990s. After telling both stories, I will draw a series of observations and conclusions.

Off to a Good Start

Anacortes, WA, June 3, 1988. Our car is packed to the gills, and the four of us (Beth; our 18-month-old daughter Leslie; Mary Anne, our driver and babysitter; and me) are squished in. It has been a stressful day. On the roof is the cause of it all—our brand-new tandem bicycle. We spent the afternoon in Seattle bike shops getting it repaired. On the way up from the Bay Area, Mary Anne drove into a parking structure lower than the height of the car plus the bike. Thankfully, the damage is repaired, and we are all set to go, if a bit frazzled. Tomorrow morning, Beth and I will start riding east up the North Cascades Scenic Highway; our destination, some 3,500 miles away, is Boston, MA. We have therefore christened our bike "Boston Bound."

It does not faze us that we have been on a tandem bike exactly once, nor that we have never been on a bike trip longer than five days. The fact we have never climbed mountains like the ones directly in front of us is equally undaunting. Beth and I are in high spirits; we are starting a great adventure.

Berkeley, CA, 1984. We have been working on Ingres for a decade. First, we built an academic prototype, then

» key insights

- **Explained is the motivation behind Postgres design decisions, as are "speedbumps" encountered.**
- **Riding a bicycle across America and building a computer software system are both long and difficult affairs, constantly testing personal fortitude along the way.**
- **Serendipity played a major role in both endeavors.**

made it fully functional, and then started a commercial company. However, Ingres Corporation, which started with our open source code base four years ago in 1980, has made dramatic progress, and its code is now vastly superior to the academic version. It does not make any sense to continue to do prototyping on our software. It is a painful decision to push the code off a cliff, but at that point a new DBMS is born. So what will Postgres be?

One thing is clear: Postgres will push the envelope on data types. By now I have read a dozen papers of the form: "The relational model is great, so I tried it on [pick a vertical application]. I found it did not work, and to fix the situation, I propose we add [some new idea] to the relational model."

Some chosen verticals were geographic information systems (GISs), computer-aided design (CAD), and library information systems. It was pretty clear to me that the clean, simple relational model would turn into a complete mess if we added random functionality in this fashion. One could think of this as "death by 100 warts."

The basic problem was the existing relational systems—specifically Ingres and System R—were designed with business data processing users in mind. After all, that was the major DBMS market at the time, and both collections of developers were trying to do better than the existing competition, namely IMS and Codasyl, on this popular use case. It never occurred to us to look at other markets, so RDBMSs were not good at them. However, a research group at the University of California at Berkeley, headed by Professor Pravin Varaiya, built a GIS on top of Ingres, and we saw firsthand how painful it was. Simulating points, lines, polygons, and line groups on top of the floats, integers, and strings in Ingres was not pretty.

Anacortes, WA: Day 1 - June 4, 1988

ALL PHOTOS BY MARY ANNE CASE

It was clear to me that one had to support data types appropriate to an application and that required user-defined data types. This idea had been investigated earlier by the programming language community in systems like EL1, so all I had to do was apply it to the relational model. For example, consider the following SQL update to a salary, stored as an integer

```
Update Employee set (salary
= salary + 1000) where name =
'George'
```

To process it, one must convert the character string `1000` to an integer using the library function string-to-integer and then call the integer + routine from the C library. To support this command with a new type, say, `foobar`, one must merely add two functions, `foobar-plus` and `string-to-foobar`, and then call them at the appropriate times. It was straightforward to add a new DBMS command, `ADDTYPE`, with the name of the new data type and conversion routines back and forth to ASCII. For each desired operator on this new type, one could add the name of the operator and the code to call to apply it.

The devil is, of course, always in the details. One has to be able to index the new data type using B-trees or hashing. Indexes require the notion of less-than and equality. Moreover, one needs commutativity and associativity rules to decide how the new type can be used with other types. Lastly, one must also deal with predicates of the form:

```
not salary < 100
```

This is legal SQL, and every DBMS will flip it to

```
salary ≥ 100
```

So one must define a negator for every operator, so this optimization is possible.

We had prototyped this functionality in Ingres,[8] and it appeared to work, so the notion of abstract data types (ADTs) would clearly be a cornerstone of Postgres.

Winthrop, WA, Day 3. My legs are throbbing as I lay on the bed in our motel room. In fact, I am sore from the hips down but elated. We have been riding since 5 A.M. this morning; telephone pole by telephone pole we struggled uphill for 50 miles. Along the way, we rose 5,000 feet into the Cascades,

Figure 1. The two timelines: cross-country bike ride and Illustra/Postgres development.

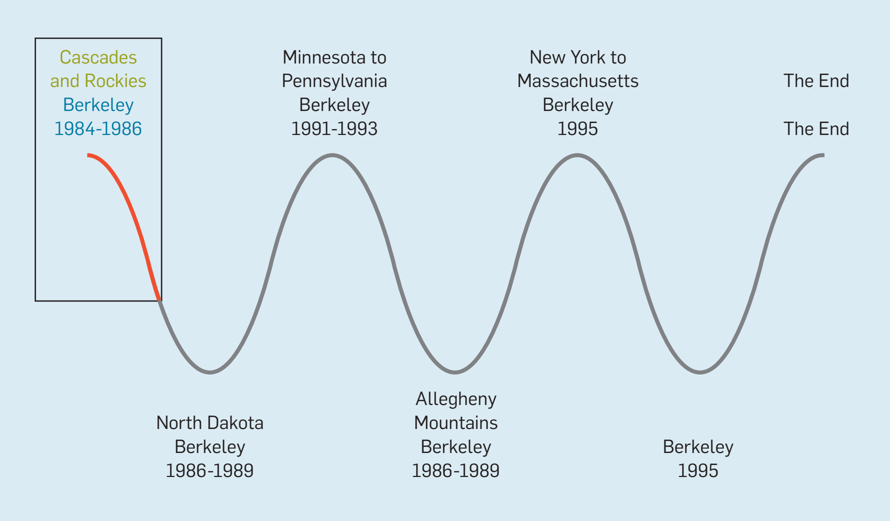


Figure 2. Correlated data illustrating why data users need referential integrity.

dname	floor	sq. ft.	budget
Shoe	3	500	40,000
Candy	2	800	50,000

name	dept	salary	age
Bill	Shoe	2,000	40
Art	Candy	3,000	35
Sam	Shoe	1,500	25
Tom	Shoe	1,000	23

putting on every piece of clothing we brought with us. Even so, we were not prepared for the snowstorm near the top of the pass. Cold, wet, and tired, we finally arrived at the top of the aptly named Rainy Pass. After a brief downhill, we climbed another 1,000 feet to the top of Washington Pass. Then it was glorious descent into Winthrop. I am now exhausted but in great spirits; there are many more passes to climb, but we are over the first two. We have proved we can do the mountains.

Berkeley, CA, 1985–1986. Chris Date wrote a pioneering paper[1] on referential integrity in 1981 in which he defined the concept and specified rules for enforcing it. Basically, if one has a table

```
Employee (name, salary, dept,
age) with primary key “name”
```

and a second table

```
Dept (dname, floor) with a
primary key “dname”
```

then the attribute dept in `Employee` is a foreign key; that is, it references a primary key in another table; an example of these two tables is shown in Figure 2. In this case, what happens if one deletes a department from the `dept` table?

For example, deleting the candy department will leave a dangling reference in the Employee table for everybody who works in the now-deleted department. Date identified six cases concerning what to do with insertions and deletions, all of which can be specified by a fairly primitive if-then rule system. Having looked at programs in Prolog and R1, I was very leery of this approach. Looking at any rule program with more than 10 statements, it is very difficult to figure out what it does. Moreover, such rules are procedural, and one can get all kinds of weird behavior depending on the order in which rules are invoked. For example, consider the following two (somewhat facetious) rules:

```
If Employee.name = ‘George’
Then set Employee.dept = ‘shoe’
If Employee.salary > 1000 and
Employee.dept = ‘candy’
Then set Employee.salary = 1000
```

Consider an update that moves `George` from the shoe department to the candy department and updates his salary to `2000`. Depending on the order the two rules are processed, one will get different final answers. Notably, if the rules are executed in the order here, then George will ultimately have a salary of `2000`; if the rule order is reversed, then his ending salary will be `1000`. Having order-dependent rule semantics is pretty awful.

A fundamental tenet of the relational model is the order of evaluation of a query, including the order in which records are accessed, is up to the system. Hence, one should always give the same final result, regardless of the query plan chosen for execution. As one can imagine, it is trivial to construct collections of rules that give different answers for different query plans—obviously undesirable system behavior.

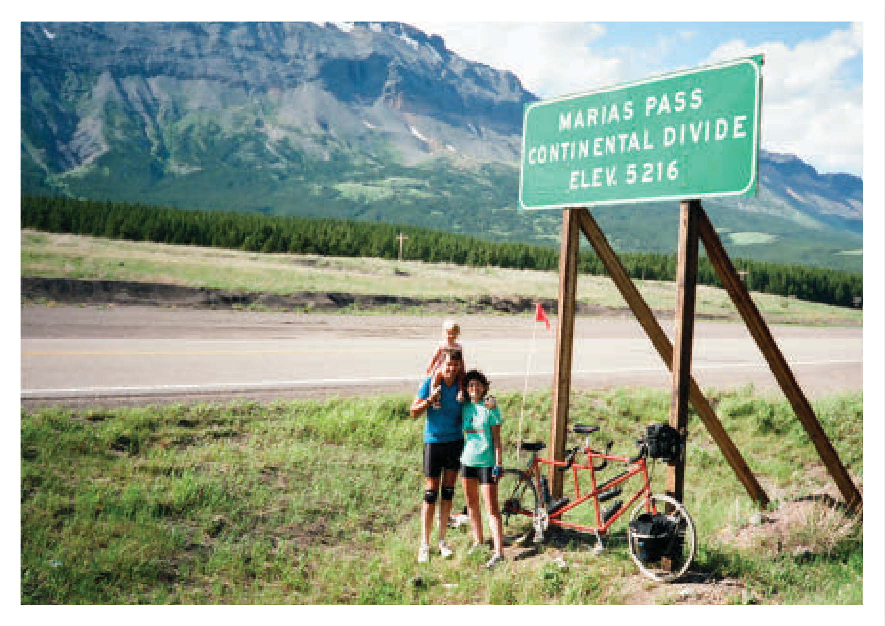
I spent many hours over a couple of years looking for something else. Ultimately, my preferred approach was to add a keyword `always` to the query language. Hence, any utterance in the query language should have the semantics that it appears to be continually running. For example, if Mike must have the same salary as Sam, then the following `always` command will do the trick

```
Always update Employee, E
set salary = E.salary
where Employee.name = ‘Mike’
and E.name = ‘Sam’
```

Whenever Mike receives a salary adjustment, this command will kick in and reset his salary to that of Sam. Whenever Sam gets a raise, it will be propagated to Mike. Postgres would have this `always` command and avoid (some of) the ugliness of an if-then rules system. This was great news; Postgres would try something different that has the possibility of working.

Marias Pass, MT, Day 15. I cannot believe it. We round a corner and see the sign for the top of the pass. We are at the Continental Divide! The endless climbs in the Cascades and the Rockies are behind us, and we can see the Great Plains stretching out in front of us. It is now downhill to Chicago! To celebrate this milestone, we pour a small vial of Pacific Ocean water we have been carrying since Anacortes to the east side of the pass where it will ultimately flow into the Gulf of Mexico.

Berkeley, CA, 1986. My experience with Ingres convinced me a database log for recovery purposes is tedious and difficult to code. In fact, the gold standard specification is in C. Mohan



Marias Pass, MT: Day 15

et al.[3] Moreover, a DBMS is really two DBMSs, one managing the database as we know it and a second one managing the log, as in Figure 3. The log is the actual system of record, since the contents of the DBMS can be lost. The idea we explored in Postgres was to support time travel. Instead of updating a data record in place and then writing both the new contents and the old contents into the log, could we leave the old record alone and write a second record with the new contents in the actual database? That way the log would be incorporated into the normal database and no separate log processing would be required, as in Figure 4. A side benefit of this architecture is the ability to support time travel, since old records are readily queryable in the database. Lastly, standard accounting systems use no overwrite in their approach to record keeping, so Postgres would be compatible with this tactic.

At a high level, Postgres would make contributions in three areas: an ADT system, a clean rules system based on the `always` command, and a time-travel storage system. Much of this functionality is described in Stonebraker and Rowe.[6,7] For more information on the scope of Postgres, one can consult the video recording of the colloquium celebrating my 70th birthday.[2] We were off and running with an interesting technical plan.

First Speedbumps

Drake, ND, Day 26. We are really depressed. North Dakota is bleak. The last few days have been the following monotony:

See the grain elevator ahead that signifies the next town
Ride for an hour toward the elevator
Pass through the town in a few minutes
See the next grain elevator …

However, it is not the absence of trees (we joke the state tree of North Dakota is the telephone pole) and the bleak landscape that is killing us. Normally, one can simply sit up straight in the saddle and be blown across the state by the prevailing winds, which are typically howling from west to east. They are howling all right, but the weather this summer is atypical. We are experiencing gale-force winds blowing east to west, straight in our faces. While we are expecting to be blown along at 17–18 miles per hour, we are struggling hard to make 7. We made only 51 miles today and are exhausted. Our destination was Harvey, still 25 miles away, and we are not going to make it. More ominously, the tree line (and Minnesota border) is still 250 miles away, and we are not sure how we will get there. It is all we can do to refuse a ride from a driver in a pickup truck offering to transport us down the road to the next town.

The food is also becoming problematic. Breakfast is dependable. We find a town, then look for the café (often the only one) with the most pickup trucks. We eat from the standard menu found in all such restaurants. However, dinner is getting really boring. There is a standard menu of fried fare; we yearn for pasta and salad, but it is never on the menu.

We have established a routine. It is in the 80s or 90s Fahrenheit every day, so Beth and I get on the road by 5 A.M. Mary Anne and Leslie get up much later; they hang around the motel, then pass us on the road going on to the town where we will spend the night. When we arrive at the new motel, one of us relieves Mary Anne while the other tries to find someplace with food we are willing to eat. Although we have camping equipment with us, the thought of an air mattress after a hard day on the road is not appealing. In fact, we never camp. Leslie has happily accommodated to this routine, and one of her favorite words, at 18-months old, is "ice machine." Our goal is 80 miles a day in the flats and 60 miles a day in the mountains. We ride six days per week.

Berkeley, CA, 1986. I had a conversation with an Ingres customer shortly

Figure 3. Traditional DBMS crash recovery.

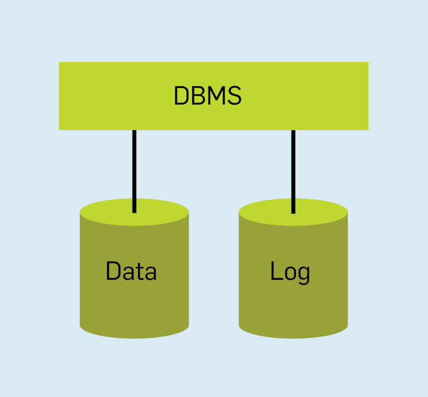
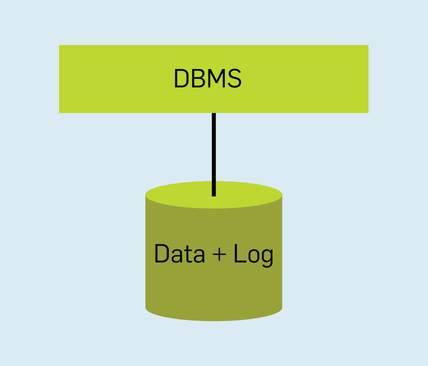


Figure 4. Postgres picture: No overwrite.



Drake, ND: Day 26

after he implemented date and time as a new data type (according to the American National Standards Institute specification). He said, "You implemented this new data type incorrectly." In effect, he wanted a different notion of time than what was supported by the standard Gregorian calendar. More precisely, he calculated interest on Wall Street-type financial bonds, which give the owner the same amount of interest, regardless of how long a month is. That is, he wanted a notion of bond time in which March 15 minus February 15 is always 30 days, and each year is divided into 30-day months. Operationally, he merely wanted to overload temporal subtraction with his own notion. This was impossible in Ingres, of course, but easy to do in Postgres. It was a validation that our ADTs are a good idea.

Berkeley, CA, 1986. My partner, the "Wine Connoisseur," and I have had a running discussion for nearly a year about the Postgres data model. Consider the Employee-Dept database noted earlier. An obvious query is to join the two tables, to, say, find the names and floor number of employees, as noted in this SQL command:

```
Select E.name, D.floor
From Employee E, Dept D
Where E.dept = D.dname
```

In a programming language, this task would be coded procedurally as something like (see code section 1).

A programmer codes an algorithm to find the desired result. In contrast, one tenet of the relational model is programmers should state what they want without having to code a search algorithm. That job falls to the query optimizer, which must decide (at scale) whether to iterate over Employee first or over Dept or to hash both tables on the join key or sort both tables for a merge or ...

My Ingres experience convinced me optimizers are really difficult, and the brain surgeon in any database company is almost certainly the optimizer specialist. Now we were considering extending the relational model to support more complex types. In its most general form, we could consider a column whose fields were pointers to arrays of structures of ... I could not wrap my brain around designing a query optimizer for something this complex. On the other hand, what should we discard? In the end, The Wine Connoisseur and I are depressed as we choose a design point with rudimentary complex objects. There is still a lot of code to support the notion we select.

Code section 1.

```
For E in Employee {
         For D in Dept {
                  If (E.dept = D.dname) then add-to-result;
         }
}
```

Berkeley, CA, 1987. The design of time travel in Postgres is in Stonebraker.[5] Although this is an elegant construct in theory, making it perform well in practice is tricky. The basic problem is the two databases in the traditional architecture of Figure 3 are optimized very differently. The data is "read-optimized" so queries are fast, while the log is "write-optimized" so one can commit transactions rapidly. Postgres must try to accomplish both objectives in a single store; for example, if 10 records are updated in a transaction, then Postgres must force to disk all the pages on which these records occurred at commit time. Otherwise, the DBMS can develop "amnesia," a complete no-no. A traditional log will group all the log records on a small collection of pages, while the data records remain read-optimized. Since we are combining both constructs into one storage structure, we have to address a tricky record placement problem to try to achieve both objectives, and our initial implementation is not very good. We spend a lot of time trying to fix this subsystem.

Berkeley, CA, 1987. The Wine Connoisseur and I had written Ingres in C and did not want to use it again. That sounded too much like déjà vu. However, C++ was not mature enough, and other language processors did not run on Unix. By this time, any thought of changing operating systems away from Unix was not an option; all the Berkeley students were being trained on Unix, and it was quickly becoming the universal academic operating system. So we elected to drink the artificial intelligence Kool-Aid and started writing Postgres in Lisp.

Once we had a rudimentary version of Postgres running, we saw what a disastrous performance mistake this was—at least one-order-of-magnitude performance penalty on absolutely everything. We immediately tossed portions of the code base off the cliff and converted everything else to C. We were back to déjà vu (coding in C), having lost a bunch of time, but at least we had learned an important lesson: Do not jump into unknown water without dipping your toe in first. This was the first of several major code rewrites.

Berkeley, CA, 1988. Unfortunately, I could not figure out a way to make our `always` command general enough to at least cover Chris Date's six referential integrity cases. After months of trying, I gave up, and we decided to return to a more conventional rule system. More code over the cliff, and more new functionality to write.

In summary, for several years we struggled to make good on the original Postgres ideas. I remember this time as a long "slog through the swamp."

The endless climbs in the Cascades and the Rockies are behind us, and we can see the Great Plains stretching out in front of us.

Another High

Carrington, ND, the next afternoon. It is really hot, and I am dead tired. I am on "Leslie duty," and after walking though town, we are encamped in the ubiquitous (and air-conditioned) local Dairy Queen. I am watching Leslie slurp down a soft serve, feeling like "God is on our side," as serendipity has intervened in a big way today. No, the wind is still blowing at gale force from east to west. Serendipity came in the form of my brother. He has come from Maine to ride with us for a week. Mary Anne picked him and his bicycle up at the Minot airport yesterday afternoon. He is fresh and a very, very strong rider. He offers to break the wind for us, like you see in bicycle races. With some on-the-job training (and a couple of excursions into the wheat fields when we hit his rear wheel), Beth and I figure out how to ride six inches behind his rear wheel. With us trying to stay synchronized with a faster-slower-faster dialog, we rode 79 miles today. It is now clear we are "over the hump" and will get out of North Dakota, a few inches behind my brother's wheel, if necessary.

Battle Lake, MN, July 4, 1988, Day 30. We are resting today and attending the annual 4th of July parade in this small town. It is quite an experience—the local band, clowns giving out candy, which Leslie happily takes, and Shriners in their little cars. It is a slice of Americana I will never forget. Rural America has taken very good care of us, whether by giving our bike a wide berth when passing, willingly cashing our travelers checks, or alerting us to road hazards and detours.

Berkeley, CA, 1992. In my experience, the only way to really make a difference in the DBMS arena is to get your ideas into the commercial marketplace. In theory, one could approach the DBMS companies and try to convince them to adopt something new. In fact, there was an obvious "friendly" one—Ingres Corporation—although it had its own priorities at the time.

I have rarely seen technology transfer happen in this fashion. There is a wonderful book by Harvard Business School professor Clayton Christiansen called *The Innovators Dilemma*. His thesis is technology disruptions are very challenging for the incumbents. Specifically, it is very difficult for established vendors with old technology to morph to a new approach without losing their customer base. Hence, disruptive ideas do not usually find a receptive audience among the established vendors, and launching a startup to prove one's ideas is the preferred option.

By mid-1992 I had ended my association with Ingres and a sufficient amount of time had passed that I was free of my non-compete agreement with the company. I was ready to start a commercial Postgres company and contacted my friend the "Tall Shark." He readily agreed to be involved. What followed was a somewhat torturous negotiation of terms with the "Head Land Shark," with me getting on-the-job training in the terms and conditions of a financing contract. Finally, I understood what I was being asked to sign. It was a difficult time, and I changed my mind more than once. In the end, we had a deal, and Postgres had $1 million in venture capital to get going.

Right away two stars from the academic Ingres team—"Quiet" and "EMP1"—moved over to help. They were joined shortly thereafter by "Triple Rock," and we had a core implementation team. I also reached out to "Mom" and her husband, the "Short One," who also jumped on board, and we were off and running, with the Tall Shark acting as interim CEO. Our initial jobs were to whip the research code line into commercial shape, convert the query language from QUEL to SQL, write documentation, fix bugs, and clean up the "cruft" all over the system.

Emeryville, CA, 1993. After a couple of naming gaffes, we chose Illustra, and our goal was to find customers willing to use (and hopefully pay for) a system from a startup. We had to find a compelling vertical market, and the one we chose to focus on was geographic data. Triple Rock wrote a collection of abstract data types for points, lines, and polygons with the appropriate functions (such as distance from a point to a line).

After an infusion of capital from new investors, including the "Entrepreneur-Turned-Shark," we again ran out of money, prompting the phone call from Kennebago noted earlier. Soon thereafter, we were fortunate to be able to hire the "Voice-of-Experience" as the real CEO, and he recruited "Smooth" to be VP of sales, complementing "Uptone," who was previously hired to run marketing. We had a real company with a well-functioning engineering team and world-class executives. The future was looking up.

Luddington, MI, Day 38. We walk Boston Bound off the Lake Michigan ferry and start riding southeast. The endless Upper Midwest is behind us; it is now less than 1,000 miles to Boston! Somehow it is reassuring that we have no more more water to cross. We are feeling good. It is beginning to look like we might make it.

The High Does Not Last

Ellicottville, NY, Day 49. Today was a very bad day. Our first problem occurred while I was walking down the stairs of the hotel in Corry, PA, in my bicycle cleats. I slipped on the marble floor and wrenched my knee. Today, we had only three good legs pushing Boston Bound along. However, the bigger problem is we hit the Alleghany Mountains. Wisconsin, Michigan, and Ohio are flat. That easy riding is over, and our bicycle maps are sending us up and then down the same 500 feet over and over again. Also, road planners around here do not seem to believe in switchbacks; we shift into the lowest of our 21 gears to get up some of these hills, and it is exhausting work. We are not, as you can imagine, in a good mood. While Beth is putting Leslie to bed, I ask the innkeeper in Ellicottville a simple question, "How do we get to Albany, NY, without climbing all these hills?"

Emeryville, CA, 1993. Out of nowhere comes our first marketing challenge. It was clear our "sweet spot" was any application that could be accelerated through ADTs. We would have an unfair advantage over any other DBMS whenever this was true. However, we faced a Catch-22 situation. After a few "lighthouse" customers, the more cautious ones clearly said they wanted GIS functionality from the major GIS vendors (such as ArcInfo and MapInfo). We needed to recruit application companies in specific vertical markets and convince them to restructure the inner core of their software into ADTs—not a trivial task. The application vendors naturally said, "Help me understand why we should engage with you in this joint project." Put more bluntly, "How many customers do you have and how much money can I expect to make from this additional distribution channel for my product?" That is, we viewed this rearchitecting as a game-changing technology shift any reasonable application vendor should embrace. However, application vendors viewed it as merely a new distribution channel. This brought up the Catch-22: Without ADTs we could not get customers, and without customers we could not get ADTs. We were pondering this depressing situation, trying to figure out what to do, when the next crisis occurred.

Oakland, CA, 1994. We were again out of money, and the Land Sharks announced we were not making good progress toward our company goals. Put more starkly, they would put up additional capital, but only at a price lower than the previous financing round. We were facing the dreaded "down round." After the initial (often painful) negotiation, when ownership is a zero-sum game between the company team and the Land Sharks, the investors and the team are usually on the same side of the table. The goal is to build a successful company, raising money when necessary at increasing stock prices. The only disagreement concerns the "spend." The investors naturally want you to spend more to make faster progress, since that would ensure them an increasing percentage ownership of the company. In contrast, the team wants to "kiss every nickel" to minimize the amount of capital raised and maximize their ownership. Resolving these differences is usually pretty straightforward. When a new round of capital is needed, a new investor is typically brought in to set the price of the round. It is in the team's interest to make this as high as possible. The current investors will be asked to support the round, by adding their pro-rata share at whatever price is agreed on.

However, what happens if the current investors refuse to support a new round at a higher price? Naturally, a new investor will follow the lead of the current ones, and a new lower price is established. At this point, there is a clause in most financing agreements that the company must ex post facto reprice the previous financing round (or rounds) down to the new price. As you can imagine, a down round is incredibly dilutive financially to the team, who would naturally say, "If you want us to continue, you need to top up our options." As such, the discussion becomes a three-way negotiation among the existing investors, the new investors, and the team. It is another painful zero-sum game.

When the dust settled, the Illustra employees were largely made whole through new options, the percentage ownership among the Land Sharks had changed only slightly, and the whole process left a bitter taste. Moreover, management had been distracted for a couple of months. The Land Sharks seemed to be playing some sort of weird power game with each other I did not understand. Regardless, Illustra will live to fight another day.

The Future Looks Up (Again)

Troy, NY, Day 56. The innkeeper in Ellicottville tells us what was obvious to anybody in the 19th century moving goods between the eastern seaboard and the middle of the country. He said, "Ride north to the Erie Canal and hang a right." After a pleasant (and flat) ride down the Mohawk Valley, we arrive at Troy and see our first road sign for Boston, now just 186 miles away. The end is three days off! I am reminded of a painted sign at the bottom of Wildcat Canyon Road in Orinda, CA, at the start of the hill that leads back to Berkeley from the East Bay. It says simply "The Last Hill." We are now at our last hill. We need only climb the Berkshires to Pittsfield, MA. It is then easy riding to Boston.

Oakland, CA, 1995. Shortly after our

down round and the Catch-22 on ADTs, serendipity occurred once more. The Internet was taking off, and most enterprises were trying to figure out what to do with it. Uptone executes a brilliant repositioning of Illustra. We became the "database for cyberspace," capable of storing Internet data like text and images. He additionally received unbelievable airtime by volunteering Illustra to be the database for "24 Hours in Cyberspace," a worldwide effort by photojournalists to create one Web page per hour, garnering a lot of positive publicity. Suddenly, Illustra was "the new thing," and we were basking in reflected glory. Sales picked up and the future looked bright. The Voice-of-Experience stepped on the gas and we hired new people. Maybe this was the beginning of the widely envied "hockey stick of growth." We were asked to do a pilot application for a very large Web vendor, a potentially company-making transaction. However, we were also in a bake-off with the traditional RDBMSs.

The Good Times Do Not Last Long

Oakland, CA, 1995. Reality soon rears its ugly head. Instead of doing a benchmark on a task we were good at (such as geographic search or integrating text with structured data and images), the Web vendor decided to compare us on a traditional bread-and-butter transaction-processing use case, in which the goal is to perform as many transactions per second as you can on a standard banking application. It justified its choice by saying, "Within every Internet application, there is a business data-processing sub-piece that accompanies the multimedia requirements, so we are going to test that first."

There was immediately a pit in my stomach because Postgres was never engineered to excel at online transaction processing (OLTP). We were focused on ADTs, rules, and time travel, not on trying to compete with current RDBMSs on the turf for which they had been optimized. Although we were happy to do transactions, it was far outside our wheelhouse. Our performance was going to be an order-of-magnitude worse than what was offered by the traditional vendors we were competing against. The problem is a collection of architectural decisions I made nearly a decade earlier that are not easy to undo; for example, Illustra ran as an operating system process for each user. This architecture was well understood to be simple to implement but suffers badly on a highly concurrent workload with many users doing simple things. Moreover, we did not compile query plans aggressively, so our overhead to do simple things was high. When presented with complex queries or use cases where our ADTs were advantageous, these shortcomings are not an issue. But when running simple business data processing, we were going to lose, and lose badly.

Wollaston Beach, MA: Day 59

We were stuck with the stark reality that we must dramatically improve transaction-processing performance, which will be neither simple nor quick. I spent hours with the Short One trying to find a way to make it happen without a huge amount of recoding, energy, cost, and delay. We drew a blank. Illustra would have to undergo a costly rearchitecting.

The Stories End

Sutton, MA, Day 59. Massachusetts roads are poorly marked, and we have never seen more discourteous drivers. Riding here is not pleasant, and we cannot imagine trying to navigate Boston Bound into downtown Boston, let alone find someplace where we can access the ocean. We settle instead for finishing at Wollaston Beach in Quincy, MA, approximately 10 miles south of Boston. After the perfunctory dragging of our bike across the beach and dipping the front wheel in the surf, we are done. We drink a glass of champagne at a beachside café and ponder what happens next.

Oakland, CA, February 1996. Serendipity occurs yet again. One of the vendors we competed against on the Web vendor's benchmark has been seriously threatened by the benchmark. It saw Illustra would win a variety of Internet-style benchmarks hands-down, and Web vendors would have substantial requirements in this area. As a result, it elected to buy Illustra. In many ways, this was the answer to all our issues. The company had a high-performance OLTP platform into which we could insert the Illustra features. It was also a big company with sufficient "throw-weight" to get application vendors to add ADTs to its system. We consummated what we thought was a mutually beneficial transaction and set to work putting Illustra features into its engine.

I will end the Illustra story here, even though there is much more to tell, most of it fairly dark—a shareholder lawsuit, multiple new CEOs, and ultimately a sale of the company. The obvious takeaway is to be very careful about the choice of company you agree to marry.

Why a Bicycle Story?

You might wonder why I would tell this bicycling story. There are three reasons. First, I want to give you an algorithm for successfully riding across America.

```
Until (Ocean) {
      Get up in the morning;
```

```
        Ride east;
        Persevere, and over-
        come any obstacles that
        arise;
         }
```

It is clear that following this algorithm will succeed. Sprinkle in some serendipity if it occurs. Now abstract it a bit by substituting `goal` for `Ocean` and `Appropriate Action` for `Ride east`

```
Until (Goal) {
        Get up in the morning;
        Appropriate action;
        Persevere, and over-
        come any obstacles that
        arise;
         }
```

Since I will be using this algorithm again, I will make it a macro

```
Make-it-happen (Goal);
```

With this preamble, I can give a thumbnail sketch of my résumé, circa 1988.

```
Make-it-happen (Ph.D.);
Make-it-happen (Tenure);
Make-it-happen (Ocean);
```

In my experience, getting a Ph.D. (approximately five years) is an example of this algorithm at work. There are ups (passing prelims), downs (failing quals the first time), and a lot of slog through the swamp (writing a thesis acceptable to my committee). Getting tenure (another five years) is an even less pleasant example of this algorithm at work.

This introduces the second reason for presenting the algorithm. The obvious question is, "Why would anybody want to do this bicycle trip?" It is long and very difficult, with periods of depression, elation, and boredom, along with the omnipresence of poor food. All I can say is, "It sounded like a good idea, and I would go again in a heartbeat." Like a Ph.D. and tenure, it is an example of `make-it-happen` in action. The obvious conclusion to draw is I am programmed to search out `make-it-happen` opportunities and get great satisfaction from doing so.

I want to transition here to the third reason for telling the bicycle story. Riding across America is a handy metaphor for building system software. Let me start by writing down the algorithm for building a new DBMS (see code section 2).

The next question is, "How do I come up with a new idea?" The answer is, "I don't know." However, that will not stop me from making a few comments. From personal experience, I never come up with anything by going off to a mountaintop to think. Instead, my ideas come from two sources: talking to real users with real problems and then trying to solve them. This ensures I come up with ideas that somebody cares about and the rubber meets the road and not the sky. The second source is to bounce possibly good (or bad) ideas off colleagues that will challenge them. In summary, the best chance for generating a good idea is to spend time in the real world and find an environment (like MIT/CSAIL and Berkeley/EECS) where you will be intellectually challenged.

If your ideas hold water and you have a working prototype, then you can proceed to phase two, which has a by-now-familiar look (see code section 3).

As with other system software, building a new DBMS is difficult, takes a decade or so, and involves periods of elation and depression. Unlike bicycling across America, which takes just muscles and perseverance, building a new DBMS involves other challenges. In the prototype phase, one must figure out new interfaces, both internal and to applications, as well as to the operating system, networking, and persistent storage. In my experience, getting them right the first time is unusual. Unfortunately, one must often build it first to see how one should have built it. You will have to throw code away and start again, perhaps multiple times. Furthermore, everything influences everything else. Ruthlessly avoiding complexity while navigating a huge design space is a supreme engineering challenge. Making the software fast and scalable just makes things more difficult. It is a lot like riding across America.

Commercialization adds its own set of challenges. The software must really work, generating the right answer, never crashing, and dealing successfully with all the corner cases, including running out of any computer resource (such as main memory and disk). Moreover, customers depend on a DBMS to never lose their data, so transaction management must be bulletproof. This

Code section 2.

```
Until (it works) {
        Come up with a new idea;
        Prototype it with the help of superb computer scientists;
        Persevere, fixing whatever problems come up; always remembering that it is never too late to throw
        everything away;
        }
```

Code section 3.

```
Until (first few customers) {
        With shoe leather, find real world users who will say, "If you build this for real, I will buy it";
        }
Recruit seasoned start-up executives;
Recruit crack engineers;
Until (success or run-out-of-money) {
        Persevere, fixing whatever problems come up;
        }
```

is more difficult than it looks, since DBMSs are multi-user software. Repeatable bugs, or "Bohrbugs," are easy to knock out of a system, leaving the killers, nonrepeatable errors, or "Heisenbugs." Trying to find nonrepeatable bugs is an exercise in frustration. To make matters worse, Heisenbugs are usually in the transaction system, causing customers to lose data. This reality has generated a severe pit in my stomach on several occasions. Producing (and testing) system software takes a long time and costs a lot of money. The system programmers who are able to do this have my admiration. In summary, building and commercializing a new DBMS can be characterized by

```
Have a good idea (or two or three);
Make-it-happen—for a decade or so;
```

This brings up the obvious question: "Why would anybody want to do something this difficult?" The answer is the same as with a Ph.D., getting tenure, or riding across America. I am inclined to accept such challenges. I spent a decade struggling to make Postgres real and would do it again in a heartbeat. In fact, I have done it multiple times since Postgres.

The Present Day

I will finish this narrative by skipping to 2016 to talk about how things ultimately turned out. For those of you who were expecting this article to be a commentary on current good (and not-so-good) ideas, you can watch my IEEE International Conference on Data Engineering 2015 talk on this topic at http://kdb.snu.ac.kr/data/stonebraker_talk.mp4 or the video that accompanies this article in the ACM Digital Library.

Moultonborough, NH, present day. Boston Bound arrived in California the same way it left, on the roof of our car. It now sits in our basement in New Hampshire gathering dust. It has not been ridden since that day at Wollaston Beach.

I am still inclined to accept physical challenges. More recently, I decided to climb all 48 mountains in New Hampshire that are over 4,000 feet. In a softer dimension, I am struggling to master the five-string banjo.

Leslie is now Director of Marketing for an angel-investor-backed startup in New York City, whose software incidentally runs on Postgres. She refused to major in computer science.

Illustra was successfully integrated into the Informix code base. This system is still available from IBM, which acquired Informix in 2001. The original Illustra code line still exists somewhere in the IBM archives. The academic Postgres code line got a huge boost in 1995 when "Happy" and "Serious" replaced the QUEL query language with a SQL interface. It was subsequently adopted by a dedicated pick-up team that shepherd its development to this day. This is a shining example of open source development in operation. For a short history of this evolution, see Momjian.[4] This open source code line has also been integrated into several current DBMSs, including Greenplum and Netezza. Most commercial DBMSs have extended their engines with Postgres-style ADTs.

I now want to conclude with three final thoughts. First, I want to mention the other DBMSs I have built—Ingres, C-Store/Vertica, H-Store/VoltDB, and SciDB—all have development stories similar to that of Postgres. I could have picked any one of them to discuss in this article. All had a collection of superstar research programmers, on whose shoulders I have ridden. Over the years, they have turned my ideas into working prototypes. Other programming superstars have converted the prototypes into bulletproof working code for production deployment. Skilled startup executives have guided the small fragile companies with a careful hand. I am especially indebted to my current business partner, "Cueball," for careful stewardship in choppy waters. Moreover, I want to acknowledge the Land Sharks, without whose capital none of this would be possible, especially the "Believer," who has backed multiple of my East Coast companies.

I am especially indebted to my partner, Larry Rowe, and the following 39 Berkeley students and staff who wrote Postgres: Jeff Anton, Paul Aoki, James Bell, Jennifer Caetta, Philip Chang, Jolly Chen, Ron Choi, Matt Dillon, Zelaine Fong, Adam Glass, Jeffrey Goh, Steven Grady, Serge Granik, Marti Hearst, Joey Hellerstein, Michael Hirohama, Chin-heng Hong, Wei Hong, Anant Jhingren, Greg Kemnitz, Marcel Kornacker, Case Larsen, Boris Livshitz, Jeff Meredith, Ginger Ogle, Mike Olson, Nels Olsen, Lay-Peng Ong, Carol Paxson, Avi Pfeffer, Spyros Potamianos, Sunita Surawagi, David Muir Sharnoff, Mark Sullivan, Cimarron Taylor, Marc Teitelbaum, Yongdong Wang, Kristen Wright, and Andrew Yu.

Second, I want to acknowledge my wife, Beth. Not only did she have to spend two months looking at my back as we crossed America, she also gets to deal with my goal orientation, desire to start companies, and, often, ruthless focus on "the next step." I am difficult to live with, and she is long-suffering. I am not sure she realizes she is largely responsible for keeping me from falling off my own personal cliffs.

Third, I want to acknowledge my friend, colleague, and occasional sounding board, Jim Gray, recipient of the ACM A.M. Turing Award in 1998. He was lost at sea nine years ago on January 28, 2007. I think I speak for the entire DBMS community when I say: Jim: We miss you every day.

References

1. Date, C. Referential integrity. In *Proceedings of the Seventh International Conference on Very Large Data Bases Conference* (Cannes, France, Sept. 9–11). Morgan Kaufmann Publishers, 1981, 2–12.
2. Madden, S. *Mike Stonebraker's 70th Birthday Event.* MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA, Apr. 12, 2014; http://webcast.mit.edu/spr2014/csail/12apr14/
3. Mohan, C., Haderle, D., Lindsay, B., Pirahesh, H., and Schwarz, P. Aries: A transaction recovery method supporting fine granularity locking and partial rollbacks using write-ahead logging. *ACM Transactions on Database Systems 17*, 1 (Mar. 1992), 94–162.
4. Momjian, B. *The History of PostgreSQL Open Source Development*; https://momjian.us/main/writings/pgsql/history.pdf
5. Stonebraker, M. The design of the Postgres storage system. In *Proceedings of the 13th International Conference on Very Large Data Bases Conference* (Brighton, England, Sept. 1–4). Morgan Kaufmann Publishers, 1987, 289–300.
6. Stonebraker, M. and Rowe, L. The design of Postgres. In *Proceedings of the 1986 SIGMOD Conference* (Washington, D.C., May 28–30). ACM Press, New York, 1986, 340–355.
7. Stonebraker, M and Rowe, L. The Postgres data model. In *Proceedings of the 13th International Conference on Very Large Data Bases Conference* (Brighton, England, Sept. 1–4). Morgan Kaufmann Publishers, 1987, 83–96.
8. Stonebraker, M., Rubenstein, B., and Guttman, A. Application of abstract data types and abstract indices to CAD databases. In *Proceedings of the ACM-IEEE Workshop on Engineering Design Applications* (San Jose, CA, May). ACM Press, New York, 1983, 107–113.

Michael Stonebraker (stonebraker@csail.mit.edu) is an adjunct professor in the MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA.

DOI:10.1145/2786752

Data from phone interactions can help address customers' complaints, *and* predict their future purchasing behavior.

BY J.P. SHIM, J. KOH, S. FISTER, AND H.Y. SEO

Phonetic Analytics Technology and Big Data: Real-World Cases

SINCE THE MID-2000S, few business topics have received as much attention as big data and business analytics,[5,8,11,13] including unstructured data derived from social media, blogs, chat, and email messages. In addition to unstructured data, YouTube, Vimeo, and other video sources represent another aspect of organizations' customer services. A 2011 IBM survey of more than 4,000 IT professionals from 93 countries and 25 industries[7] identified big data and business analytics as a major business trend for most organizations, along with mobile, cloud, and social business technologies. This trend is also reflected in a number of professional reports and academic journals, including *McKinsey Quarterly* and *MIS Quarterly*. The related skills can also potentially help give organizations a competitive advantage.

Big data takes many forms, including Web and social-media data, machine-to-machine data, transaction data, biometric data, and human-generated data. Human-generated data is our focus here, including vast quantities of unstructured data (such as call-center agents' notes, voice recordings, email messages, paper documents, surveys, and electronic medical records). A number of call analytics technologies are available, including voice searching and indexing for call centers through company-specific phonic-indexing technology. One important application is real-time monitoring that, in a call-center setting, can help address agitated callers and get supervisors involved more quickly. Analytics can process hundreds of hours of audio files in a day, depending on server load, and provide organizations detailed reports on ways to improve customer calls and related job functions, detect problems in operational sectors, and even uncover root problems in products. These systems capture, categorize, store, and analyze unstructured data and can be customized for each customer to include language identification, audio entity extraction, and real-time monitoring.

Here, we review speech and call analytics, especially phonetic search-pattern technology and actual use of voice

» key insights

- **Practitioners and academic researchers are paying increased attention to big data and business analytics in various industries, notably in call centers.**
- **The benefits of speech and call analytics and voice searching/indexing technologies can be fully understood only through real-world data and corporate cases.**
- **Such understanding involves learning the technology, as well as its legal, marketing, customer psychology, and return-on-investment implications.**

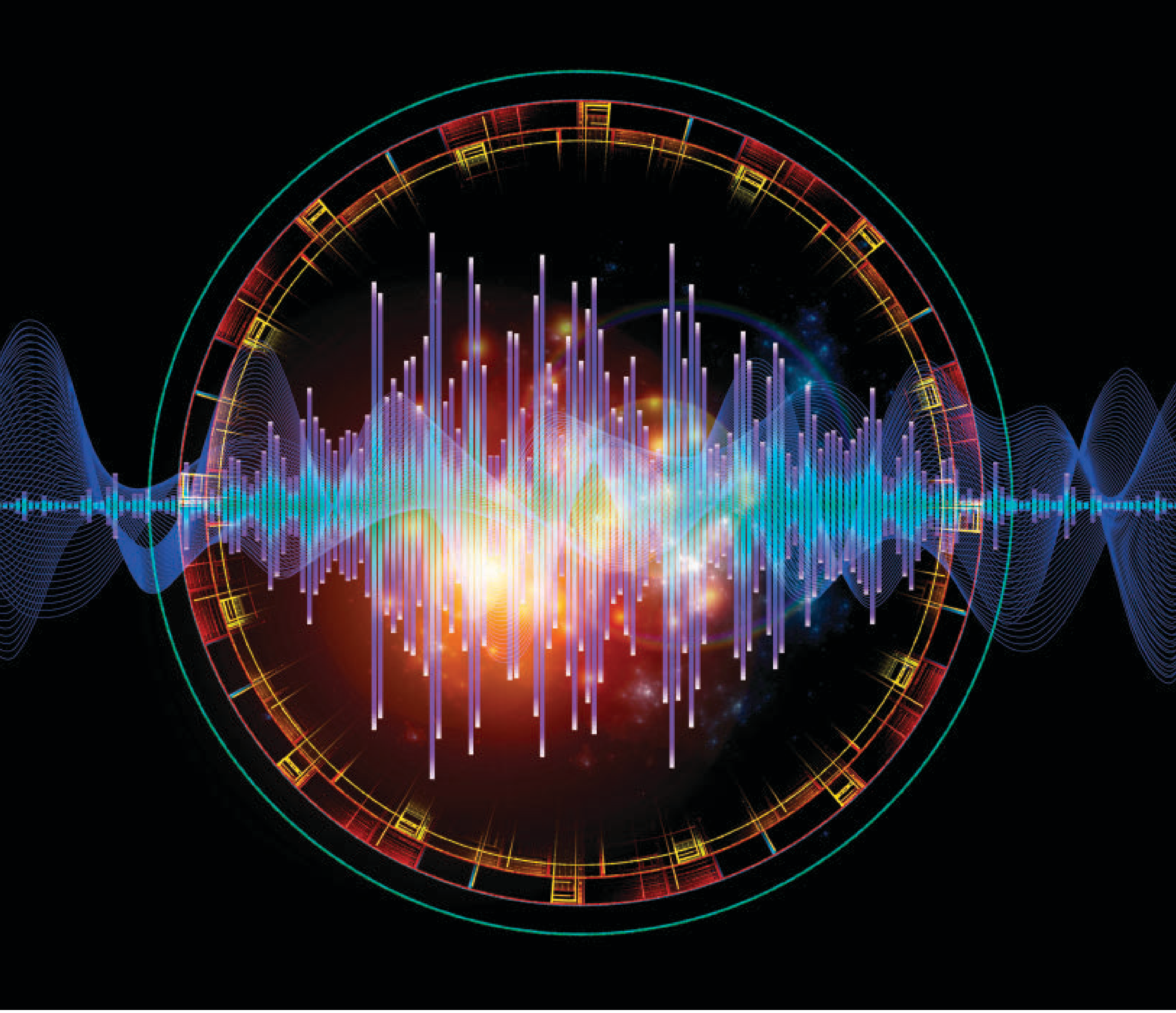


IMAGE BY AGSANDREW

searching and indexing, along with the benefits of phonic search-pattern technology. We cover real-world data users, including call-center solution vendors and their clients. We also offer insight that can help managers increase customer satisfaction and potentially give themselves a competitive edge.

Overview

The first big data center was built in 1965 when the U.S. government had to store 742 million tax returns and 175 million sets of fingerprints. The birth of the Web in 1989 was a milestone in the history of big data housed and accessed on and through the Internet. In 2004, emerging big data tools like Hadoop began to help business managers and researchers understand that data.[17]

A Gartner report in 2013[4] described big data as pertaining to big volume, velocity, and variety of information assets that require new forms of processing to enhance decision making. These elements reflect how quickly the data is processed (volume), how much of it is structured and unstructured (variety), and how data flows in from among various sources (velocity). More recently, big data has begun to also reflect veracity, or the abnormalities in data and its business value.

Big data takes several forms: Web and social media data, including clickstream and interaction data from social media; machine-to-machine data, including from sensors, meters, and other devices; big transaction data, including health care claims, telecommunications call records, and utility billing records; biometric data, including fingerprints, genetics, handwriting, and retinal scans; and human-generated data, including unstructured and semistructured data (such as call-center agents' notes, voice recordings, email messages, paper documents, surveys, and electronic medical records).[4]

Speech and call analytics use phonetic search technology to analyze customer interactions, identify critical areas in need of improvement, and drive business transformation. Phonetic search technology is a method of speech or voice recognition.

Phonetic search-pattern technology. Organizations increasingly use technology based on phonetics, or the systematic study of the sounds of human speech. Within all human languages, there are approximately 400 distinct sounds, or "phonemes," though most use only a fraction of the total. By collecting them, organizations are able to capture a true record of what is said in an audio track that can be searched more accurately and flexibly than human analysts could otherwise do on their own.

The process works in two phases. In one, recorded audio is input into the system, and a time-aligned phonetic index is generated automatically. Because phonemes are simply uttered sounds, indexing them is not affected by background noise, language, dialect, or speaking style. The other begins when a search is requested by a human analyst. Searches can be done directly on words or phrases or through special operators (such as Boolean strings and time-based proximity to other content). A search engine identifies and matches the phonetic equivalent of the search string and returns results ranked by relevance.

The technology can potentially deliver several benefits:

Greater speed. Phonemes are the tiniest building blocks of language. Using them enables quicker audio processing and enhanced ability to find words and phrases in context without complex and difficult-to-maintain dictionaries;

Greater accuracy. Today's spoken human languages are constantly changing. New words, industry terms, blended words, proper names, slang, code words, brand names, and even the nonstandard mixing of different languages are all easily processed through the phonetic approach; and

Greater flexibility. Since the technology is not dictionary-based, the system does not have to be trained on dialects or accents.

Applications: Nexidia

Atlanta, GA-based phonetic-search-and-indexing-technology firm Nexidia, through real-time monitoring analytics and call-operator pop-ups, provides solutions to end users that spare them having to send a service vehicle to a client-user's location, saving the client money and supporting its bottom line. Technology customers support high interaction volumes around product and technical support. Such interaction can make or break a company's innovation life cycle and make the technology's reputation.

As the health-care market shifts to a more consumer-driven, retail-like business model, health-care insurance companies must bolster service to remain competitive by reducing operational costs and increasing enrollment. Solutions must help improve the customer experience, increase first-call resolution, and maximize enrollment.

Failure by insurance companies to comply can cost them in terms of financial results and brand reputation. They must thus take a proactive, continuously vigilant stance. Phonetics-using insurance companies are able to detect potential conflicts during calls and provide quick access to relevant call segments for monitoring high-risk transactions. Compliance violations come in many forms, including non-inclusion of language (such as mini-Miranda rights and other disclosures, abusive language, threat of wage garnishment, and harassment) that generate most consumer complaints, according to FICO, a credit-scoring system, and its Engagement Analyzer software platform.

Call and Phonetic Analytics

Here, we discuss several technologies and cases of phonetic recognition and analytics, self-identification/authentication solutions, and related applications in the context of call centers. Real-world systems have been implemented or are being planned in a number of directions.

Phonetic recognition, analytics, authentication solutions. A representative example of a call-center technology solution is Impact 360 from Verint Systems, which provides analytic software and hardware for the security, surveillance, and business-intelligence markets.[2] It helps generate useful feedback from a market and state-of-the-art consumer trends, helping define and implement marketing strategies and customer segmentation.[3]

Another real-world application is Voice of the Customer Analysis, or VoCA, from Lucis, which has a strategic partnership with Verint Systems. VoCA analyzes an organization's customer-interaction data collected in call centers, assuming such interactions could include information on customer behavior, emotion, and market trends. The technology is intended to help dramatically reduce customer complaints, provide customized service for each customer, and ultimately yield enhanced customer loyalty. Such a solution can help improve a user organization's analytics ability, business insight, company reputation through complaint management, accurate decision making, as reflected in the voices of customers, and marketing based on well-understood market trends.

Another example is Bridgetec's Catch All, Catch You, and Catch Who systems. Catch All helps organizations transform unstructured phonetic data into well-structured text-based data through its speech-to-text engine for the phonetic data of customers inbound to call centers. The technology also includes a data-mining function using text-based keyword extraction that reports statistics to user organizations. Reporting statistics is useful for identifying customer-related issues and determining the overall status of the call center. Catch All involves an ontology-based technology that includes syntax morphemes and keyword indexing. Catch You and Catch Who deal with customer voice commands and authentication. Catch All helps user organizations evaluate employee performance and implement quality assurance based on voice analytics. Through Catch All, an organization can identify which customers responded to its telemarketing from the possibly millions of calls collected in its database.

The Interaction Analytics speech analytics tool from NICE Systems includes phonetic-recognition technology that analyzes why customers contacted the organization about complaints. The technology is useful for real-time speech analytics, phonetic indexing, speech-to-text transcription, speaker separation, emotion detection, and talk-over analysis.[12]

Catch You and Nuance Recognizer are considered by phonetics analysts the two major commercial phonetic-recognition applications, performing voice-command functions that replace the button-type automatic response system (ARS), thus increasing call or contact center efficiency and speed.

The Hyundai credit card company implemented this technology in 2010, replacing a typical ARS it was using in its call centers at the time.

FreeSpeech from Nuance Communications is a representative voice-verification/authentication system.[16] Voice verification and authentication identify customers through their unique voices and tones. The technology matches customers with digital audio files stored in a digital audio database, since each human voice has a unique frequency. Another Nuance solution is VocalPassword, which is able to verify customer identity during interaction with self-service voice applications (such as voice response) and mobile apps. The customer recites a passphrase, and the application verifies the person's identity by comparing their vocal tone against those in its database.[18]

Phonetic recognition, analytics, authentication. Here, we discuss real-world applications of phonetic analytics in call centers. The first is NH Credit Card of South Korea, which has used Emo-Ray phonetic analytics to determine whether calling customers are angry.[6] The system is able to identify which calls should be treated with special care based on analysis of a customer's vocal tone, voice frequency, and other identifying characteristics. It can thus identify calls from angry customers through phonetic analytics. The hit ratio of its performance in distinguishing between angry and normal reportedly reached 90% in January 2014, following a large data security breach in South Korea. Moreover, NH Credit Card of South Korea analyzes the call histories of customers referred to its call center.

The second case involves the phonetic analytics used by AXA Direct Insurance[14] call center agents to request guidance as to how they should respond to customer calls, aiming to optimize response time. AXA Direct was thus reportedly able to reduce inefficient call interactions up to 61% in July 2014, as covered by News1, a South Korean news agency. This cost savings reflected AXA's reduced total call time following improved efficiency of its call center agents' search time. In the long-term, AXA could improve its overall first-call resolution and customer satisfaction.

Because phonemes are simply uttered sounds, indexing them is not affected by background noise, language, dialect, or speaking style.

A third real-world example of phonetic solutions being used in call centers involves voice verification,[10,15] whereby call-center agents are able to identify calling customers through technology from Nuance Communication and Bridgetec. Call centers verify caller identities through their unique vocal tones, saving time otherwise needed to authenticate customers and promising to enhance customer satisfaction. For example, South Korea's Ministry of Public Administration and Security forbids all companies operating in South Korea from collecting and storing Social Security Numbers (SSNs) in their databases due to the risk of leaking personal information. The technology's potential diffusion into corporate call centers would require highly accurate voice verification and monitoring potential legal violations. It would also require sufficient time to acquire and use customers' voice and frequency data. SK Telecom, a South Korea mobile service provider, has implemented such voice verification in its call center.

We may thus categorize phonetic technologies according to their function—recognition, analytics, and authentication—along with their major solutions and benefits for call centers; Table 1 classifies the benefits of phonetic analytics for both solution clients and end users. The main benefits include efficiency, customer satisfaction, and support for overall business strategy.

Implications

Organizations using phonetic-search-pattern technology can reduce call-handling times and cost and improve customer satisfaction. Just as call centers have progressed into contact centers, speech analytics has likewise progressed into interaction analytics, as channels (such as text and social media) have been added. The goal is to make sense of all the unstructured data flowing into call centers and getting it into the hands of the managers who need it.

Phonetic analytics technology helps eliminate barriers to understanding interaction between organizations and their customers in call and contact centers. By defining and tracking metrics as they relate to both organiza-

Table 1. Applications and cases of phonetic analytics in call centers.

Functions	Description	Major Solutions	Keywords
Recognition	▸ Performs voice command/recognition, substituting for button-type automated response systems to boost operating efficiency. ▸ Runs digital devices only through voice command. ▸ Enables human-computer conversations.	▸ Catch You (Bridgetec) ▸ Nuance Recognizer (Nuance)	▸ Voice recognition ▸ Voice command ▸ Speech-recognition automatic response system ▸ Speech-recognition interactive voice response
Analytics	▸ Helps transform unstructured phonetic data into well-structured text-based data through the Speech To Text engine for phonetic data of customers inbound to call centers. ▸ Analyzes and categorizes customer interaction data collected in call centers, since interaction with customers in call centers may contain all necessary customer information. ▸ Provides a cross-channel analytics platform enabling companies to transform valuable but hidden information in customer interaction into business results.	▸ Impact360 (Verint Systems) ▸ Voice of Customer Analytics (Lucis) ▸ Interaction Analytics (NICE Systems) ▸ Catch All (Bridgetec)	▸ Voice analytics ▸ Speech analytics ▸ Interaction analytics ▸ Voice-of-the-customer analysis ▸ Cross-channel analytics ▸ Customer interaction ▸ Voice dictation ▸ Speech to text ▸ Data mining ▸ Big data ▸ Key Management Service
Authentication	▸ Uses the unique voice tones of calling customers; call center agents verify identities. ▸ To authenticate calling customers, the system uses the unique voice tone of calling customers; without passwords, call centers are able to verify customer identities automatically through voice tones in conversations. ▸ Transparently retrieves the biometric voice characteristics required for verification within seconds regardless of what is said, accent, or language. ▸ Verifies customer identities during interaction with voice application (such as interactive voice response and mobile apps); verifies customer identity by comparing calling customer voice tones with those recorded in a database.	▸ Catch Who (Bridgetec) ▸ FreeSpeech (Nuance) ▸ Vocal Password (Nuance)	▸ Voice authentication ▸ Voice verification ▸ Voice recognition ▸ Social verification ▸ Voice biometrics ▸ Authentication through passphrases ▸ Security ▸ VoiceXML

tional and agent performance, behavioral issues between call-center agents and customers are identified, as are the business processes and procedures in the way of achieving strategic business goals. The technology does more than provide surface-level information and statistics on call-center/customer interaction. Agent behavior affecting the business can be identified, then help determine how to respond. Business managers should thus consider the following aspects of call and phonetic analytics technology.

Technology. Big data in call analytics and recognition must be secure. Some call data in the banking industry cannot, by law, be used in other industries. Constantly changing words, abbreviations, and technical terms must be updated for call analytics, and voice quality needs to be studied. The major technical challenge involves the number of non-English languages in the world. The technology of phonetic analytics should be able to understand and recognize even the relatively obscure languages people use. Well-structured data, rather than unstructured data, can fit the technical dimension of phonetic technology. However, technological advances in unstructured, phonetic recognition and analytics strive to increase business speed/efficiency, leading to a potential revolution in business intelligence.

Legal. Call-center managers must understand privacy and security[18] to be able to build secure data centers reflecting payment-card industry standards, ensuring individual customer privacy. For example, in the case of the South Korea privacy-protection law (signed August 7, 2014) forbidding public institutions from collecting and storing South Korean citizens' SSNs, phonetic-analytics technology may be used to verify customer identity in lieu of SSNs.

Market. Phonetic analytics is expensive. The many analytics solutions, need to secure data volume, and significant speech-analytics-related costs are often barriers to adoption. The major vendors must thus provide clear benefits to potential clients willing to pay when they perceive more benefit than cost in such technology. However, a new market combining phonetic analytics and the Internet of Things has emerged; for example, driving- and cooking-related devices can be connected through phonetic search

Benefits		
Solution Users (Call Centers)	**End Users (Callers)**	**Cases**
► Improved customer satisfaction ► Improved customer accessibility ► Improved operating efficiency through increased self service ► Reduced communication fees	► User convenience ► Time savings ► Reduced customer effort	► Phonetic recognition replaced automatic response system when customers dial the Hyundai Credit Card call center.
Efficiency/Speed ► Improved efficiency and cost savings ► Reduced talk time and improved first-call resolution ► Reduced agent overload and supervision **Customer Satisfaction** ► Managed voice of the customer ► Maximum effectiveness and customer satisfaction ► Improved customer experience ► Reduced customer churn **Strategic Use** ► Identify customer-related issues and obtain internal customer-satisfaction data ► Help gain useful feedback from current market and trends ► Implement marketing-strategy planning	► Proactive search for customer needs ► Increased first-call resolution ► Customer-oriented services	► The Emo-Ray in Nonghyup Credit Card indicates which calls should be treated with extra care based on analytics of customer speech tones and voice frequency. ► AXA Direct provides call center agents with hints about how they should respond to customers calls.
► Lower communication fees and better cost management ► Improved call center efficiency ► Reduced call duration and average talk time ► Reduced call center fraud ► Improved customer experience ► Improved operating efficiency through increased self-service	► Time savings ► Reduced cost of calls	► SK Telecom in Korea has adopted authentication technology in its call centers. ► Using the authentication system, many global banks are able to identify who is calling in seconds.

technology without users having to touch them.

Customer psychology. In addition to concern for the privacy of their personal data, customers are reluctant to use phonetic technology when interacting with corporate call centers, reflecting a type of common business logic. Customers' trust and consent is required for an organization to adopt phonetic analytics technology but, K.L. Keller[9] wrote, it is very difficult for customers to change their purchasing habits for the sake of new products and services.

Return on investment. Speech analytics technology is expensive. Numerous business leaders believe in the potential of big data and analytics but find big data return on investment difficult to measure.[1] The cost of maintaining huge databases and analytics is also significant but may soon decrease as more and more companies capture, process, analyze, and store vast amounts of data.

Beyond the five issues we have outlined here, the most significant barrier preventing adoption of phonetic technology is customer psychology and habits. Most customers are unlikely to change their purchasing habits if they are not familiar with new voice-recognition-and-authentication technologies. Difficulties in phonetic data collection will thus continue to hamper development of phonetic analytics. Moreover, customers in the banking and stock-trading industries are even more reluctant to adopt the new technology, as data error or system failure can potentially produce huge losses.

These problems call for initiatives from solution vendors and solution buyers, or call-center organizations, and for collaboration between them. First, solution vendors must update their technologies to minimize any possible errors; for example, a stock-trading company requires a 100% reliable phonetic system that permits no technical error. Second, solution clients, or call centers, must encourage customers to adopt the technology by providing free calls and discounted service charges. Customers' trust and consent is required before phonetic recognition, analytics, and authentication is implemented. Finally, solution vendors and solution clients must collaboratively clarify and share their aims in implementing phonetic technology based on their partnership. Their objectives may be cost reduction (through efficiency), performance en-

Table 2. Key issues in phonetic analytics technology for practitioners.

Aspect of Business Model	Issues and Implications for Practitioners Solution Vendors	 Solution Clients	 Policymakers
Technology	► Secure data ► Technological advancement in ill-structured data processing	► Secure data ► Advances in phonetic analytics	NA
Legal	► Use of personal information	► Management of personal information ► Customer permission	► Privacy-protection law
Market	► New market combined with Internet of Things ► Market extension ► Reasonable price for solution	► Adoption cost	► Market regulation
Customer Psychology	NA	► Customer habits ► Ease of use customers perceive	► Privacy agreement
ROI	► Costly solutions	► Unconvinced ROI	NA

hancement, and/or customer satisfaction. The objectives may vary depending on organizational circumstances. Moreover, governments must communicate with industry stakeholders by reviewing whether phonetic technology could violate privacy-protection laws. However, phonetic-analytics technology is still in an early stage of development, with questions concerning government policy, the technology itself, the phonetics market, and customer purchasing habits. Finally, the three concepts discussed here—phonetic recognition, analytics, and authentication—must be clarified, as they overlap and are sometimes used interchangeably; Table 2 summarizes the implications of phonetic analytics technology for solution vendors, clients, and policymakers.

Conclusion

Business intelligence and analytics may provide an opportunity for organizations to learn more about their own customers' purchasing power, product placement, feedback, long-tail marketing, targeted and personalized recommendations, and increased sales through enhanced customer satisfaction.[5] With access to more and more data, organizations are able to solve their customers' problems more quickly and efficiently and improve job functions and authority issues. Data can be assimilated quickly and customized to an organization's individual circumstances, identifying problem areas and providing recommendations and coaching tools for call-center agents. Analysts can help define queries and search information for benchmarking and root-cause analysis and recommendations for problem solving. The goal is to provide organized and easily accessible information, quicker problem solving, increased service value, and ultimately more business.

The future of phonetic analysis involves lots of electronic data. Since social media, blog posts, chats, and email messages are already in text form, organizations are potentially better able to produce a more complete picture of their business environment. Along with unstructured data, YouTube and Vimeo videos are yet another type of customer service platform. That is, both structured and unstructured data can be aggregated into analyses that then help paint a bigger picture. One promising idea for emergency calling, or 911, applications is for the software to track phone calls and paint a picture for first responder(s) sent to the scene of a crime or other location. While business analytics receives considerable attention, virtually all the studies we are aware of have neglected or given only cursory attention to call analytics, or phonetic search and indexing technology. We hope our own research, as outlined here, aids decision makers and managers dealing with unstructured tasks to identify patterns and trends in consumer behavior.

References

1. Bertolucci, J. Big data ROI still tough to measure. *InformationWeek* (May. 29, 2013).
2. Bodner, D. *Verint Impact 360 Speech Analytics Helps Shanghai Unicom Listen and Take Action Based on the Voice of Its Customers.* Verint Systems, Inc., Melville, NY, Apr. 22, 2014; http://bit.ly/1OM0LTD
3. Bodner, D. *Speech Analytics: Use the Voice of Your Customer to Optimize Your Business.* Verint Systems, Inc., Melville, NY, 2014; http://bit.ly/1Oj8vMe
4. Buytendijk, F. and Laney, D. *Drive Value from Big Data Through Six Emerging Best Practices.* Gartner Inc., Stamford, CT, Oct. 29, 2013; http://gtnr.it/1Plvagi
5. Chen, H., Chiang, R.H.L., and Storey, V. Business intelligence and analytics: From big data to big impact. *MIS Quarterly 36*, 4 (Dec. 2012), 1165–1188.
6. Choi, K.M. ChosunBiz. Call centers, they know more about me than I do; http://bit.ly/1U5v1N5
7. Gokhale, V. *The 2011 IBM Tech Trends Report: The Clouds Are Rolling In ... Is Your Business Ready?* IBM, New York, Nov. 23, 2011; http://ibm.co/1Plc0VR
8. Jagadish, H.V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., and Shahabi, C. Big data and its technical challenges. *Commun. ACM 57*, 7 (July 2014), 86–94.
9. Keller, K.L. *Strategic Brand Management: Building, Measuring, and Managing Brand Equity. Fourth Edition.* Prentice Hall, Upper Saddle River, NJ, 2012.
10. Kim, S.J. Bridgetec Catch All: There is a solution in communications with customers. *Newsprime* (May. 26, 2014); http://bit.ly/1YyzO0b
11. Kohavi, R., Rothleder, N.J., and Simoudis, E. Emerging trends in business analytics. *Commun. ACM 45*, 8 (Aug. 2002), 45–48.
12. Kostman, D. *Speech Analytics: Innovative Speech Technologies to Unveil Hidden Insights.* NICE Systems, Inc., Ra'anana, Israel, 2014; http://bit.ly/1Mx71wE
13. Lavalle, S., Shockley, R., Hopkins, M.S., and Kruschwltz, N. Big data, analytics, and the path from insights to value. *Sloan Management Review 52*, 2 (Winter 2011), 21–31.
14. Lee, H.C. AXA Direct could reduce inefficient calls up to 61% by using call content analysis in call centers. *News1* (July 2, 2014); http://news1.kr/articles/?1751498
15. Park, S.Y. Caller authentication solutions: Callers' identity can be identified by their voice. *Digital Times* (Mar. 9, 2014); http://bit.ly/1RHvtU8
16. Ricci, P. *Authentication Via Conversation.* Nuance, Inc., Burlington, MA, 2015;http://bit.ly/1ScHoIN
17. Stoecker, D. Hadoop analytics. Alteryx, Inc., Irvine, CA, 2015; http://www.alteryx.com/solutions/hadoop-analytics
18. Vaughan-Nichols, S. Voice authentication speaks to the marketplace. *Computer 37*, 3 (Mar. 2004), 13–15.

J.P. Shim (jpshim@gsu.edu) is executive director of the Korean-American Business Center at Robinson College of Business and a professor of computer information systems at Georgia State University, Atlanta, GA, and professor emeritus at Mississippi State University, Starkville, MS.

J. Koh (kjoon@chonam.ac.kr) is an associate professor of management information sysetms in the College of Business Administration at Chonnam National University, Kwang-Ju, South Korea.

S. Fister (steven.metocean@gmail.com) is a solutions architect at MetOcean Solutions Ltd., New Plymouth, New Zealand.

H.Y. Seo (gydud80@naver.com) is a manager at The Asian Culture Institute, Kwang-Ju, South Korea.

18th International Conference on Human-Computer Interaction with Mobile Devices and Services

MobileHCI 2016

September 6-9, Florence, Italy

DEADLINES

FEBRUARY 12, 2016
for Full and Short Papers

FEBRUARY 26, 2016
for Workshop proposals

MAY 13, 2016
for Posters, Demos, and Tutorials

GENERAL CHAIRS

Fabio Paternò (CNR-ISTI, Italy)
Kaisa Väänänen (TUT, Finland)

PAPER CHAIRS

Karen Church (Yahoo, USA)
Jonna Hakkila (U. Lapland, Finland)
Antonio Kruger (DFKI, Germany)
Marcos Serrano (U. Toulouse, France)

MobileHCI is sponsored by

Association for Computing Machinery

SIGCHI

MobileHCI seeks contributions in the form of innovations, insights, or analyses related to human interaction and experiences with mobile devices and services. We seek richness and diversity in topic as well as approach, method, and viewpoint. In no particular order, this includes contributions in:

Systems & Infrastructures. The design, architecture, deployment, and evaluation of systems and infrastructures that support development of or interaction with mobile devices and services.

Devices & interaction techniques. The design, construction, usage, and evaluation of devices and techniques that create valuable new capabilities for mobile human-computer interaction.

Applications & experiences. Descriptions of the design, empirical study of interactive applications, or analysis of usage trends that leverage mobile devices and systems.

Methodologies & tools. New methods and tools designed for or applied to studying or building mobile user interfaces, applications, and mobile users.

Theories & models. Critical analysis or organizing theory with clearly motivated relevance to the design or study of mobile human-computer interaction; taxonomies of design or devices.

Visions. Well-argued and well-supported visions of the future of mobile computing or non-traditional topics that bear on mobility.

We solicit original research and technical papers not published elsewhere focusing on the following topics (but not limited to):

- Novel user interfaces and interaction techniques
- Mobile social networks and systems
- Design of location-based or context-aware services for mobile devices
- Context-aware systems
- Multimodal interaction (including audio and speech)
- User-centred design tools and methods for mobile systems
- Ethnographical and field studies with mobile technology
- Group interaction and mobility communities
- Services for mobile devices
- Design, evaluation and case studies of mobile applications
- Wearable computing, smart clothes, new devices and sensors
- Design and tools for cross-device user interfaces
- Mobile entertainment, storytelling and location based gaming
- Aesthetic interaction and experience design
- Affective computing in the mobile context
- Personal assistance with mobile devices
- Mobility and work environments
- Evaluation and usability of mobile devices and services
- Mobile accessibility

mobilehci.acm.org/2016

DOI:10.1145/2845915

DANIEL ABADI

RAKESH AGRAWAL

ANASTASIA AILAMAKI

MAGDALENA BALAZINSKA

PHILIP A. BERNSTEIN

MICHAEL J. CAREY

SURAJIT CHAUDHURI

JEFFREY DEAN

ANHAI DOAN

MICHAEL J. FRANKLIN

JOHANNES GEHRKE

LAURA M. HAAS

ALON Y. HALEVY

JOSEPH M. HELLERSTEIN

YANNIS E. IOANNIDIS

H.V. JAGADISH

DONALD KOSSMANN

SAMUEL MADDEN

SHARAD MEHROTRA

TOVA MILO

JEFFREY F. NAUGHTON

RAGHU RAMAKRISHNAN

VOLKER MARKL

CHRISTOPHER OLSTON

BENG CHIN OOI

CHRISTOPHER RÉ

DAN SUCIU

MICHAEL STONEBRAKER

TODD WALTER

JENNIFER WIDOM

Database researchers paint big data as a defining challenge. To make the most of the enormous opportunities at hand will require focusing on five research areas.

The Beckman Report on Database Research

A GROUP OF database researchers meets periodically to discuss the state of the field and its key directions going forward. Past meetings were held in 1989,[6] 1990,[11] 1995,[12] 1996,[10] 1998,[7] 2003,[1] and 2008.[2] Continuing this tradition, 28 database researchers and two invited speakers met in October 2013 at the Beckman Center on the University of California-Irvine campus for two days of discussions. The meeting attendees represented a broad cross-section of interests, affiliations, seniority, and geography. Attendance was capped at 30 so the meeting would be as interactive as possible. This article summarizes the conclusions from that meeting; an extended report and participant presentations are available at http://beckman.cs.wisc.edu.

IMAGE BY MAKSIM KABAKOU

The meeting participants quickly converged on big data as a defining challenge of our time. Big data arose due to the confluence of three major trends. First, it has become much cheaper to generate a wide variety of data, due to inexpensive storage, sensors, smart devices, social software, multiplayer games, and the Internet of Things, which connects homes, cars, appliances, and other devices. Second, it has become much cheaper to process large amounts of data, due to advances in multicore CPUs, solid state storage, inexpensive cloud computing, and open source software. Finally, data management has become democratized. The process of generating, processing, and consuming data is no longer just for database professionals. Decision makers, domain scientists, application users, journalists, crowd workers, and everyday consumers now routinely do it.

Due to these trends, an unprecedented volume of data needs to be captured, stored, queried, processed, and turned into knowledge. These goals are remarkably well aligned with those that have driven the database research community for decades. Many early systems for big data abandoned database management system (DBMS) principles, such as declarative programming and transactional data consistency, in favor of scalability and

» key insights

- Thirty leaders from the database research community met in October 2013 to discuss the state of the field and important future research directions.
- Big data was identified as a defining challenge for the field. Five related challenges were called out: developing scalable data infrastructures, coping with increased diversity in both data and data management, addressing the end-to-end data-to-knowledge pipeline, responding to the adoption of cloud-based computing, and accomodating the many and changing roles of individuals in the data life cycle.
- College-level database education needs modernization to catch up with the many changes in database technology of the past decade and to meet the demands of the emerging disciplines of data science.

fault tolerance on commodity hardware. However, the latest generation of big data systems is rediscovering the value of these principles and is adopting concepts and methods that have been long-standing assets of the database community. Building on these principles and assets, the database community is well positioned to drive transformative improvements to big data technology.

But big data also brings enormous challenges, whose solutions will require massive disruptions to the design, implementation, and deployment of data management solutions. The main characteristics of big data are volume, velocity, and variety. The database community has worked on volume and velocity for decades, and has developed solutions that are mission critical to virtually every commercial enterprise on the planet. The unprecedented scale of big data, however, will require a radical rethinking of existing solutions.

Variety arises from several sources. First, there is the problem of integrating and analyzing data that comes from diverse sources, with varying formats and quality. This is another long-standing topic of database work, yet it is still an extremely labor-intensive journey from raw data to actionable knowledge. This problem is exacerbated by big data, causing a major bottleneck in the data processing pipeline. Second, there is the variety of computing platforms needed to process big data: hardware infrastructures; processing frameworks, languages, and systems; and programming abstractions. Finally, there is a range of user sophistication and preferences. Designing data management solutions that can cope with such extreme variety is a difficult challenge.

Moving beyond the three Vs, many big data applications will be deployed in the cloud, both public and private, on a massive scale. This requires new techniques to offer predictable performance and flexible interoperation. Many applications will also require people to solve semantic problems that still bedevil current automatic solutions. This can range from a single domain expert to a crowd of workers, a user community, or the entire connected world (for example, Wikipedia). This will require new techniques to help people be more productive and to reduce the skill level needed to solve these problems.

Many big data applications will be deployed in the cloud, both public and private, on a massive scale. This requires new techniques to offer predictable performance and flexible interoperation.

Finally, big data brings important community challenges. We must rethink the approach to teaching data management, reexamine our research culture, and adapt to the emergence of data science as a discipline.

Research Challenges

The meeting identified five big data challenges: scalable big/fast data infrastructures; coping with diversity in data management; end-to-end processing of data; cloud services; and the roles of people in the data life cycle. The first three challenges deal with the volume, velocity, and variety aspects of big data. The last two deal with deploying big data applications in the cloud and managing the involvement of people in these applications.

These big data challenges are not an exclusive agenda to be pursued at the expense of existing work. In recent years the database community has strengthened core competencies in relational DBMSs and branched out into many new directions. Some important issues raised repeatedly during the meeting are security, privacy, data pricing, data attribution, social and mobile data, spatiotemporal data, personalization and contextualization, energy-constrained processing, and scientific data management. Many of these issues cut across the identified big data challenges and are captured in the discussion here.

It is important to note that some of this work is being done in collaboration with other computer science fields, including distributed systems, artificial intelligence, knowledge discovery and data mining, human-computer interaction, and e-science. In many cases, these fields provided the inspiration for the topic and the data management community has joined in, applying its expertise to produce robust solutions. These collaborations have been very productive and should continue to grow.

Scalable big/fast data infrastructures. *Parallel and distributed processing.* In the database world, parallel processing of large structured datasets has been a major success, leading to several generations of SQL-based products that are widely used by enterprises. Another success is data warehousing, where database researchers defined the key abstraction of data cube (for online analytic processing, or OLAP) and strategies for querying it in parallel, along with support for materialized views and replication. The distributed computing field has achieved success in scaling up data processing for less structured data on large numbers of unreliable, commodity machines using constrained programming models such as MapReduce. Higher-level languages have been layered on top, to enable a broader audience of developers to use scalable big data platforms. Today, open source platforms such as Hadoop[3]—with its MapReduce programming model, large-scale distributed file system, and higher-level languages, such as Pig[5] and Hive[4]—are seeing rapid adoption for processing less structured data, even in traditional enterprises.

Query processing and optimization. Given the enthusiastic adoption of declarative languages for processing big data, there is a growing recognition that more powerful cost-aware query optimizers and set-oriented query execution engines are needed, to fully exploit large clusters of many-core processors, scaling both "up" and "out." This will create challenges for progress monitoring, so a user can diagnose and manage queries that are running too slowly or consuming excessive resources. To adapt to the characteristics of previously unseen data and reduce the cost of data movement between stages of data analysis, query processors will need to integrate data sampling, data mining, and machine learning into their flows.

New hardware. At datacenter scale, the ratio between the speed of sequential processing and interconnects is changing with the advent of faster networks, full bisection bandwidth networks between servers, and remote direct memory access. In addition to clusters of general-purpose multicore processors, more specialized processors should be considered. Commercially successful database machines have shown the potential of hardware-software co-design for data management. Researchers should continue to explore ways of leveraging specialized processors, for example, graphics processing units, field-programmable gate arrays, and application-specific integrated circuits, for processing very large datasets. These changes in communications and processing technologies will require a reconsideration of parallel and distributed query-processing algorithms, which have traditionally focused on more homogeneous hardware environments.

Cost-efficient storage. The database research community must learn how best to leverage emerging memory and storage technologies. Relative to commodity magnetic disks, solid-state disks are expensive per gigabyte but cheap per I/O operation. Various non-volatile random-access memory technologies are under development, all with different speed, power, and durability characteristics.

Both server-attached and network-attached storage architectures need to be considered. Distributed file systems like HDFS, which are server-attached yet shared across the network, are a hybrid of both approaches. How best to use this range of storage configurations reopens many questions reminiscent of past debates of shared memory vs. shared disk vs. shared nothing, questions many have considered to be "closed" for parallel relational systems.

High-speed data streams. For data that arrives at ever-higher speeds, new scalable techniques for ingesting and processing streams of data will be needed. Algorithms will need to be tuned carefully to the behavior of hardware, for example, to cope with non-uniform memory access and limited transfer rates across layers of the memory hierarchy. Some very high-speed data sources, often with lower information density, will need to be processed online and then discarded without being persisted in its entirety. Rather, samples and aggregations of such data will need to be selected and stored persistently to answer queries that arrive after the raw data is no longer available. For such data, progressive query processing will be important to provide incremental and partial results with increasing accuracy as data flows through the processing pipeline.

Late-bound schemas. For data that is persisted but processed just once (if ever), it makes little sense to pay the substantial price of storing and indexing it first in a database system. Instead, it should be stored as a binary file and interpreted as a structured record only if and when it is read later. Record structure may be self-describing via attribute-value pairs, such as JavaScript Object Notation (JSON), interpreted via predefined schemas, or deduced using data mining. To offer the benefits of database queries in such scenarios, we need query engines that can run efficiently over raw files with late-bound schemas.

Consistency. Today's world brings new requirements for data capture, updates, and simple and fast data access. Handling high rates of data capture and updates for schema-less data has led to the development of NoSQL systems. There are many such systems, with a range of transaction models. Most provide only basic data access and weak atomicity and isolation guarantees, making it difficult to build and reason about reliable applications. As a result, a new class of big data system has emerged that provides full-fledged database-like features over key-value stores or similar substrates. For some applications, the stored data is still managed and updated as "the source of truth" for an enterprise. For others, such as the Internet of Things, the stored data reflects ongoing events in the outside world that applications can use to recognize and respond to situations of interest. This creates an opportunity to revisit programming models and mechanisms for data currency and consistency and to design new models and techniques for developing robust applications.

Metrics and benchmarks. Finally, scalability should be measured not only in petabytes of data and queries per second, but also total cost of ownership (including management and energy use), end-to-end processing speed (that is, time from raw data arrival to eventual insights), brittleness (for example, the ability to continue despite failures such as partial data parse errors), and usability (especially for entry-level users). To measure progress against such broader metrics, new types of benchmarks will be required.

Diversity in data management. *No one-size-fits-all.* Today's data-driven world involves a richer variety of data types, shapes, and sizes than traditional enterprise data, which is stored in a data warehouse optimized for analysis tasks. Today, data is often stored in different representations managed by different software systems with different application programming interfaces, query processors, and analysis tools. It seems unlikely a single, one-size-fits-all, big data system will suffice for this degree of diversity. Instead, we expect multiple classes of systems to emerge, each addressing a particular need (for example, data deduplication, analysis of large graphs, diverse scientific experiments, and real-time stream processing) or exploiting a particular type of hardware platform (for example, clusters of inexpensive machines or large multicore servers). Addressing these scenarios will require applying expertise in set-oriented parallel processing and in efficiently handling datasets that do not fit in main memory.

Cross-platform integration. Given this diversity of systems, platforms will need to be integrated or federated to enable data analysts to combine and analyze data across systems. This will involve not only hiding the heterogeneity of data formats and access languages, but also optimizing the performance of accesses that span diverse big data systems and of flows that move data between them. It will also require managing systems that run on diverse devices and span large datacenters. Disconnected devices will become increasingly common, raising challenges in reliable data ingestion, query processing, and data inconsistency in such sometimes-connected, wide-area environments.

Programming models. A diverse and data-driven world requires diverse programming abstractions to operate on very large datasets. A single data analysis language for big data, such as an extension of SQL, will not meet everyone's needs. Rather, users must be able to analyze their data in the idiom they find most natural: SQL, Pig, R, Python, a domain-specific language, or a lower-level constrained programming model such as MapReduce or Valiant's bulk synchronous processing model. This also suggests the development of reusable middle-layer components that can support multiple language-specific bindings, such as scalable support for matrix multiplication, list comprehension, and stylized iterative execution models. Another potentially fruitful focus is tools for the rapid development of new domain-specific data analysis languages—tools that simplify the implementation of new scalable, data-parallel languages.

Data processing workflows. To handle data diversity, we need platforms that can span both "raw" and "cooked" data. The cooked data can take many forms, for example, tables, matrices, or graphs. Systems will run end-to-end workflows that mix multiple types of data processing, for example, querying data with SQL and then analyzing it with R. To unify diverse systems, lazy computation is sometimes beneficial—lazy data parsing, lazy conversion and loading, lazy indexing and view construction, and just-in-time query planning. Big data systems should become more interoperable like "Lego bricks." Cluster resource managers, such as Hadoop 2.0's YARN, provide some inspiration at the systems level, as do workflow systems for the Hadoop ecosystem and tools for managing scientific workflows.

End-to-end processing of data. The database research community should pay more attention to end-to-end processing of data. Despite years of R&D, surprisingly few tools can go from raw data all the way to extracted knowledge without significant human intervention at each step. For most steps, the intervening people need to be highly computer savvy.

Data-to-knowledge pipeline. The steps of the raw-data-to-knowledge pipeline will be largely unchanged: data acquisition; selection, assessment, cleaning, and transformation (also called "data wrangling"); extraction and integration; mining, OLAP, and analytics; and result summarization, provenance, and explanation. In addition to greater scale, what has significantly changed is the greater diversity of data and users. Data today comes in a wide variety of formats. Often, structured and unstructured data must be used together in a structured fashion. Data tools must exploit human feedback in every step of the analytical pipeline, and must be usable by subject-matter experts, not just by IT professionals. For example, a journalist may want to clean, map, and publish data from a spreadsheet file of crime statistics. Tools must also be tailored to data scientists, the new class of data analysis professionals that has emerged.

Tool diversity. Since no one-size-fits-all tool will cover the wide variety of data analysis scenarios ahead, we need multiple tools, each solving a step of the raw-data-to-knowledge pipeline. They must be seamlessly integrated and easy to use for both lay and expert users, with best-practice guidance on when to use each tool.

Tool customizability. Tools should be able to exploit domain knowledge, such as dictionaries, knowledge bases, and rules. They should be easy to customize to a new domain, possibly using machine learning to automate the customization process. Handcrafted rules will remain important, though, as many analysis applications require very high precision, such as e-commerce. For such applications, analysts often write many rules to cover "corner cases" that are not amenable to learning and generalization. Thus, tools should provide support for writing, evaluating, applying, and managing handcrafted rules.

Open source. Few tools in this area are open source. Most are expensive proprietary products that address certain processing steps. As a result, existing tools cannot easily benefit from ongoing contributions by the data integration research community.

Understanding data. Explanation, provenance, filtering, summarization, and visualization requirements will be critical to making analytic tools easy to use. Capturing and managing appropriate meta-information is key to enable explanation, provenance, reuse, and visualization. Visual analytics is receiving growing attention in the database, visualization, and HCI communities. Continued progress in this area is essential to help users cope with big data volumes.

Knowledge bases. The more knowledge we have about a target domain, the better that tools can analyze the domain. As a result, there has been a growing trend to create, share, and use domain knowledge to better understand data. Such knowledge is often captured in knowledge bases (KBs) that describe the most important entities and relationships in a domain, such as a KB containing profiles of tens of thousands of biomedical researchers along with their publications, affiliations, and patents. Such KBs are used for improving the accuracy of the raw-data-to-knowledge pipeline, answering queries about the domain, and finding domain experts. Many companies have also built KBs for answering user queries, annotating text, supporting e-commerce, and analyzing social media. The KB trend will likely accelerate, leading to a proliferation of community-maintained "knowledge centers" that offer tools to query, share, and use KBs for data analysis.

While some progress has been made on this topic, more work is needed on tools to help groups of users with different skill levels collaboratively build, maintain, query, and share domain-specific KBs.

Cloud services. Cloud computing comes in three main forms: Infrastructure as a Service (IaaS), where the service is virtualized hardware; Platform as a Service (PaaS), where the service is virtualized infrastructure software such as a DBMS; and Software as a Service (SaaS), where the service is a virtualized application such as a customer relationship management solution. From a data platform perspective, the ideal goal is a PaaS for data, where users can upload data to the cloud, query it as they do today over their on-premise SQL databases, and selectively share the data and results easily, all without worrying about how many instances to rent, what operating system to run on, how to partition databases across servers, or how to tune them. Despite the emergence of services such as Database.com from Salesforce.com, Big Query from Google, Redshift from Amazon, and Azure SQL Database from Microsoft, we have yet to achieve the full ideal. Here, we outline some of the critical challenges to realize the complete vision of a Data PaaS in the cloud.

A diverse and data-driven world requires diverse programming abstractions to operate on very large datasets.

Elasticity. Data can be prohibitively expensive to move. Network-attached storage makes it easier to scale out a database engine. However, network latency and bandwidth limit database performance. Server-attached storage reduces these limitations, but then server failures can degrade availability and failover can interfere with load balancing and hence violate service-level agreements (SLAs).

An open question is whether the same cloud storage service can support both transactions and analytics; how caching best fits into the overall picture is also unclear. To provide elasticity, database engines and analysis platforms in a Data PaaS will need to operate well on top of resources that can be allocated quickly during workload peaks but possibly preempted for users paying for premium service.

Data replication. Latency across geographically distributed datacenters makes it difficult to keep replicas consistent yet offer good throughput and response time to updates. Multi-master replication is a good alternative, when conflicting updates on different replicas can be automatically synchronized. But the resulting programming model is not intuitive to mainstream programmers. Thus, the challenge is how best to trade-off availability, consistency performance, programmability, and cost.

System administration and tuning. In the world of Data PaaS, database and system administrators simply do not exist. Therefore, all administrative tasks must be automated, such as capacity planning, resource provisioning, and physical data management. Resource control parameters must also be set automatically and be highly responsive to changes in load, such as buffer pool size and admission control limits.

Multitenancy. To be competitive, a Data PaaS should be cheaper than an on-premises solution. This requires providers to pack multiple tenants together to share physical resources to smooth demand and reduce cost. This introduces several problems. First, the service must give security guarantees against information leakage across tenants. This can be done by isolating user databases in separate files and running the database engine in separate virtual machines (VMs). However, this is inefficient for small databases, and makes it difficult to balance resources between VMs running on the same server. An alternative is to have users share a single database and database engine instance. But then special care is needed to prevent cross-tenant accesses. Second, users want an SLA that defines the level of performance and availability they need. Data PaaS providers want to offer SLAs too, to enable tiered pricing. However, it is challenging to define SLAs that are understandable to users and implementable by PaaS providers. The implementation challenge is to ensure performance isolation between tenants, to ensure a burst of demand from one tenant does not cause a violation of other tenants' SLAs.

Data sharing. The cloud enables sharing at an unprecedented scale. One problem is how to support essential services such as data curation and provenance collaboratively in the cloud. Other problems include: how to find useful public data, how to relate self-managed private data with public data to add context, how to find high-quality data in the cloud, how to share data at fine-grained levels, how to distribute costs when sharing computing and data, and how to price data. The cloud also creates new life-cycle challenges, such as how to protect data if the current cloud provider fails and to preserve data for the long term when users who need it have no personal or financial connection to those who provide it. The cloud will also drive innovation in tools for data governance, such as auditing, enforcement of legal terms and conditions, and explanation of user policies.

Hybrid clouds. There is a need for interoperation of database services among the cloud, on-premise servers, and mobile devices. One scenario is off-loading. For example, users may run applications in their private cloud during normal operation, but tap into a public cloud at peak times or in response to unanticipated workload surges. Another is cyber-physical systems, such as the Internet of Things. For example, cars will gather local sensor data, upload some of it into the cloud, and obtain control information in return based on data aggregation from many sources. Cyber-physical systems involve data streaming from multiple sensors and mobile devices, and must cope with intermittent connectivity and limited battery life, which pose difficult challenges for real-time and perhaps mission-critical data management in the cloud.

We need to build platforms that allow people to curate data easily and extend relevant applications to incorporate such curation.

Roles of humans in the data life cycle. Back when data management was an enterprise-driven activity, roles were clear: developers built databases and database-centric applications, business analysts queried databases using (SQL-based) reporting tools, end users generated data and queried and updated databases, and database administrators tuned and monitored databases and their workloads. Today, a single individual can play multiple roles in the data life cycle, and some roles may be served by crowdsourcing. Thus, human factors need to be considered for query understanding and refinement, identifying relevant and trustworthy information sources, defining and incrementally refining the data processing pipeline, visualizing relevant patterns, obtaining query answers, and making the various micro-tasks doable by domain experts and end users. We can classify people's roles into four general categories: producers, curators, consumers, and community members.

Data producers. Today, virtually anyone can generate a torrent of data from mobile phones, social platforms and applications, and wearable devices. One key challenge for the database community is to develop algorithms and incentives that guide people to produce and share the most useful data, while maintaining the desired level of data privacy. When people produce data, how can we help them add metadata quickly and accurately? For example, when a user uploads an image, Facebook automatically identifies faces in the image so users can optionally tag them. Another example is tools to automatically suggest tags for a tweet. What else can we do, and what general principles and tools can we provide?

Data curators. Data is no longer just in databases controlled by a DBA and curated by the IT department. Now, a wide variety of people are empowered to curate it. Crowdsourcing is one approach. A key challenge, then, is to obtain high-quality datasets from a process based on often-imperfect human curators. We need to build platforms that allow people to curate data easily and extend relevant applications to incorporate such curation. For these people-centric challenges, data provenance and explanation will be crucial, as will privacy and security.

Data consumers. People want to use messier data in complex ways, raising many challenges. In the enterprise, data consumers usually know how to ask SQL queries, over a structured database. Today's data consumers may not know how to formulate a query at all, for example, a journalist who wants to "find the average temperature of all cities with population over 100,000 in Florida" over a structured dataset. Enabling people to get such answers themselves requires new query interfaces, for example, based on multi-touch, not just console-based SQL. We need multimodal interfaces that combine visualization, querying, and navigation. When the query to ask is not clear, people need other ways to browse, explore, visualize, and mine the data, to make data consumption easier.

Online communities. People want to create, share, and manage data with other community members. They may want to collaboratively build community-specific knowledge bases, wikis, and tools to process data. For example, many researchers have created their own pages on Google Scholar, thereby contributing to this "community" knowledge base. Our challenge is to build tools to help communities produce usable data as well as to exploit, share, and mine it.

Community Challenges

In addition to research challenges, the database field faces many community issues. These include database education, data science, and research culture. Some of these are new, brought about by big data. Other issues, while not new, are exacerbated by big data and are becoming increasingly important.

Database education. The database technology taught in standard database courses today is increasingly disconnected from reality. It is rooted in the 1980s, when memory was small relative to database size, making I/O the bottleneck to most database operations, and when servers used relatively expensive single-core processors. Today, many databases fit in main memory, and many-core servers make parallelism and cache behavior critical to database performance. Moreover, although SQL DBMSs are still widely used, so are key-value stores, data stream processors, and MapReduce frameworks. It is time to rethink the database curriculum.

Data science. As we discussed earlier, big data has generated a rapidly growing demand for data scientists who can transform large volumes of data into actionable knowledge. Data scientists need skills not only in data management, but also in business intelligence, computer systems, mathematics, statistics, machine learning, and optimization. New cross-disciplinary programs are needed to provide this broad education. Successful research and educational efforts related to data science will require close collaboration with these other disciplines and with domain specialists. Big data presents computer science with an opportunity to influence the curricula of chemistry, earth sciences, sociology, physics, biology, and many other fields. The small computer science parts of those curricula could be grown and redirected to give data management and data science a more prominent role.

Research culture. Finally, there is much concern over the increased emphasis of citation counts instead of research impact. This discourages large systems projects, end-to-end tool building, and sharing of large datasets, since this work usually takes longer than solving point problems. Program committees that value technical depth on narrow topics over the potential for real impact are partly to blame. It is unclear how to change this culture. However, to pursue the big data agenda effectively, the field needs to return to a state where fewer publications per researcher per time unit is the norm, and where large systems projects, end-to-end tool sets, and data sharing are more highly valued.

Going Forward

This is an exciting time for database research. In the past it has been guided by, but also restricted by, the rigors of the enterprise and relational database systems. The rise of big data and the vision of a data-driven world present many exciting new research challenges related to processing big data; handling data diversity; exploiting new hardware, software, and cloud-based platforms; addressing the data life cycle, from creating data to analyzing and sharing it; and facing the diversity, roles, and number of people related to all aspects of data. It is also time to rethink approaches to education, involvement with data consumers, and our value system and its impact on how we evaluate, disseminate, and fund our research.

Acknowledgments. We thank the reviewers for invaluable suggestions. The Beckman meeting was supported by donations from the Professor Ram Kumar Memorial Foundation, Microsoft Corporation, and @WalmartLabs. C

References

1. Abiteboul, S. et al. The Lowell database research self-assessment. *Commun. ACM 48*, 5 (May 2005), 111–118.
2. Agrawal, R. et al. The Claremont report on database research. *Commun. ACM 52*, 6 (June 2009), 56–65.
3. Apache Software Foundation. Apache Hadoop; http://hadoop.apache.org, accessed Sept. 12, 2014.
4. Apache Software Foundation. Apache Hive; http://hive.apache.org, accessed on Nov. 9, 2014.
5. Apache Software Foundation. Apache Pig; http://pig.apache.org, accessed on July 4, 2014.
6. Bernstein, P. et al. Future directions in DBMS research—The Laguna Beach participants. *ACM SIGMOD Record 18*, 1 (1989), 17–26.
7. Bernstein, P. et al. The Asilomar report on database research. *ACM SIGMOD Record 27*, 4 (1998), 74–80.
8. [C11] Cattell, R. Scalable SQL and NoSQL data stores. *SIGMOD Record 39*, 4 (2011), 12–27.
9. Dean, J. and Ghemawat, S. MapReduce: Simplified data processing on large clusters. *Commun. ACM 51*, 1 (2008), 107–113.
10. Silberschatz, A. et al. Strategic directions in database systems—breaking out of the box. *ACM Computing Surveys 28*, 4 (1996), 764–778.
11. Silberschatz, A., Stonebraker, M. and Ullman, J.D. Database systems: Achievements and opportunities. *Commun. ACM 34*, 10 (Oct. 1991), 110–120.
12. Silberschatz, A., Stonebraker, M. and Ullman, J.D. Database research: Achievements and opportunities into the 21st century. *ACM SIGMOD Record 25*, 1 (1996), 52–63.

The following authors served as editors of this article (the third author also served as corresponding author):

Philip A. Bernstein (philbe@microsoft.com) is a Distinguished Scientist at Microsoft Research, Redmond, WA.

Michael J. Carey (micarey@ics.uci.edu) is a professor in the Bren School of Information and Computer Sciences at the University of California, Irvine.

AnHai Doan (anhai@cs.wisc.edu) is a professor in the Department of Computer Science at the University of Wisconsin-Madison.

ACM's Career & Job Center

Are you looking for your next IT job?

Do you need Career Advice?

The **ACM Career & Job Center** offers ACM members a host of career-enhancing benefits:

- A **highly targeted focus** on job opportunities in the computing industry
- **Access to hundreds** of industry job postings
- Resume posting **keeping you connected** to the employment market while letting you maintain full control over your confidential information
- **Job Alert system** that notifies you of new opportunities matching your criteria
- **Career coaching** and guidance available from trained experts dedicated to your success
- **Free access** to a content library of the best career articles compiled from hundreds of sources, and much more!

Visit ***ACM's Career & Job Center*** *at:*
http://jobs.acm.org

acm Association for Computing Machinery

Advancing Computing as a Science & Profession

The **ACM Career & Job Center** is the perfect place to begin searching for your next employment opportunity!

Visit today at http://jobs.acm.org

research highlights

DOI:10.1145/2856453

To view the accompanying paper, visit doi.acm.org/10.1145/2856449

Technical Perspective
Catching Lies (and Mistakes) in Offloaded Computation

By Michael Mitzenmacher and Justin Thaler

CONSIDER A CLIENT who wants to run a computer program on some dataset, but lacks the processing power to do so. The client, or verifier, thus accesses a powerful but untrusted prover, who must not only run the program and return the output, but also provide a formal guarantee the output is correct. This framework captures a wide variety of real-world scenarios. The verifier and prover may model a client and a commercial cloud computing service, or a CPU and a fast but potentially faulty coprocessor, or a peripheral device and a mainframe computer.

How can the verifier obtain a guarantee of correctness, without just ignoring the prover and executing the program locally? It is not obvious this is even possible, at least without making strong assumptions about the behavior of a "cheating" prover. Indeed, researchers have proposed many solutions that rely on such assumptions, including replication, auditing, and the use of trusted hardware. In contrast, the research area known as verifiable computing (VC) is more ambitious: it seeks solutions that make no assumptions about the behavior of a cheating prover.

Theoretical computer scientists discovered surprisingly efficient VC protocols in the late 1980s and early 1990s. While these came in several flavors (known as interactive proofs, probabilistically checkable proofs, and argument systems), they all provide the following guarantee: if the output returned by the prover is incorrect, then the verifier will catch the prover in a lie with high probability, no matter how hard the prover tries to convince the verifier otherwise. Moreover, these protocols ensure the verifier does little more than read the input, and the prover does little more than execute the program.

These discoveries had a transformative effect on the theory of computational complexity, with many consequences still being explored today. But despite their remarkable asymptotics, all these protocols were thought to be wildly impractical, and with good reason. Naive implementations would have comically high concrete costs—the prover would need millions of years to prove correctness, even for small computations, and the verifier would save time relative to local execution only on truly enormous inputs.

But the last few years have seen this viewpoint challenged, as several research groups have developed VC protocols with drastically reduced costs. These groups have pursued several different tacks, following the various theoretical approaches. The resulting collection of implementations combine algorithmic improvements with systems work to achieve costs that approach genuine practicality.

The Pinocchio system described in the following paper refines an important theoretical advance by Gennaro et al.[1] Together, these two works represent a dramatic improvement in speed, generality, and functionality. Pinocchio provides a non-interactive argument system that supports programs written in a subset of C, and automatically executes the program in a verifiable manner. Pinocchio's verifier does a one-time pre-computation to construct a public key based on the computation to be performed; if the same computer program is run on multiple inputs, the same key can be used for them all. The proofs produced by Pinocchio's prover are short (288 bytes) and quick to verify. The authors demonstrate the system's capabilities with several test programs, ranging from matrix multiplication to a lattice gas simulation.

It is worth placing Pinocchio in a broader context. The various implemented VC protocols offer a wide range of trade-offs between expressiveness, features, and efficiency—Blumberg and Walfish provided a detailed comparison of these trade-offs.[2] Broadly speaking, interactive proofs are the least general, but have the lowest costs when they apply. Argument systems, particularly non-interactive ones like Pinocchio, are more expensive: several of Pinocchio's costs remain very high, particularly the prover's runtime. In addition, the one-time pre-computation for the verifier can be orders of magnitude more expensive than local execution, meaning many inputs are required for the verifier to save work. But these expenses come with a substantial increase in generality and features. In particular, a critical feature supported by Pinocchio is zero-knowledge, best explained with an example: Suppose a computer program takes two inputs, one from the verifier and one from the prover, and the prover's input is sensitive. One might want the prover to run the program and provide the answer to the verifier, without revealing any extra information about the prover's input. A standard example is a person who wants to compare his or her salary to others', yet the actual salaries of others must be kept private. Pinocchio is the first implemented system to provide such zero-knowledge proofs.

Continued improvements in the cost of VC protocols may render them genuinely practical for a wide range of applications, thereby offering a new way to deal with issues of trust and correctness in real systems. But a crucial point is when multiple parties hold sensitive inputs, there may be no alternative to zero-knowledge proofs (in contrast, local execution is always an alternative, however unattractive, to outsourcing computation when the input is not sensitive). Thus, even if efficiency improvements tail off, there are important scenarios where systems like Pinocchio are the only option. In these settings, the efficiency improvements achieved by the authors are crucial, and the kinds of overheads we see in systems like Pinocchio may already be acceptable. C

References

1. Gennaro, R. et al. Quadratic span programs and succinct NIZKs without PCPs. EUROCRYPT, 2013, 626–645.
2. Walfish, M. and Blumberg, A.J. Verifying computations without reexecuting them: From theoretical possibility to near-practicality. *Commun. ACM 58*, 2 (Feb. 2015), 74–84.

Michael Mitzenmacher is a CS professor at Harvard, Cambridge, MA. **Justin Thaler** is a research scientist at Yahoo Labs, New York, NY.

DOI:10.1145/2856449

Pinocchio: Nearly Practical Verifiable Computation

By Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova

Abstract

To instill greater confidence in computations outsourced to the cloud, clients should be able to *verify* the correctness of the results returned. To this end, we introduce Pinocchio, a built system for efficiently verifying general computations while relying only on cryptographic assumptions. With Pinocchio, the client creates a public evaluation key to describe her computation; this setup is proportional to evaluating the computation once. The worker then evaluates the computation on a particular input and uses the evaluation key to produce a proof of correctness. The proof is only 288 bytes, regardless of the computation performed or the size of the IO. Anyone can check the proof using a public verification key.

Crucially, our evaluation on seven applications demonstrates that Pinocchio is efficient in practice too. Pinocchio's verification time is a fixed 10 ms plus 0.4–15 μs per IO element: 5–7 orders of magnitude less than previous work[23]; indeed Pinocchio is the first general-purpose system to demonstrate verification cheaper than native execution (for some apps). The worker's proof effort is still expensive, but Pinocchio reduces it by 19×–60× relative to prior work. As an additional feature, Pinocchio allows the worker to include private inputs in the computation and prove that she performed the computation correctly without revealing any information about the private inputs to the client. Finally, to aid development, Pinocchio provides an end-to-end toolchain that compiles a subset of C into programs that implement the verifiable computation protocol.

1. INTRODUCTION

Since computational power is often asymmetric (particularly for mobile devices), a relatively weak client may wish to outsource computation to one or more powerful workers. For example, a scientist might want to run a protein folding simulation in the cloud or make use of volunteer distributed computing. In such settings, the client should be able to *verify* the results returned, to guard against malicious or malfunctioning workers. Even from a legitimate worker's perspective, verifiable results are beneficial, since they are likely to command a higher price. They also allow the worker to shed liability: any undesired outputs are provably the result of data the client supplied.

Considerable systems and theory research has looked at the problem of verifying computation (Section 6). However, most of this work has either been function specific, relied on assumptions we prefer to avoid, or simply failed to pass basic practicality requirements. Function specific solutions[13, 24] are often efficient, but only for a narrow class of computations. More general solutions often rely on assumptions that may not apply. For example, systems based on replication[5] assume uncorrelated failures, while those based on Trusted Computing[19] or other secure hardware[16] assume that physical protections cannot be defeated. Finally, the theory community has produced a number of beautiful, general-purpose protocols[1, 9, 12, 15] that offer compelling asymptotics. In practice however, because many rely on complex Probabilistically Checkable Proofs (PCPs)[1] or fully homomorphic encryption (FHE),[11] the performance is currently unacceptable—verifying small instances would take millions of years (Section 5.1). Recent work[7, 22, 23] has improved these protocols considerably, but efficiency is still problematic, and the protocols lack features like public verification. Without public verification, anyone who can verify a proof can also produce a cheating proof.

In contrast, Pinocchio is a concrete system for efficiently verifying general computations while making only cryptographic assumptions. In particular, Pinocchio supports public verifiable computation (VC),[9, 20] which allows an untrusted worker to produce *signatures of computation*. Initially, the client chooses a function and generates a public evaluation key and a (small) public verification key. Given the evaluation key, a worker verifiably computes the function on an input and produces a proof (or signature) to accompany the result. Anyone (not just the client) can then use the verification key to check the correctness of the worker's result for the specific input used.

As an additional feature, Pinocchio supports zero-knowledge VC, in which the worker convinces the client that it knows one or more private inputs with a particular property, without revealing any information about the input. For example, Pantry[4] uses Pinocchio to compute MapReduce jobs (e.g., image matching) over private data (e.g., DMV photos) held by a server. Recent work also employs Pinocchio to anonymize Bitcoin transactions by proving, in zero knowledge, that the transactions do not create or destroy money.[2, 8]

Pinocchio's asymptotics are excellent: cryptographic operations required for key setup and proof generation are linear in the size of the original computation, and verification requires time linear in the size of the inputs and outputs. Even more surprising, Pinocchio's proof is constant sized, *regardless* of the computation performed. Crucially, our

The original version of this paper was published in the *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, 238–252.

evaluation (Section 5) demonstrates that these asymptotics come with small constants.

Compared with previous work,[23] Pinocchio improves verification time by 5–7 *orders of magnitude* and requires less than 10 ms for applications with reasonably sized IO, enabling Pinocchio to beat native C execution for some apps. We also improve the worker's proof efforts by 19×–60× relative to prior work. The resulting proof is tiny, 288 bytes (only slightly more than an RSA-2048 signature), regardless of the computation. Making a proof zero-knowledge is also cheap, adding negligible overhead (213 μs to key generation and 0.1% to proof generation).

While these improvements are promising, additional progress is needed before the worker's proof overhead reaches true practicality. However, even now, this overhead may be acceptable in scenarios that require high assurance, or that need the zero-knowledge properties Pinocchio supports.

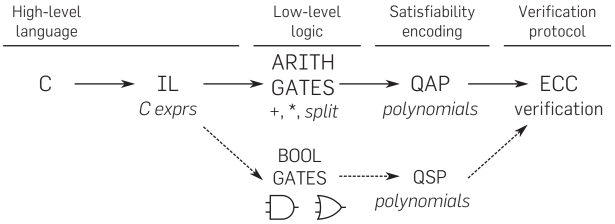
To achieve efficient VC, Pinocchio combines *quadratic programs*, a computational model introduced by Gennaro et al.,[10] with a series of theoretical refinements and systems engineering to produce an end-to-end toolchain for verifying computations. Specifically, via an improved protocol and proof technique relative to Gennaro et al., we slash the cost of key generation by 61%, and the cost of producing a proof by 64%. From a developer's perspective, Pinocchio provides a compiler that transforms C code into a circuit representation (we support both Boolean and arithmetic), converts the circuit into a quadratic program, and then generates programs to execute the cryptographic protocol (Figure 1).

Pinocchio's end-to-end toolchain allows us to implement real applications that benefit from verification. In particular, we implement two forms of matrix multiplication, multivariate polynomial evaluation, image matching, all-pairs shortest paths, a lattice-gas scientific simulator, and SHA-1. We find (Section 5) that the first three apps translate efficiently into arithmetic circuits, and hence Pinocchio can verify their results faster than native execution of the same program. The latter four apps translate less efficiently, due to their reliance on inequality comparisons and bitwise operations, and yet they may still be useful for zero-knowledge applications.

In summary, this paper contributes:

1. An end-to-end system for efficiently verifying computation performed by one or more untrusted workers. This includes a compiler that converts C code into a format suitable for verification, as well as a suite of tools for running the actual protocol.
2. Theoretical and systems-level improvements that bring performance down by 5–7 orders of magnitude relative to prior work,[23] and hence into the realm of plausibility.
3. An evaluation on seven real C apps, showing verification faster than 32-bit native integer execution for some apps.

Figure 1. Overview of Pinocchio's Toolchain. Pinocchio takes a high-level C program all the way through to a distributed set of executables that run the program in a verified fashion.



2. BACKGROUND

2.1. Verifiable computation (VC)

A public VC scheme allows a computationally limited client to outsource to a worker the evaluation of a function F on input u. The client can then verify the correctness of the returned result $F(u)$ while performing less work than required for the function evaluation.

More formally, we define public VC as follows, generalizing previous definitions.[9, 10, 20]

Definition 1 (Public Verifiable Computation). *A public verifiable computation scheme $\mathcal{VC}$ consists of a set of three polynomial-time algorithms* (KeyGen, Compute, Verify)*:*

- $(EK_F, VK_F) \leftarrow \mathsf{KeyGen}(F, 1^\lambda)$*: The randomized key generation algorithm takes the function F to be outsourced and security parameter λ; it outputs a public evaluation key EK_F, and a public verification key VK_F.*
- $(y, \pi_y) \leftarrow \mathsf{Compute}(EK_F, u)$*: The deterministic worker algorithm uses the public evaluation key EK_F and input u. It outputs $y \leftarrow F(u)$ and a proof π_y of y's correctness.*
- $\{0, 1\} \leftarrow \mathsf{Verify}(VK_F, u, y, \pi_y)$*: Given the verification key VK_F, the deterministic verification algorithm outputs 1 if $F(u) = y$, and 0 otherwise.*

Prior work gives formal definitions for correctness, security, and efficiency,[10] so we merely summarize:

- **Correctness**. For any function F, and any input u to F, if we generate keys for F, and run Compute with the resulting evaluation key EK_F, then Verify will always accept.
- **Security**. For any function F and any probabilistic polynomial-time adversary, the adversary cannot produce a proof for IO $\hat{u}, \hat{y}$ such that $F(\hat{u}) \neq \hat{y}$ but Verify accepts the proof.
- **Efficiency.** KeyGen is assumed to be a one-time operation whose cost is amortized over many calculations, but we require that Verify is cheaper than evaluating F.

Several previous VC schemes[9] were not public, but rather *designated verifier*, meaning that the verification key VK_F must be kept secret. Indeed, in these schemes, even revealing the output of the verification function (i.e., whether or not the worker had been caught cheating) could lead to attacks on the system. A public VC scheme avoids such issues.

Zero-Knowledge Verifiable Computation. We also consider an extended setting where the outsourced computation is a function, $F(u, w)$, of two inputs: the client's input u and an auxiliary input w from the worker. A VC scheme is *zero-knowledge* if the client learns nothing about the worker's input beyond the output of the computation.

2.2. Quadratic programs

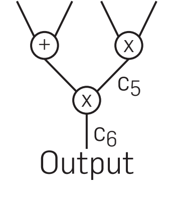
Gennaro, Gentry, Parno, and Raykova (GGPR) showed how to compactly encode computations as quadratic programs,[10] so as to obtain efficient VC and zero-knowledge VC schemes. Specifically, they show how to convert any arithmetic circuit into a comparably sized Quadratic Arithmetic Program (QAP).

Standard results show that polynomially sized circuits are equivalent (up to a logarithmic factor) to Turing machines that run in polynomial time, though of course the actual efficiency of computing via circuits versus on native hardware depends heavily on the application; for example, an arithmetic circuit for matrix multiplication adds essentially no overhead, whereas a Boolean circuit for integer multiplication is far less efficient than executing a single 32-bit assembly instruction.

An arithmetic circuit consists of wires that carry values from a field $\mathbb{F}$ and connect to addition and multiplication gates—see Figure 2 for an example.

Before formally defining QAPs, we walk through the steps for encoding the circuit in Figure 2 into an equivalent QAP. First, we select two arbitrary values, $r_5, r_6 \in \mathbb{F}$ to represent the two multiplication gates (the addition gates will be compressed into their contributions to the multiplication gates). We define three sets of polynomials $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$ by letting the polynomials in $\mathcal{V}$ encode the left input into each multiplication gate, the $\mathcal{W}$ encode the right input into each gate, and the $\mathcal{Y}$ encode the outputs. Thus, for the circuit in Figure 2, we define six polynomials for each set $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$, four for the input wires, and two for the outputs from the multiplication gates. We define these polynomials based on each wire's contributions to the multiplication gates. Specifically all of the $v_k(r_5) = 0$, except $v_3(r_5) = 1$, since the third input wire contributes to the left input of c_5's multiplication gate. Similarly, $v_k(r_6) = 0$, except for $v_1(r_6) = v_2(r_6) = 1$, since the first two inputs both contribute to the left input of c_6's gate. For $\mathcal{W}$, we look at right inputs. Finally, $\mathcal{Y}$ represents outputs; none of the input wires is an output, so $y_k(r_5) = y_k(r_6) = 0$ for $k \in \{1, \ldots, 4\}$, and $y_5(r_5) = y_6(r_6) = 1$. As we explain below, we can use this encoding of the circuit to efficiently check that it was evaluated correctly.

Figure 2. Arithmetic Circuit and Equivalent QAP. Each wire value comes from, and all operations are performed over, a field $\mathbb{F}$. The polynomials in the QAP are defined in terms of their evaluations at the two roots, r_5 and r_6. See text for details.



	(r_5, r_6)		(r_5, r_6)		(r_5, r_6)
$v_1(r_i)$	(0,1)	$w_1(r_i)$	(0,0)	$y_1(r_i)$	(0,0)
$v_2(r_i)$	(0,1)	$w_2(r_i)$	(0,0)	$y_2(r_i)$	(0,0)
$v_3(r_i)$	(1,0)	$w_3(r_i)$	(0,0)	$y_3(r_i)$	(0,0)
$v_4(r_i)$	(0,0)	$w_4(r_i)$	(1,0)	$y_4(r_i)$	(0,0)
$v_5(r_i)$	(0,0)	$w_5(r_i)$	(0,1)	$y_5(r_i)$	(1,0)
$v_6(r_i)$	(0,0)	$w_6(r_i)$	(0,0)	$y_6(r_i)$	(0,1)

$t(x) = (x - r_5)\,(x - r_6)$

More generally, we define a QAP, an encoding of an arithmetic circuit, as follows.

Definition 2 (Quadratic Arithmetic Program (QAP)[10]). *A QAP Q over field $\mathbb{F}$ contains three sets of $m + 1$ polynomials $\mathcal{V} = \{v_k(x)\}$, $\mathcal{W} = \{w_k(x)\}$, $\mathcal{Y} = \{y_k(x)\}$, for $k \in \{0 \ldots m\}$, and a target polynomial $t(x)$. Suppose F is a function that takes as input n elements of $\mathbb{F}$ and outputs n′ elements, for a total of $N = n + n'$ I/O elements. Then we say that* Q *computes F if:* $(c_1, \ldots, c_N) \in \mathbb{F}^N$ *is a valid assignment of F's inputs and outputs,* if and only if *there exist coefficients $(c_{N+1}, \ldots, c_m)$ such that $t(x)$ divides $p(x)$, where:*

$$p(x) = \left(v_0(x) + \sum_{k=1}^{m} c_k \cdot v_k(x)\right) \cdot \left(w_0(x) + \sum_{k=1}^{m} c_k \cdot w_k(x)\right) - \left(y_0(x) + \sum_{k=1}^{m} c_k \cdot y_k(x)\right). \quad (1)$$

In other words, there must exist some polynomial $h(x)$ such that $h(x) \cdot t(x) = p(x)$. The size of Q is m, and the degree is the degree of $t(x)$.

Building a QAP Q for a general arithmetic circuit C is fairly straightforward. We pick an arbitrary root $r_g \in \mathbb{F}$ for each multiplication gate g in C and define the target polynomial to be $t(x) = \Pi_g(x - r_g)$. We associate an index $k \in [m] = \{1 \ldots m\}$ to each input of the circuit and to each output from a multiplication gate. Finally, we define the polynomials in $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$ by letting the polynomials in $\mathcal{V}$ encode the left input into each gate, the $\mathcal{W}$ encode the right input into each gate, and the $\mathcal{Y}$ encode the outputs. For example, $v_k(r_g) = 1$ if the kth wire is a left input to gate g, and $v_k(r_g) = 0$ otherwise. Similarly, $y_k(r_g) = 1$ if the kth wire is the output of gate g, and $y_k(r_g) = 0$ otherwise. Thus, if we consider a particular gate g and its root r_g, Equation (1) simplifies to: $\left(\sum_{k=1}^{m} c_k \cdot v_k(r_g)\right) \cdot \left(\sum_{k=1}^{m} c_k \cdot w_k(r_g)\right) = \left(\sum_{k \in I_{left}} c_k\right) \cdot \left(\sum_{k \in I_{right}} c_k\right) = c_g y_k(r_g) = c_g$, which just says that the output value of the gate is equal to the product of its inputs, the very definition of a multiplication gate. For example, in the QAP for the circuit in Figure 2, if we evaluate $p(x)$ at r_5, we get $(c_3) \cdot (c_4) = c_5$, which directly encodes the first multiplication gate, and similarly, at r_6, $p(x)$ simplifies to $(c_1 + c_2) \cdot (c_5) = c_6$, that is, an encoding of the second multiplication gate.

In short, the divisibility check that $t(x)$ divides $p(x)$ decomposes into $\deg(t(x))$ separate checks, one for each gate g and root r_g of $t(x)$, that $p(r_g) = 0$.

The actual construction[10] is a bit more complex, as it handles addition and multiplication by constants. Nonetheless, GGPR show that for any arithmetic circuit with d multiplication gates and N I/O elements, one can construct an equivalent QAP with degree (the number of roots r_g) d and size (number of polynomials in each set) $d + N$. Note that addition gates and multiplication-by-constant gates do not contribute to the size or degree of the QAP. Thus, these gates are essentially "free" in QAP-based VC schemes.

Strong QAPs. In their QAP-based VC scheme, described below, GGPR unfortunately require a strong property from the QAP. Note that Definition 2 only considers the case where the same set of coefficients c_i are applied to all three sets of polynomials. GGPR additionally require the if-and-only-if condition in Definition 2 to hold even when different coefficients a_i, b_i, c_i are applied—that is, when $p(x) = \left(\Sigma_{k=1}^{m} c_k \cdot v_k(x)\right) \cdot \left(\Sigma_{k=1}^{m} b_k \cdot w_k(x)\right) - \left(\Sigma_{k=1}^{m} a_k \cdot y_k(x)\right)$. They show how to convert any QAP into a *strong QAP* that satisfies this stronger condition. Unfortunately, this strengthening step increases the QAP's degree to $3d + 2N$, more than *tripling* it. This in turn, more than triples the cost of key generation, the size of the evaluation key, and the worker's effort to produce a proof.

2.3. Building VC from quadratic programs

To construct a VC protocol from a quadratic program, we map each polynomial—for example, $v_k(x)$—of the quadratic program to an element $g^{v_k(s)}$ in an elliptic curve group $\mathbb{G}$, where s is a secret value selected by the client, and g is a generator of $\mathbb{G}$. These group elements are given to the worker. For a given input, the worker evaluates the circuit directly to obtain the output and the values of the internal circuit wires. These values correspond to the coefficients c_i of the quadratic program. Thus, the VC worker can evaluate $v(s) = \Sigma_{k \in [m]} c_k \cdot v_k(s)$ "in the exponent" to get $g^{v(s)}$; it computes $w(s)$ and $y(s)$, in the exponent, similarly.

To allow the worker to prove that Equation (1) holds, we also, as part of the evaluation key, give the worker $g^{(s^i)}$ terms. The worker computes $h(x) = p(x)/t(x) = \sum_{i=0}^{d} h_i \cdot x^i$, and then uses the h_i, along with $g^{(s^i)}$ terms, to compute $g^{h(s)}$. To oversimplify, the proof consists of $(g^{v(s)}, g^{w(s)}, g^{y(s)}, g^{h(s)})$. To check that $p(s) = h(s)t(s)$, the verifier uses a *bilinear map* that allows him to take two elliptic curve elements and "multiply" their exponents together to create an element in a new group. The actual protocol[10] is a bit more complex, because additional machinery is needed to ensure that the worker incorporates the client's input u correctly, and that the worker indeed generates (say) $v(s)$ in the exponent as some linear function of the $v_k(s)$ values.

Regarding efficiency, GGPR[10] show that the one-time setup of KeyGen runs in time linear in the original circuit size, $O(|C|)$. The worker performs $O(|C|)$ cryptographic work, but he must also perform $O(|C| \log^2 |C|)$ noncryptographic work to calculate $h(x)$. To achieve this performance, the worker exploits the fact that the evaluation vectors $(v_k(r_1), \ldots, v_k(r_d))$ are all very sparse (also for the w and y polynomials). The proof itself is constant size, with only 9 group elements for QAPs, though the verifier's work is still linear, $O(N)$, in the size of the inputs and outputs of the function.

In terms of security, GGPR[10] show this VC scheme is sound under the d-PKE and q-PDH assumptions, which are weak versions of assumptions in prior work.

Zero Knowledge. Making the VC scheme zero-knowledge is remarkably simple. One simply includes the target polynomial $t(x)$ itself in the polynomial sets $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$. This allows the worker to "randomize" its proof by adding $\delta_v t(s)$ in the exponent to $v_{mid}(s)$, $\delta_w t(s)$ to $w(s)$, and $\delta_y t(s)$ to $y(s)$ for random δ_v, δ_w, δ_y, and modifying the other elements of the proof accordingly. The modified value of $p(x)$ remains divisible by $t(x)$, but the randomization makes the scheme statistically zero-knowledge.[10]

3. THEORETICAL REFINEMENTS

In this section, we improve GGPR's protocol[10] to significantly reduce key generation time, evaluation key size, and worker effort. We analyze our improvements empirically in Section 5.

Our main optimization is that we construct a VC scheme that uses a *regular* QAP (as in Definition 2), rather than a *strong* QAP. Recall that GGPR show how to transform a regular QAP into a strong QAP, but the transformation more than *triples* the degree of the QAP. Consequently, when they plug their strong QAP into their VC construction, the strengthening step more than triples the key generation time, evaluation key size, and worker computation. We take a different approach that uses a regular QAP, and hence we do not need a strengthening step at all. Instead, we embed additional structure into our new VC proof that ensures that the worker uses the same linear combination to construct the v, w, and y terms of its proof.[a] Surprisingly, this additional structure comes at no cost, and our VC scheme is actually *less* complicated than GGPR's! Finally, we expand the expressivity and efficiency of the functions QAPs can compute by designing a number of custom circuit gates for specialized functions.

3.1. Our new VC protocol

Next we describe our more efficient VC scheme, with some remarks afterwards on some its properties.

Protocol 1 (Verifiable Computation from regular QAPs).

- $(EK_F, VK_F) \leftarrow$ KeyGen$(F, 1^\lambda)$: *Let F be a function with N input/output values from $\mathbb{F}$. Convert F into an arithmetic circuit C; then build the corresponding QAP $Q = (t(x), \mathcal{V}, \mathcal{W}, \mathcal{Y})$ of size m and degree d. Let $I_{mid} = \{N+1, \ldots, m\}$, that is, the non-IO-related indices.*
  Let e be a non-trivial bilinear map $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, and let g be a generator of G.
  Choose r_v, r_w, s, α_v, α_w, α_y, β, γ at random from $\mathbb{F}$ and set $r_y = r_v \cdot r_w$, $g_v = g^{r_v}$, $g_w = g^{r_w}$ and $g_y = g^{r_y}$.
  Construct the public evaluation key EK_F as:

$$\begin{aligned}\Big(&\{g_v^{v_k(S)}\}k \in I_{mid}, \quad \{g_w^{w_k(S)}\}k \in I_{mid}, \quad \{g_y^{y_k(S)}\}k \in I_{mid}, \\ &\{g_v^{\alpha_v v_k(S)}\}k \in I_{mid}, \{g_w^{\alpha_w w_k(S)}\}k \in I_{mid}, \{g_y^{\alpha_y y_k(S)}\}k \in I_{mid}, \\ &\{g^{S^i}\}i \in [d], \qquad \{g_v^{\beta v_k(S)} g_w^{\beta w_k(S)} g_y^{\beta y_k(S))}\}k \in I_{mid} \Big),\end{aligned}$$

  and the public verification key as: $VK_F = (g^1, g^{\alpha_v}, g^{\alpha_w}, g^{\alpha_y}, g^{\gamma}, g^{\beta\gamma}, g_y^{t(s)}, \{g_v^{v_k(s)}, g_w^{w_k(s)}, g_y^{y_k(s)}\}k \in \{0\} \cup [N])$.
- $(y, \pi_y) \leftarrow$ Compute(EK_F, u): *On input u, the worker evaluates the circuit for F to obtain $y \leftarrow F(u)$; he also learns the values*

[a] Our proof contains a term that enforces this linear constraint without increasing the degree. GGPR's generic strengthening step checked the consistency of the linear combinations via additional multiplication gates, which increased the degree of the QAP.

$\{c_i\}_{i\in[m]}$ of the circuit's wires.
He solves for $h(x)$ (the polynomial such that $p(x) = h(x)\cdot t(x)$), and computes the proof π_y as:

$$\begin{array}{lllll}(& g_v^{v_{mid}(S)}, & g_w^{w_{mid}(S)}, & g_y^{y_{mid}(S)}, & g^{h(S)}, \\ & g_v^{\alpha_v v_{mid}(S)}, & g_w^{\alpha_w w_{mid}(S)}, & g_y^{\alpha_y y_{mid}(S)} & \\ & g_v^{\beta_v v_{mid}(S)} & g_w^{\beta w_{mid}(S)} & g_y^{\beta y_{mid}(S)} &),\end{array}$$

where $v_{mid}(x) = \Sigma k \in I_{mid}\ c_k \cdot v_k(x)$, and similarly for $w_{mid}(s)$ and $y_{mid}(s)$. Since these are linear equations, he can compute them "in the exponent" using the material in the evaluation key, for example, $g^{v_m id(s)} = \prod_{k\in I_{mid}} (g^{v_k(s)})^{c_k}$.

- $\{0,1\} \leftarrow$ Verify(VK_F, u, y, π_y)*: The verification of an alleged proof with elements $g^{V_{mid}}, g^{W_{mid}}, g^{Y_{mid}}, g^H, g^{V'_{mid}}, g^{W'_{mid}}, g^{Y'_{mid}}$, and g^Z uses the public verification key VK_F and the pairing function e for the following checks.*
  - *Divisibility check for the QAP: using elements from VK_F, compute a term representing the I/O, u and y, by representing them as coefficients $c_1, \ldots, c_N \in \mathbb{F}$ and computing: $g_v^{v_{io}(s)} = \prod_{k\in[N]} \left(g_v^{v_k(s)}\right)^{c_k}$ (and similarly for $g_w^{w_{io}(s)}$ and $g_y^{y_{io}(s)}$). Check:*

$$e\left(g_v^{v_0(s)} g_v^{v_{io}(s)} g_v^{V_{mid}}, g_w^{w_0(s)} g_w^{w_{io}(s)} g_w^{W_{mid}}\right) = \tag{2}$$

$$e\left(g_y^{t(s)}, g^H\right) e\left(g_y^{y_0(s)} g_y^{y_{io}(s)} g_y^{Y_{mid}}, g\right). \tag{3}$$

  - *Check that the linear combinations computed over $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$ are in their appropriate spans:*

$$e\left(g_v^{V'_{mid}}, g\right) = e\left(g_v^{V_{mid}}, g^{\alpha_v}\right), \quad e\left(g_w^{W'_{mid}}, g\right) = e\left(g_W^{W_{mid}}, g^{\alpha_W}\right),$$
$$e\left(g_y^{Y'_{mid}}, g\right) = e\left(g_y^{Y_{mid}}, g^{\alpha_y}\right).$$

  - *Check that the same coefficients were used in each of the linear combinations over $\mathcal{V}$, $\mathcal{W}$, and $\mathcal{Y}$:*

$$e\left(g^Z, g^\gamma\right) = e\left(g_v^{V_{mid}} g_w^{W_{mid}} g_y^{Y_{mid}}, g^{\beta\gamma}\right).$$

  In a designated verifier setting (where the verifier knows s, α, etc.), pairings are only needed for divisibility check, and the I/O term can be computed directly over $\mathbb{F}$, rather than "in the exponent."

The correctness of the VC scheme follows from the properties of the QAP. Regarding security, we have the following:

Theorem 1. *Let d be an upper bound on the degree of the QAP used in the VC scheme, and let q = 4d + 4. The VC scheme is sound under the d-PKE, q-PDH, and 2q-SDH assumptions.*

The proof of Theorem 1 is in the full version of the paper.

Security Intuition. As intuition for why the VC scheme is sound, note that it seems hard for an adversary who does not know α to construct *any* pair of group elements h, h^α *except* in the obvious way: by taking pairs $(g_1, g_1^\alpha), (g_2, g_2^\alpha), \ldots$ that he is given, and applying the same linear combination (in the exponent) to the left and right elements of the pairs. This hardness is formalized in the d-PKE assumption, a sort of "knowledge-of-exponent" assumption, that says that the adversary must "know" such a linear combination, in the sense that this linear combination can be extracted from him. Roughly, this means that, in the security proof, we can extract polynomials $V_{mid}(x)$, $W_{mid}(x)$, $Y_{mid}(x)$ such that V_{mid} (from the proof) equals $V_{mid}(s)$, $W_{mid} = W_{mid}(s)$ and $Y_{mid} = Y_{mid}(s)$, and that moreover these polynomials are in the linear spans of the $v_k(x)$'s, $w_k(x)$'s, and $y_k(x)$'s, respectively. If the adversary manages to provide a proof of a false statement that verifies, then these polynomials must not actually correspond to a QAP solution. So, either $p(x)$ is not actually divisible by $t(x)$ (in this case we break $2q$-SDH) or $V(x) = v_{io}(x) + V_{mid}(x)$, $W(x)$ and $Y(x)$ do not use the same linear combination (in this case we break q-PDH because in the proof we choose β in a clever way).

Zero Knowledge. We can apply GGPR's rerandomization technique[10] (Section 2.3) to provide statistical zero-knowledge for our new VC construction. The worker chooses $\delta_v, \delta_w, \delta_y \xleftarrow{R} \mathbb{F}$ and in his proof, instead of the polynomials $v_{mid}(x)$, $v(x)$, $w(x)$, and $y(x)$, he uses the following randomized versions $v_{mid}(x) + \delta_v t(x)$, $v(x) + \delta_v t(x)$, $w(x) + \delta_w t(x)$, and $y(x) + \delta_y t(x)$.

Performance. Our main improvement is that our VC scheme only requires a regular QAP, rather than a strong QAP, which improves performance by more than a factor of 3. Moreover, the scheme itself is simpler, leading to fewer group elements in the keys and proof, fewer bilinear maps for Verify, etc.

3.2. Expressive circuit constructions

The QAP that we use in our VC scheme is defined over $\mathbb{F}_p$, where p is a large prime. We can, as explained previously, derive a QAP over $\mathbb{F}_p$ that efficiently computes any function F that can be expressed in terms of addition and multiplication modulo p. This provides no obvious way to express some operations, such as $a \geq b$ using mod-p arithmetic. On the other hand, given a and b as bits, comparison is easy. Hence, one might infer that Boolean circuits are more general.

However, we design an arithmetic *split gate* to translate an arithmetic wire $a \in \mathbb{F}_p$, known to be in $[0, 2^k - 1]$, into k binary output wires. Given such binary values, we can compute Boolean functions using arithmetic gates: NAND(a, b) = 1 − ab, AND(a, b) = ab, OR(a, b) = 1 − (1 − a)(1 − b). Each embedded Boolean gate costs only one multiply.

Surprisingly, this arithmetic embedding gives a fairly efficient VC scheme. Embedding introduces an expensive initial gate that constrains each input to $\{0, 1\}$, but henceforth, each embedded gate preserves the $\{0, 1\}$ invariant, adding only 1 to the degree and size of the QAP. Furthermore, the expression $\sum_{i=1}^{k} 2^{i-1} a_i$ combines a bitwise representation of a back into a single wire. Because the sum consists of additions and multiplications by constants, recombination is free; it doesn't increase the size of the QAP.

In our full paper, we also design a gate that enforces equality between two wires and a gate that checks whether a wire is equal to zero. These can be composed (Thm 11 in Ref.[10]) with other gates.

4. IMPLEMENTATION

We implemented a compiler that takes a subset of C to an equivalent arithmetic circuit (Section 4.1). Our VC suite then compiles the circuit representation to the equivalent QAP, and generates code to run the VC protocol, including key generation, proof computation, and proof verification (Section 4.2). The toolchain compiles a large collection of applications and runs them with verification (Section 4.3). Source code for the toolchain is available.[b]

4.1. Compiler toolchain

The applications described below (Section 4.3) and evaluated in Section 5 are each compiled using *qcc*, our C-to-arithmetic-expression compiler, a 3525-line Python program. They are also compiled with gcc to produce the Native timings in Figures 5 and 6.

The compiler understands a substantial subset of C, including global, function, and block-scoped variables; arrays, structs, and pointers; function calls, conditionals, loops; and static initializers (Figure 3). It also understands arithmetic and bitwise Boolean operators and preprocessor syntax.

Since the "target machine" (arithmetic circuits) supports only expressions, not mutable state and iteration, we restrict the C program's semantics accordingly. For example, pointers and array dereferences must be compile-time constants; otherwise, each dynamic reference would produce conditional expressions of size proportional to the addressable memory. Function calls are inlined, while preserving C variable scope and pointer semantics.

Imperative conditionals compile to conditional expressions that encode the imperative side effects. Static conditions are collapsed at compile time. Similarly, loops with statically computable termination conditions are automatically unrolled completely.

The only scalar type presently supported is **int**; a compiler flag selects the integer size. The compiler inserts masking expressions to ensure that a k-bit int behaves exactly as the corresponding C type, including overflow.

The compiler's intermediate language is a set of expressions of C-like operators, such as +, *, <=, ? :, &, and ^.

The compiler back-end expands each expression into the arithmetic gate language of mul, add, const-mul, wire-split, etc., eliminating common subexpressions. It carefully bounds the bit-width of each wire value:

- inputs have the compiler-specified **int** width;
- each constant has a known width (e.g., $13 = 1101_2$ has bit width 4);
- a bitwise op produces the *max* of its arguments' widths;
- add can produce *max* + 1 bits (for a carry); and
- mul can produce $2 \cdot max$ bits.

When the width nears the available bits in the field (254), the compiler generates a split gate to truncate the value back to the specified **int** width. Tracking bit width minimizes the cost of split gates.

Figure 3. Fixed-Matrix Multiplication. The qcc compiler unrolls the loops and decodes the struct and array references to generate an arithmetic expression for Out in terms of In.

```
int mat[SIZE*SIZE] = { 0x12, ... };
struct In { int vector[SIZE]; };
struct Out { int result[SIZE]; };

void compute(struct In *in, struct Out *out){
  int i, j, k, t;
  for (i=0; i<SIZE; i+=1) {
    int t=0;
    for (k=0; k<SIZE; k+=1) {
      t = t + mat->[i*SIZE+k] * in->vector[k];
    }
    out->result[i] = t;
  }
}
```

4.2. Quadratic programs and cryptographic protocol

The next pipeline stage accepts a Boolean or arithmetic circuit and builds a QSP or QAP (Section 2). Then, per Section 3.1, it compiles the quadratic program into a set of cryptographic routines for the client (key generation and verification) and the worker (computation and proof generation). For comparison, we also implement the original GGPR[10]; GGPR protocol Section 5 shows that Pinocchio's enhancements reduce overhead by 18–64%.

The key-generation routine runs at the client, with selectable public verification and zero-knowledge features (Section 5.2). The code transmits the evaluation key over the network to the worker; to save bandwidth, the program transmits as C and the worker compiles it locally.

The computation routine runs at the server, collecting input from the client, using the evaluation key to produce the proof, and transmitting the proof back to the client (or, if desired, a different verifier). The verification routine uses the verification key and proof to determine if the worker cheated.

Our cryptographic code is single-threaded, but each stage is embarrassingly parallel. Prior work[23] shows that standard techniques can parallelize work across cores, machines, or GPUs. For the cryptographic code, we use a high-speed elliptic curve library[18] with a 256-bit BN-curve that provides 128 bits of security. The quadratic-program-construction and protocol-execution code is 10,832 lines of C and C++.

Faster Exponentiation. Generating the evaluation key *EK* requires exponentiating the same base g to many different powers. We optimize this operation by adapting Pippenger's multi-exponential algorithm for use with a single base. Essentially this means that we build a table of intermediate powers of g, allowing us to compute any particular exponent with only a few multiplications.

In a similar vein, the worker's largest source of overhead is applying the coefficients from the circuit "in the exponent" to compute $g^{Y(s)}$, etc. We optimize this operation via a sliding-window technique to build a small table of powers for each pair of bases. In practice, these tables can improve performance by a factor of three to four, even counting the time to build the tables in the first place.

Polynomial Asymptotics. To generate a proof, the worker

[b] https://vc.codeplex.com.

must compute the polynomial $h(x)$ such that $t(x) \cdot h(x) = P(x)$ (Section 1). Since we store $P(x)$ in terms of its evaluations at the roots of the quadratic program (recall Figure 2), the worker must first interpolate to find $P(x)$ and then perform a polynomial division to arrive at $h(x)$.

Note that all of these computations take place in a normal field, whereas all of the worker's other steps involve cryptographic operations, which are about three orders of magnitude more expensive.

Thus, one might naïvely conclude, as we did, that simple polynomial algorithms, such as Lagrangian interpolation and "high-school" polynomial multiplication, suffice. However, we quickly discovered that the $O(n^2)$ behavior of these algorithms, at the scale required for verifiable computing, dwarfed the linear number of cryptographic operations (Section 5). Hence we implemented an FFT-based $O(n\log n)$ polynomial multiplication library and used a polynomial interpolation algorithm that builds a binary tree of polynomials, giving total time $O(n\log^2 n)$. Even so optimized, solving for $h(x)$ is the second largest source of worker overhead.

Preparing for the Future; Learning from the Past. In our implementation and evaluation, we assume a worst case scenario in which the client decides, without any warning, to outsource a new function, and similarly that the worker only ever computes a single instance for a given client. In practice, neither scenario is plausible. When the client first installs Pinocchio, the program, could build the single base exponent table discussed earlier. Further, it can choose a random s and begins computing powers of s in the background, since these are entirely independent of the computation. The worker can optimize similarly, given the client's key.

4.3. Applications

Pinocchio runs several applications; each can be instantiated with some static *parameters*, and then each instance can be executed with dynamic *inputs*. While it may be possible to use custom verification checks for some of these applications (e.g., matrix multiplication), we include them to illustrate their performance within a general-purpose system like Pinocchio.

Fixed Matrix multiplies an $n \times n$ matrix parameter M by an n-length input vector A, and outputs the resulting n-length vector $M \cdot A$. We choose five parameter settings that range from $|M| = 200 \times 200$ to $|M| = 1000 \times 1000$.

Two Matrices has parameter n, takes as input two $n \times n$ matrices M_1 and M_2, and outputs the $n \times n$ matrix $M_1 \cdot M_2$. Matrix operations are widely used, for example, in collaborative filtering ($|M| = 30 \times 30$ to $|M| = 110 \times 110$).

MultiVar Poly evaluates a k-variable, m-degree multivariate polynomial. The $(m + 1)^k$ coefficients are parameters, the k variables $x_1, \ldots, x_k$ are the inputs, and the polynomial's scalar value is the output ($k = 5$, $m = 6$, 16,807 coeff. to $k = 5$, $m = 10$; 644,170 coeff.).

Image Matching is parameterized by an $i_w \times i_h$ rectangular image and parameters k_w, k_h. It takes as input a $k_w \times k_h$ image kernel, and outputs the minimum difference and the point (x, y) in the image where it occurs ($i_w \times i_h = 25$, $k_w \times k_h = 9$ to $i_w \times i_h = 2025$, $k_w \times k_h = 9$).

Shortest Paths implements the Floyd-Warshall $O(n^3)$ graph algorithm, useful for network routing and matrix inversion. Its parameter n specifies the number of vertices, its input is an $n \times n$ edge matrix, and its output is an $n \times n$ matrix of all-pairs shortest paths ($n = 8$, $e = 64$ to $n = 24$, $e = 576$).

LGCA is a Lattice-Gas Cellular Automata implementation that converges to Navier-Stokes. It has parameter n, the fluid lattice size, and k, the iteration count. It inputs one n-cell lattice and outputs another reflecting k steps ($n = 294$, $k = 5$ to $n = 294$, $k = 40$).

SHA-1 has no parameters. Its input is a 13-word (416-bit) input string, and it outputs its 5-word (160-bit) SHA-1 hash.

5. EVALUATION

We experiment on a Lenovo X201 ThinkPad. We run on a single core of a 2.67 GHz Intel Core i7 with 8 GB of RAM.

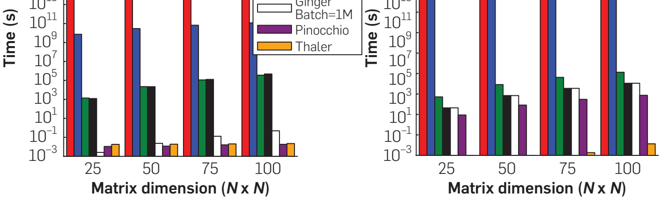
Below, we focus on comparisons with previous work and app-level performance. In the full paper, we present microbenchmarks to quantify the basic cost units of our protocol. Our results show that the optimizations described in Section 4.2.1 reduce costs by 2–3 orders of magnitude for polynomial operations, and factors of 3–10 for exponentiations. At the macro level, relative to the original GGPR protocol, KeyGen and Compute are more than twice as fast, and even verification is 24% faster. Pinocchio also drastically reduces the size of the evaluation key and even manages to reduce the size of GGPR's already svelte 9 element proof to 8 elements.

5.1. Comparison with related work

Figure 4 plots Pinocchio's performance against that of related systems. We use the multiplication of two matrices as our test application since it has appeared in several prior papers, though simpler, non-cryptographic verification procedures exist. Since all of these prior schemes are designated verifier, we measure against Pinocchio's designated verifier mode.

We compare against (1) a naïve version of a PCP-based scheme[22]; (2) GGP,[9] an early scheme that defined VC, but which relies on FHE; (3) Pepper,[22] an optimized refinement of (1); (4) Ginger,[23] a further refinement of Pepper; (5) Ginger with a batch of one million simultaneous

Figure 4. Performance Relative to Related Schemes. Pinocchio reduces costs by orders of magnitude (note the log scale on the *y*-axis). We graph the time necessary to (a) verify and (b) produce a proof result for multiplying two *N* × *N* matrices.



(a) Per-Instance Verification Latency

(b) Worker Latency

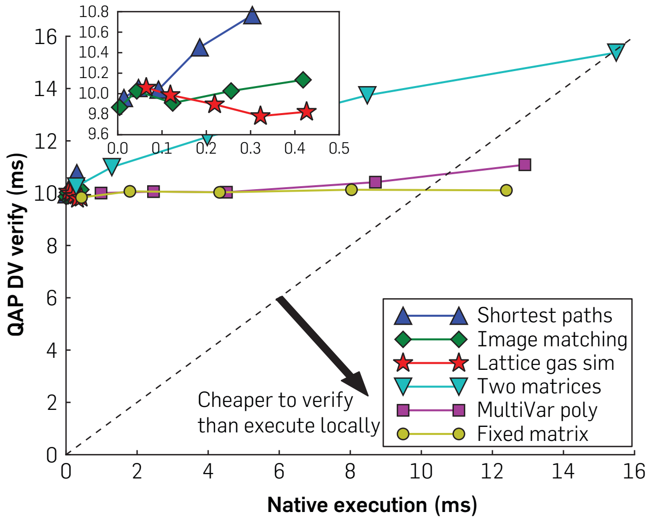
instances (see below); and (6) a subsequent system by Thaler,[25] tailored specifically for matrix multiplication and extending work based on interactive protocols.[7, 12] See Section 6 for more details on these schemes and the tradeoffs between them. Since most of these schemes are ridiculously impractical, we model, rather than measure, their performance. For GGP, we built a model of its performance based on recent performance results for FHE; for Thaler, we extrapolated from reported results[25]; while for the others, we used previously published models.[22, 23] For Pinocchio, however, we use real numbers from our implementation.

Figure 4 shows that Pinocchio continues the recent trend of reducing costs by orders of magnitude. A naive PCP-based scheme requires trillions of years to produce or verify a single proof. The FHE-based GGP protocol improves this performance significantly but remains impractical. Pepper and Ginger have made huge improvements over prior work, but, as we discuss in more detail in Section 6, they do not offer public verification or zero knowledge.

In addition to offering new properties, Pinocchio significantly improves performance and security. Except for Thaler's work, the systems shown in Figure 4 amortize setup work across many work instances,[c] but the characteristics of the amortization differ. To reach a break-even point, where the client does less work verifying than performing the work locally, Pepper and Ginger must batch work instances, whereas GGP and Pinocchio must perform enough instances to amortize key setup costs. These approaches have very different effects on latency. A client cannot benefit from Pepper or Ginger until it has accumulated an entire batch of instances. In Pinocchio, key setup can be precomputed, and henceforth every instance (including the first one) enjoys a better-than-break-even latency. Figure 4 shows the minimum latency achievable by each system. Compared with Ginger for a single instance, Pinocchio's verifier is ~120,000×–17,000,000× faster, and the worker is 19×–60× faster. To improve performance, Ginger's parameters are chosen such that the probability that the adversary can successfully cheat can be as high as $\frac{1}{2^{20}}$,(Figure 2 in Ref.[23]) while in Pinocchio, the probability is roughly $\frac{1}{2^{128}}$.

Finally, Pinocchio's verification is more efficient than Thaler's custom protocol, but Thaler's protocol is the only one to achieve practicality for the worker, showing the price the other systems pay for generality.

Figure 5. Cost of Verification versus Local. Verification must be cheaper than native execution for outsourcing to make sense, though for applications that want zero-knowledge, more expensive verification may be acceptable. All apps trend in the right direction, and three apps cross the plane where verification is cheaper than native. Error bars, often too small to see, represent 95% confidence intervals ($N = 50$, $\sigma \leq 2\%$).



5.2. End-to-end application performance

We measure Pinocchio's performance for the applications and parameter settings described in Section 4.3. All applications are written in C and compile to both QAPs and to native executables. We measure performance using 32-bit input values, so we can compare against the native C version. This obviously makes things more challenging for Pinocchio, since Pinocchio operates over a 254-bit field using multi-precision integers, whereas the local execution uses the CPU's native 32-bit operations.

Figure 5 plots Pinocchio's verification time against the time to execute the same app natively; each line represents a parameterized app, and each point represents a particular parameter setting. Our key finding is that, for sufficiently large parameters, three apps cross the line where outsourcing makes sense; that is, verifying the results of an outsourced computation is cheaper than local native execution. Note that the slope of each app's line is dictated by the size of the app parameters we experimented with (e.g., we reached larger parameters for fixed matrix than for two matrices).

On the downside, the other three apps, while trending in the right direction, fail to cross the outsourcing threshold. The difference is that these three apps perform large numbers of inequality comparisons and/or bitwise operations. This makes our circuit-based representation less efficient relative to native, and hence on our current experimental platform, we cannot push the application parameter settings to the point where they would beat local execution. Nonetheless, these applications may still be useful in settings that require Pinocchio's zero-knowledge proofs.

Fortunately, additional experiments show that enabling zero-knowledge proofs adds a negligible, fixed cost to key generation (213 μs), and re-randomizing a proof to make it zero-knowledge requires little effort (e.g., 300 ms or 0.1% for the multivariate polynomial app).

Figure 6 provides more details of Pinocchio's performance. For KeyGen, our experiments conservatively assume that the client does no precomputation in

[c] In contrast, Pinocchio's public verifier (not shown) enables a client to benefit from a third party's key setup work.

Figure 6. Application Performance. Pinocchio's performance for a sampling of the parameter settings (Section 4.3). All programs are compiled directly from C. The first two columns indicate the number of application inputs and outputs, and the number of gates in the corresponding arithmetic circuit. KeyGen is a one-time setup cost per application; Compute is the time the worker spends proving it computed correctly; Verify is the time the client spends checking the proof. Verification values in bold indicate verification is cheaper than computing the circuit locally; those with stars (*) indicate verification is cheaper than native execution. Public verification, while more expensive, allows anyone to check the results; private verification is faster, but allows anyone who can verify a proof to potentially generate a cheating proof. The Circuit column reports the time to evaluate the application's circuit representation, while Native indicates the time to run the application as a local, native executable. The last three columns indicate the size of the keys necessary to produce and verify proofs, as well as the size of the proof itself.

<table>
<tr><th></th><th></th><th>Mult</th><th>KeyGen</th><th>Compute</th><th>Verify</th><th>(ms)</th><th>Circuit</th><th>Native</th><th>EvalKey</th><th>VerKey</th><th>Proof</th></tr>
<tr><th></th><th>|IO|</th><th>gates</th><th>pub(s)</th><th>(s)</th><th>Pub</th><th>Priv</th><th>(ms)</th><th>(ms)</th><th>(MB)</th><th>(KB)</th><th>(B)</th></tr>
<tr><td>Fixed matrix, Medium</td><td>1201</td><td>600</td><td>0.7</td><td>0.4</td><td><b>39.5</b></td><td><b>10.0</b></td><td>123.7</td><td>4.3</td><td>0.3</td><td>37.9</td><td>288</td></tr>
<tr><td>Fixed matrix, Large</td><td>2001</td><td>1000</td><td>1.5</td><td>0.9</td><td><b>58.9</b></td><td><b>*10.1</b></td><td>337.4</td><td>12.4</td><td>0.5</td><td>62.9</td><td>288</td></tr>
<tr><td>Two matrices, Medium</td><td>14,701</td><td>347,900</td><td>79.8</td><td>269.4</td><td>340.7</td><td><b>12.1</b></td><td>124.9</td><td>4.0</td><td>97.9</td><td>459.8</td><td>288</td></tr>
<tr><td>Two matrices, Large</td><td>36,301</td><td>1,343,100</td><td>299.3</td><td>1127.8</td><td>882.2</td><td><b>*15.4</b></td><td>509.5</td><td>15.5</td><td>374.8</td><td>1134.8</td><td>288</td></tr>
<tr><td>MultiVar poly, Medium</td><td>7</td><td>203,428</td><td>41.9</td><td>246.1</td><td><b>11.6</b></td><td><b>10.0</b></td><td>93.1</td><td>4.5</td><td>55.9</td><td>0.6</td><td>288</td></tr>
<tr><td>MultiVar poly, Large</td><td>7</td><td>571,046</td><td>127.1</td><td>711.6</td><td><b>*12.7</b></td><td><b>*11.1</b></td><td>267.2</td><td>12.9</td><td>156.8</td><td>0.6</td><td>288</td></tr>
<tr><td>Image matching, Medium</td><td>13</td><td>86,345</td><td>26.4</td><td>41.1</td><td>11.1</td><td>9.9</td><td>5.5</td><td>0.1</td><td>23.6</td><td>0.8</td><td>288</td></tr>
<tr><td>Image matching, Large</td><td>13</td><td>277,745</td><td>67.0</td><td>144.4</td><td><b>11.4</b></td><td><b>10.1</b></td><td>18.0</td><td>0.4</td><td>75.8</td><td>0.8</td><td>288</td></tr>
<tr><td>Shortest paths, Medium</td><td>513</td><td>366,089</td><td>85.4</td><td>198.0</td><td>25.5</td><td><b>10.0</b></td><td>18.7</td><td>0.1</td><td>99.6</td><td>16.4</td><td>288</td></tr>
<tr><td>Shortest paths, Large</td><td>1153</td><td>1,400,493</td><td>317.5</td><td>850.2</td><td><b>48.9</b></td><td><b>10.8</b></td><td>69.5</td><td>0.3</td><td>381.4</td><td>36.4</td><td>288</td></tr>
<tr><td>Lattice gas sim, Medium</td><td>21</td><td>144,063</td><td>38.2</td><td>76.4</td><td><b>10.9</b></td><td><b>9.9</b></td><td>91.4</td><td>0.2</td><td>39.6</td><td>1.1</td><td>288</td></tr>
<tr><td>Lattice gas sim, Large</td><td>21</td><td>283,023</td><td>75.6</td><td>165.8</td><td><b>10.9</b></td><td><b>9.8</b></td><td>176.6</td><td>0.4</td><td>77.7</td><td>1.1</td><td>288</td></tr>
<tr><td>SHA-1</td><td>22</td><td>23,785</td><td>12.0</td><td>15.7</td><td><b>11.1</b></td><td><b>9.9</b></td><td>18.8</td><td>0.0</td><td>6.5</td><td>1.1</td><td>288</td></tr>
</table>

anticipation of outsourcing a function, and for Compute, we assume that the worker only does a single work instance before throwing away all of its state. As discussed in Section 4.2.1, in practice, we would take advantage of both precomputation and caching of previous work, which on average saves at least 43% of the effort for KeyGen and 16% of the effort for Compute.

In Figure 6, we see again that three apps (starred) beat native execution, including one in the public verifier setting (which requires more expensive operations per IO). The data also reinforces the point that using a circuit representation imposes a significant cost on image matching, shortest paths, and the lattice gas sim relative to native, suggesting a target for optimization. Relative to the circuit representation, Pinocchio's verification is cheap: both the public and the designated verifier "win" most of the time when compared to the circuit execution. Specifically, the designated verifier wins in 12 of 13 (92%) application settings. Public verification is more expensive, particularly for large IO, but still wins in 9 of 13 (69%) settings.

Since Pinocchio offers public verification, some clients will benefit from the KeyGen work of others, and hence only care about the verification costs. For example, a cellphone carrier might perform the one-time KeyGen so that its customers can verify computations done by arbitrary workers.

However, in other settings, for example, a company outsourcing work to the cloud, the key generator and verifier may be the same entity, and will wish to amortize the cost of key generation via the savings from verification. Figure 6 shows that most apps have a low "break even" point vs. circuit execution: the median for the designated verifier is 555 instances and for public verifier is 500 instances. Every instance afterwards is a net "win," even for the key generator.

Figure 6 holds more good news for Pinocchio: the keys it generates are reasonably sized, with the evaluation key (which describes the entire computation) typically requiring 10s or 100s of MB. The weak verifier's key (which grows linearly with the I/O) is typically only a few KB, and even at its largest, for two-matrix multiplication, it requires only slightly more than 1 MB. This suggests that the keys are quite portable and will not require excessive bandwidth to transmit.

Finally, from the client's perspective, if the worker's efforts are free, then the worker's additional overhead of generating a proof is irrelevant, as long as it doesn't hurt response latency. Our results, combined with prior work on parallelization,[23] suggest that latency can be brought down to reasonable levels. And indeed in high-assurance scenarios, scenarios where the client is incapable of performing the calculation itself (e.g., a power-limited device), or scenarios where the worker's resources are otherwise idle, the client may very well view the worker as "free."

However, in other scenarios, such as cloud computing, the worker's efforts are not free. Even here, however, Chen and Sion[6] estimate that the cost of cloud computing is about 60× cheaper than local computing for a small enterprise. This provides an approximate upper-bound for the amount of extra work we should be willing to add to the worker's overhead.

6. RELATED WORK

When implementing verified computation, prior efforts focused on either interactive proofs or PCPs. One effort[7, 25] builds on the interactive proofs of Goldwasser et al.[12] (GKR). They target a streaming setting where the client cannot store all of the data it wishes to compute over; the system currently requires the function computed to be highly parallelizable. On the plus side, it does not require cryptography, and it is secure against computationally unbounded adversaries.

Setty et al. produced a line of PCP-based systems called Pepper[22] and Ginger.[23] They build on a particular type of PCP called a linear PCP,[14] in which the proof can be represented as a linear function. This allows the worker to use a linearly homomorphic encryption scheme to create a commitment to its proof while relying only on standard cryptographic assumptions. Through a combination of theoretical and systems-level improvements, this work made tremendous progress in making PCP-based systems practical. Indeed, for applications that can tolerate large batch sizes, the amortized costs of verification can be quite low.

A few downsides remain, however. Because the work builds on the Hadamard PCP,[1] the setup time, network overhead, and the prover's work are quadratic in the size of the original computation, unless the protocol is hand-tailored. To achieve efficiency, the verifier cannot verify the results until a full batch returns. The scheme is designated verifier, meaning that third parties cannot verify the results of outsourced computations without sharing the client's secret key and risking fraud. The scheme also does not support zero-knowledge proofs.

Concurrent work[21] also builds on the quadratic programs of Gennaro et al.[10] They observe that QAPs can be viewed as linear PCPs and hence can fit into Ginger's cryptographic framework.[23] Their work shows worker computation improvements similar to those of Pinocchio. They retain PCPs and Ginger's cryptographic protocol, so they rely on simpler cryptographic assumptions than Pinocchio, but they must still batch computations to obtain an efficient verifier. They also remain designated verifier and do not support zero-knowledge proofs.

A subsequent line of work[3] expands application expressivity by combining Pinocchio's cryptographic protocol with an innovative encoding of RAM accesses. They also propose an elegant program encoding based on a general-purpose CPU, but this leads to overheads, for applications like matrix multiplication, of 5–7 orders of magnitude compared with Pinocchio.

Several systems provide compilers for zero-knowledge (ZK) proofs.[17] In general, these systems are likely to exhibit better performance than Pinocchio for their particular subset of functionality, but they do not possess the same level of efficient generality.

7. CONCLUSION

We have presented Pinocchio, a system for public verifiable computing. Pinocchio uses quadratic programs, a new method for encoding computation, combined with a highly efficient cryptographic protocol to achieve both asymptotic and concrete efficiency. Pinocchio produces 288-byte proofs, regardless of the size of the computation, and the proofs can be verified rapidly, typically in tens of milliseconds, beating native execution in several cases. This represents five to seven *orders of magnitude* performance improvement over prior work.[23] The worker also produces the proof 19×–60× faster. Pinocchio even slashes the cost of its underlying protocol, cutting the cost of both key and proof generation by more than 60%. The end result is a cryptographic protocol for efficiently signing computations. Combined with a compiler for real C programs, Pinocchio brings VC much closer to practicality.

References

1. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M. Proof verification and the hardness of approximation problems. *J. ACM 45*, 3 (1998).
2. Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., Virza, M. Zerocash: Decentralized anonymous payments from Bitcoin. In *Proceedings of the IEEE Symposium on Security and Privacy* (2014).
3. Ben-Sasson, E., Chiesa, A., Tromer, E., Virza, M. Succinct non-interactive zero knowledge for a von Neumann architecture. In *Proceedings of USENIX Security* (2014).
4. Braun, B., Feldman, A.J., Ren, Z., Setty, S., Blumberg, A.J., Walfish, M. Verifying computations with state. In *Proceedings of the ACM SOSP* (2013).
5. Castro, M., Liskov, B. Practical Byzantine fault tolerance and proactive recovery. *ACM Trans. Comp. Syst. 20*, 4 (2002).
6. Chen, Y., Sion, R. To cloud or not to cloud? Musings on costs and viability. In *Proceedings of the ACM Symposium on Cloud Computing* (2011).
7. Cormode, G., Mitzenmacher, M., Thaler, J. Practical verified computation with streaming interactive proofs. In *ITCS* (2012).
8. Danezis, G., Fournet, C., Kohlweiss, M., Parno, B. Pinocchio coin: Building Zerocoin from a succinct pairing-based proof system. In *ACM Workshop on Language Support for Privacy Enhancing Technologies* (2013).
9. Gennaro, R., Gentry, C., Parno. B. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *Proceedings of IACR CRYPTO* (2010).
10. Gennaro, R., Gentry, C., Parno, B., Raykova, M. Quadratic span programs and succinct NIZKs without PCPs. In *EUROCRYPT* (2013). Originally published as Cryptology ePrint Archive, Report 2012/215.
11. Gentry, C. A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009).
12. Goldwasser, S., Kalai, Y.T., Rothblum, G.N. Delegating computation: Interactive proofs for muggles. In *STOC* (2008).
13. Golle, P., Mironov, I. Uncheatable distributed computations. In *Proceedings of CT-RSA* (2001).
14. Ishai, Y., Kushilevitz, E., Ostrovsky, R. Efficient arguments without short PCPs. In *IEEE Conference on Computational Complexity* (2007).
15. Kilian, J. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *STOC* (1992).
16. Lee, R.B., Kwan, P., McGregor, J.P., Dwoskin, J., Wang, Z. Architecture for protecting critical secrets in microprocessors. In *Proceedings of the International Symposium on Computer Architecture* (*ISCA*) (2005).
17. Meiklejohn, S., Erway, C.C., Küpçü, A., Hinkle, T., Lysyanskaya, A. ZKPDL: A language-based system for efficient zero-knowledge proofs and electronic cash. In *Proceedings of USENIX Security* (2010).
18. Naehrig, M., Niederhagen, R., Schwabe, P. New software speed records for cryptographic pairings. In *Proceedings of LATINCRYPT* (2010).
19. Parno, B., McCune, J.M., Perrig, A. *Bootstrapping Trust in Modern Computers*. Springer, New York/Dordrecht/Heidelberg/London, 2011. DOI: 10.1007/978-1-4614-1460-5.
20. Parno, B., Raykova, M., Vaikuntanathan, V. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In *IACR Theory of Cryptography Conference* (*TCC*) (2012).
21. Setty, S., Braun, B., Vu, V., Blumberg, A.J., Parno, B., Walfish, M. Resolving the conflict between generality and plausibility in verified computation. In *Proceedings of the ACM European Conference on Computer Systems* (*EuroSys*) (Apr. 2013).
22. Setty, S., McPherson, R., Blumberg, A.J., Walfish, M. Making argument systems for outsourced computation practical (sometimes). In *Proceedings of the ISOC NDSS* (2012).
23. Setty, S., Vu, V., Panpalia, N., Braun, B., Blumberg, A.J., Walfish, M. Taking proof-based verified computation a few steps closer to practicality. In *Proceedings of USENIX Security* (2012).
24. Sion, R. Query execution assurance for outsourced databases. In *The Very Large Databases Conference* (*VLDB*) (2005).
25. Thaler, J. Time-optimal interactive proofs for circuit evaluation. In *Proceedings of CRYPTO* (2013).

Bryan Parno and Jon Howell ({parno, howell}@microsoft.com), Microsoft Research.

Craig Gentry (cbgentry@us.ibm.com), IBM Research.

Mariana Raykova (mariana@cs.columbia.edu), SRI International.

Watch the authors discuss their work in this exclusive *Communications* video. http://cacm.acm.org/videos/pinocchio-nearly-practical-verifiable-computation

DOI:10.1145/2863699

To view the accompanying paper, visit doi.acm.org/10.1145/2863701 rh

Technical Perspective
Program Synthesis Using Stochastic Techniques

By Sumit Gulwani

PROGRAM SYNTHESIS INVOLVES discovering a program from an underlying space of programs that satisfies a given specification using some search technique.[3] It has many applications including algorithm discovery, optimized implementations, programming assistance,[5] and synthesis of small scripts to automate repetitive tasks for end users.[4] Its success relies heavily on efficient search algorithms to navigate the underlying huge state space of programs. The authors of the following paper have developed a stochastic search technique and applied it to program optimization. The impressive results of their implementation STOKE on hard program optimization benchmarks illustrate the promising potential of stochastic search to hard program synthesis problems.

The specification for program synthesis can be in the form of a logical declarative relationship between inputs and outputs. Examples or demonstration traces are a popular specification choice for end-user programming.[4] In program optimization, when viewed as a synthesis problem, the specification consists of inefficient programs that need to be translated into functionally equivalent but more efficient programs.

Multiple solutions may satisfy the Boolean constraints in the specification. In such cases, preferences can be specified using an optimization function. In programming-by-examples, where the number of solutions may be several powers of 10, ranking functions over program features are used to guess an intended program.[4] In program optimization, the goal is to prefer programs with smaller runtimes. STOKE's use of sum of average latencies of the involved instructions serves as a good static approximation to the intended measure.

The search space in program synthesis requires a trade-off: expressive enough to describe programs of interest, while restricted enough to allow efficient synthesis. Various domain-specific languages have been designed for synthesis purposes[1,3] that meet this trade-off. In program optimization, a common choice is loop-free instruction sequences of bounded length. While prior techniques restrict the space to 10–15 opcodes or require specifying a small set of relevant opcodes for a given problem instance, STOKE significantly advances the state of the art by allowing nearly 400 x86-64 opcodes.

Search Technique

A simple search strategy is to enumerate programs in the underlying space in order of increasing size. However, this does not scale to huge search spaces of the kind considered by STOKE. Another strategy is to reduce the (second-order) search problem to (first-order) constraint solving[3] and leverage off-the-shelf SAT/SMT solvers like Z3.[2] This allows building over huge engineering advances made in SAT/SMT solving, but does not allow effectively incorporating optimization constraints. Version-space algebra-based techniques[4] incorporate preferences by computing the set of all/many solutions in a first phase, and then selecting the highest-ranked solution in a second phase. STOKE also leverages a two-phased approach. Its first phase finds algorithmically distinct solutions, while the second phase finds efficient implementations of code sequences discovered by the first phase.

Stochastic search. STOKE uses stochastic search for each of its two phases. This includes an appropriately defined cost metric, and MCMC sampling to select a next candidate. The first-phase cost metric is based on functional equivalence to the target input sequence (a Boolean constraint). The second-phase cost metric combines functional equivalence measure with a performance metric (an optimization constraint). In order to define a smooth cost metric over Boolean program equivalent constraints, STOKE uses two clever heuristics: use of Hamming distance to measure closeness of generated bit-values to the target on a representative test input set, and rewarding generation of (almost) correct values in incorrect locations.

Interdisciplinary inspiration. STOKE combines techniques from software engineering, programming languages, and numerical optimization. It uses test input generation (Intel's PinTool) for generating representative test inputs for evaluating equivalence cost metrics during MCMC sampling. It uses automated theorem proving (Microsoft's Z3) for verifying equivalence of the synthesized sequence in a post-processing step. Recent extensions that search over loopy program spaces leverage invariant inference techniques for verifying equivalence. STOKE is a great exercise in interdisciplinary inspiration for efficient search algorithms for hard synthesis problems. This is timely and significant, given recent renewed interest and promising developments in the area of program synthesis across various communities. C

References

1. Alur, R. et al. Syntax-guided synthesis. In *Proceedings of 2013 FMCAD*.
2. Bjørner, N. Taking satisfiability to next level with Z3. In *Proceedings of 2012 IJCAR*.
3. Gulwani, S. Dimensions in program synthesis. In *Proceedings of 2010 PPDP*.
4. Gulwani, A., Harris, W., and Singh, R. Spreadsheet data manipulation using examples. *Commun. ACM*, (2012).
5. Solar-Lezama, A. Program Synthesis by Sketching. Ph.D. thesis, UC Berkeley, 2008.

Sumit Gulwani (sumitg@microsoft.com) is research manager and principal researcher at Microsoft Corp., Redmond, WA.

DOI:10.1145/2863701

Stochastic Program Optimization

By Eric Schkufza, Rahul Sharma, and Alex Aiken

Abstract

The optimization of short sequences of loop-free, fixed-point assembly code sequences is an important problem in high-performance computing. However, the competing constraints of transformation correctness and performance improvement often force even special purpose compilers to produce sub-optimal code. We show that by encoding these constraints as terms in a cost function, and using a Markov Chain Monte Carlo sampler to rapidly explore the space of all possible code sequences, we are able to generate aggressively optimized versions of a given target code sequence. Beginning from binaries compiled by `llvm -O0`, we are able to produce provably correct code sequences that either match or outperform the code produced by `gcc -O3`, `icc -O3`, and in some cases expert handwritten assembly.

1. INTRODUCTION

For many application domains there is considerable value in producing the most performant code possible. However, the traditional structure of a compiler's optimization phase is ill-suited to this task. Factoring the optimization problem into a collection of small subproblems that can be solved independently—although suitable for generating consistently good code—can lead to sub-optimal results. In many cases, the best possible code can only be obtained through the simultaneous consideration of mutually dependent issues such as instruction selection, register allocation, and target-dependent optimization.

Previous approaches to this problem have focused on the exploration of all possibilities within some limited class of code sequences. In contrast to a traditional compiler, which uses performance constraints to drive the generation of a single sequence, these systems consider multiple sequences and select the one that is best able to satisfy those constraints. An attractive feature of this approach is completeness: if a code sequence exists that meets the desired constraints it is guaranteed to be found. However, completeness also places practical limitations on the type of code that can be considered. These techniques are either limited to sequences that are shorter than the threshold at which many interesting optimizations take place or code written in simplified languages.

We overcome this limitation through the use of incomplete search: the competing requirements of correctness and performance improvement are encoded as terms in a cost function which is defined over the space of all loop-free x86_64 instruction sequences, and the optimization task is formulated as a cost minimization problem. While the search space is highly irregular and not amenable to exact optimization techniques, we show that the common approach of employing a Markov Chain Monte Carlo (MCMC) sampler to explore the function and produce low-cost samples is sufficient for producing high-quality code.

Although the technique sacrifices completeness it produces dramatic increases in the quality of the resulting code. Figure 1 shows two versions of the Montgomery multiplication kernel used by the OpenSSL RSA encryption library. Beginning from code compiled by `llvm -O0` (116 lines, not shown), our prototype stochastic optimizer STOKE produces code (right) that is 16 lines shorter and 1.6 times faster than the code produced by `gcc -O3` (left), and even slightly faster than the expert handwritten assembly included in the OpenSSL repository.

2. RELATED WORK

Although techniques that preserve completeness are effective within certain domains, their general applicability remains limited. The shortcomings are best highlighted in the context of the code sequence shown in Figure 1.

The code does the following: two 32-bit values, `ecx` and `edx`, are concatenated and multiplied by the 64-bit value `rsi` to produce a 128-bit product. The 64-bit values in `rdi` and `r8` are added to that product, and the result is stored in `r8` and `rdi`. The version produced by `gcc -O3` (left) implements the 128-bit multiplication as four 64-bit multiplications and a summation of the results. In contrast, the version produced by STOKE (right), uses a hardware intrinsic which requires that the inputs be permuted and moved to distinguished register locations so that the multiplication may be performed in a single step. The odd looking move on line 4 (right) produces the non-obvious but necessary side effect of zeroing the upper 32 bits of `rdx`.

Massalin's superoptimizer[12] explicitly enumerates sequences of code of increasing length and selects the first that behaves identically to the input sequence on a set of test cases. Massalin reports optimizing instruction sequences of up to length 12, but only after restricting the set of enumerable opcodes to between 10 and 15. In contrast, STOKE uses a large subset of the nearly 400 x86_64 opcodes, some with over 20 variations, to produce the 11 instruction kernel shown in Figure 1. It is unlikely that Massalin's approach would scale to an instruction set of this size.

Denali[9] and Equality Saturation,[17] achieve improved scalability by only considering code sequences that are equivalent to the input sequence; candidates are explored through successive application of equality preserving transformations. Because both techniques are goal-directed, they dramatically improve the number of primitive instructions and the length of sequences that can be considered in practice. However, both also rely heavily on expert knowledge. It is unclear whether an expert would know to encode

This work was supported by NSF grant CCF-0915766 and the Army High Performance Computing Research Center.

Figure 1. Montgomery multiplication kernel from the OpenSSL RSA library. Compilations shown for `gcc -O3` (left) and a stochastic optimizer (right).

```
[r8:rdi] = rsi * [ecx:edx] + r8 + rdi
```

```
# gcc -O3

movq rsi, r9
movl ecx, ecx
shrq 32, rsi
andl 0xffffffff, r9d
movq rcx, rax
movl edx, edx
imulq r9, rax
imulq rdx, r9
imulq rsi, rdx
imulq rsi, rcx
addq rdx, rax
jae .L0
movabsq 0x100000000, rdx
addq rdx, rcx
.L0:
movq rax, rsi
movq rax, rdx
shrq 32, rsi
salq 32, rdx
addq rsi, rcx
addq r9, rdx
adcq 0, rcx
addq r8, rdx
adcq 0, rcx
addq rdi, rdx
adcq 0, rcx
movq rcx, r8
movq rdx, rdi
```

```
# STOKE

shlq 32, rcx
movl edx, edx
xorq rdx, rcx
movq rcx, rax
mulq rsi
addq r8, rdi
adcq 0, rdx
addq rdi, rax
adcq 0, rdx
movq rdx, r8
movq rax, rdi
```

the multiplication transformation shown in Figure 1, or whether a set of expert rules could ever cover the set of all possible interesting optimizations.

Bansal's peephole superoptimizer[2] automatically enumerates 32-bit x86 optimizations and stores the results in a database for later use. By exploiting symmetries between code sequences that are equivalent up to register renaming, Bansal was able to scale this approach to optimizations mapping code sequences of up to length 6 to sequences of up to length 3. The approach has the dual benefit of hiding the high cost of superoptimization by performing search once-and-for-all offline and eliminating the dependence on expert knowledge. To an extent, the use of a database also allows the system to overcome the low upper bound on instruction length through the repeated application of the optimizer along a sliding code window. Regardless, the kernel shown in Figure 1 has a property shared by many real world code sequences that no sequence of local optimizations will transform the code produced by `gcc -O3` into the code produced by STOKE.

Finally, Sketching[16] and Brahma[7] address the closely related component-based sequence synthesis problem. These systems rely on either a declarative specification, or a user-specified partial sequence, and operate on statements in simplified bit-vector calculi rather than directly on hardware instructions. Liang et al.[10] considers the task of learning code sequences from test cases alone, but at a similarly high level of abstraction. Although useful for synthesizing non-trivial code, the internal representations used by these systems preclude them from reasoning about the low-level performance properties of the code that they produce.

3. COST MINIMIZATION

Before describing our approach to x86_64 optimization, we discuss optimization as cost minimization in the abstract. To begin, we define a cost function with terms that balance the competing constraints of *transformation correctness* (eq(·)), and *performance improvement* (perf(·)). We refer to an input sequence as the *target* ($\mathcal{T}$) and a candidate compilation as a *rewrite* ($\mathcal{R}$), and say that a function $f(X; Y)$ takes inputs X and is defined in terms of Y

$$\mathrm{cost}(\mathcal{R};\, \mathcal{T}) = w_{\mathrm{e}} \cdot \mathrm{eq}(\mathcal{R};\, \mathcal{T}) + w_{\mathrm{p}} \cdot \mathrm{perf}(\mathcal{R}) \qquad (1)$$

The eq(·) term measures the similarity of two code sequences and returns zero if and only if they are equal. For our purposes, code sequences are regarded as functions of registers and memory contents and we say they are equal if for all machine states that agree on the live inputs defined by the target, the two code sequences produce identical side effects on the live outputs defined by the target. Because optimization is undefined for ill-formed code sequences, it is unnecessary that eq(·) be defined for a target or rewrite that produce exceptional behavior. However nothing prevents us from doing so, and it would be straightforward to define eq(·) to preserve exceptional behavior as well.

In contrast, the perf(·) term quantifies the performance improvement of a rewrite; smaller values represent larger improvements. Crucially, the extent to which this term is accurate directly affects the quality of the resulting code.

Using this notation, the connection to optimization is straightforward. We say that an optimization is any of the set of rewrites for which the perf(·) term is improved, and the eq(·) term is zero

$$\{\, r \mid \mathrm{perf}(r) \le \mathrm{perf}(\mathcal{T}) \wedge \mathbf{eq}(r;\, \mathcal{T}) = \mathbf{0} \,\} \qquad (2)$$

Discovering these optimizations requires the use of a cost minimization procedure. However, in general we expect cost functions of this form to be highly irregular and not amenable to exact optimization techniques. The solution to this problem is to employ the common strategy of using an MCMC sampler. Although a complete discussion of the technique is beyond the scope of this article, we summarize the main ideas here.

MCMC sampling is a technique for drawing elements from a probability density function in direct proportion to its value: regions of higher probability are sampled more often than regions of low probability. When applied to cost minimization, it has two attractive properties. In the limit, the most samples will be taken from the minimum (optimal) values of a function. And in practice, well before that behavior is observed, it functions as an intelligent hill climbing method which is robust against irregular functions that are dense with local minima.

A common method[6] for transforming an arbitrary function such as cost(·) into a probability density function is shown below, where β is a constant and Z is a partition function that normalizes the resulting distribution: Although computing Z is generally intractable, the Metropolis–Hastings algorithm is designed to explore density functions such as $p(\cdot)$ without having to compute Z directly[8]

$$p(\mathcal{R};\mathcal{T}) = \frac{1}{Z}\exp\left(-\beta\cdot\mathrm{cost}(\mathcal{R};\mathcal{T})\right) \tag{3}$$

The basic idea is simple. The algorithm maintains a current rewrite $\mathcal{R}$ and proposes a modified rewrite $\mathcal{R}^*$. The *proposal* $\mathcal{R}^*$ is then either accepted or rejected. If it is accepted, $\mathcal{R}^*$ becomes the current rewrite. Otherwise another proposal based on $\mathcal{R}$ is generated. The algorithm iterates until its computational budget is exhausted, and as long as the proposals are *ergodic* (sufficient to transform any code sequence into any other through some sequence of applications) the algorithm will in the limit produce a sequence of samples distributed in proportion to their cost.

This global property depends on the local acceptance criteria for a proposal $\mathcal{R} \to \mathcal{R}^*$, which is governed by the Metropolis–Hastings acceptance probability shown below. We use the notation $q(\mathcal{R}^*|\mathcal{R})$ to represent the proposal distribution from which a new rewrite $\mathcal{R}^*$ is sampled given the current rewrite, $\mathcal{R}$. This distribution is key to a successful application of the algorithm. Empirically, the best results are obtained for a distribution which makes both local proposals that make minor modifications to $\mathcal{R}$ and global proposals that induce major changes

$$\alpha(\mathcal{R}\to\mathcal{R}^*;\mathcal{T}) = \min\left(1, \frac{p(\mathcal{R}^*;\mathcal{T})\cdot q(\mathcal{R}|\mathcal{R}^*)}{p(\mathcal{R};\mathcal{T})\cdot q(\mathcal{R}^*|\mathcal{R})}\right) \tag{4}$$

The important properties of the acceptance criteria are the following: If $\mathcal{R}^*$ is better (has a higher probability/lower cost) than $\mathcal{R}$, the proposal is always accepted. If $\mathcal{R}^*$ is worse (has a lower probability/higher cost) than $\mathcal{R}$, the proposal may still be accepted with a probability that decreases as a function of the ratio between $\mathcal{R}^*$ and $\mathcal{R}$. This property prevents search from becoming trapped in local minima while remaining less likely to accept a move that is much worse than available alternatives. In the event that the proposal distribution is *symmetric*, $q(\mathcal{R}^*|\mathcal{R}) = q(\mathcal{R}|\mathcal{R}^*)$, the acceptance probability can be reduced to the much simpler Metropolis ratio, which is computed directly from cost($\cdot$):

$$\begin{aligned}\alpha(\mathcal{R}\to\mathcal{R}^*;\mathcal{T}) &= \min\left(1, \frac{p(\mathcal{R}^*;\mathcal{T})}{p(\mathcal{R};\mathcal{T})}\right)\\ &= \min\left(1, \exp(k)\right)\\ \text{where } k &= -\beta\cdot(\mathrm{cost}(\mathcal{R}^*;\mathcal{T}) - \mathrm{cost}(\mathcal{R};\mathcal{T}))\end{aligned} \tag{5}$$

4. X86_64 OPTIMIZATION

We now address the practical details of applying cost minimization to x86_64 optimization. As x86_64 is arguably the most complex instance of a CISC architecture, we expect this discussion to generalize well to other architectures. A natural choice for the implementation of the eq($\cdot$) term is the use of a *symbolic validator* (val($\cdot$)),[4] and a *binary indicator function* ($\mathbf{1}(\cdot)$), which returns one if its argument is true, and zero otherwise

$$\mathrm{eq}(\mathcal{R};\mathcal{T}) = 1 - \mathbf{1}(\mathrm{val}(\mathcal{T},\mathcal{R})) \tag{6}$$

However, the total number of invocations that can be performed per second using current symbolic validator technology is quite low. For even modestly sized code sequences, it is well below 1000. Because MCMC sampling is effective only insofar as it is able to explore sufficiently large numbers of proposals, the repeated computation of Equation (6) would drive that number well below a useful threshold.

This observation motivates the definition of an approximation to the eq($\cdot$) term which is based on *test cases* (τ). Intuitively, we execute the proposal $\mathcal{R}^*$ on a set of inputs and measure "how close" the output matches the target for those same inputs by counting the number of bits that differ between live outputs (i.e., the Hamming distance). In addition to being much faster than using a theorem prover, this approximation of equivalence has the added advantage of producing a smoother landscape than the 0/1 output of a symbolic equality test; it provides a useful notion of "almost correct" that can help to guide search

$$\begin{aligned}\mathrm{eq}'(\mathcal{R};\mathcal{T},\tau) = &\sum_{t\in\tau}\mathrm{reg}(\mathcal{R};\mathcal{T},t) + \mathrm{mem}(\mathcal{R};\mathcal{T},t)\\ &+\sum_{t\in\tau}\mathrm{err}(\mathcal{R};\mathcal{T},t)\end{aligned} \tag{7}$$

In Equation (7), reg($\cdot$) is used to compare the *side effects* (eval($\cdot$)) that both functions produce on the *live register outputs* (ρ) defined by the target. These outputs can include general purpose, SSE, and condition registers. reg($\cdot$) computes the number of bits that the results differ by using the *population count function* (pop($\cdot$)), which returns the number of 1-bits in the 64-bit representation of an integer

$$\mathrm{reg}(\mathcal{R};\mathcal{T},t) = \sum_{r\in\rho}\mathrm{pop}(\mathrm{eval}(\mathcal{T},r)\oplus\mathrm{eval}(\mathcal{R},r)) \tag{8}$$

For brevity, we omit the definition of mem($\cdot$), which is analogous. The remaining term, err($\cdot$), is used to distinguish code sequences that exhibit undefined behavior, by counting and then penalizing the number of segfaults, floating-point exceptions, and reads from undefined memory or registers, that occur during execution of a rewrite. We note that sigsegv($\cdot$) is defined in terms of $\mathcal{T}$, which determines the set of addresses that may be successfully dereferenced by a rewrite for a particular test case. Rewrites must be run in a sandbox to ensure that this behavior can be detected safely at runtime. The extension to additional types of exceptions is straightforward

$$\begin{aligned}\mathrm{err}(\mathcal{R};\mathcal{T},t) = &\, w_{ss}\cdot\mathrm{sigsegv}(\mathcal{R};\mathcal{T},t)\\ &+ w_{sf}\cdot\mathrm{sigfpe}(\mathcal{R};t)\\ &+ w_{ur}\cdot\mathrm{undef}(\mathcal{R};t)\end{aligned} \tag{9}$$

The evaluation of eq′($\cdot$) may be implemented either by JIT compilation, or the use of a hardware emulator. In our experiments (Section 8) we have chosen the former, and shown the ability to dispatch test case evaluations at a rate of between 1 and 10 million per second. Using this implementation, we define an optimized method for computing eq(·), which achieves sufficient throughput to be useful for MCMC sampling

$$\mathrm{eq}^*\,(\mathcal{R};\mathcal{T},\tau)=\begin{cases}\mathrm{eq}(\mathcal{R};\mathcal{T}), & \text{if } \mathrm{eq}'(\mathcal{R};\mathcal{T},\tau)=0\\ \mathrm{eq}'(\mathcal{R};\mathcal{T},\tau), & \text{otherwise}\end{cases}\tag{10}$$

Besides improved performance, Equation (10) has two desirable properties. First, failed computations of eq(·) will produce a counterexample test case[4] that can be used to refine τ. Although doing so changes the search space defined by cost(·), in practice the number of failed validations that are required to produce a robust set of test cases that accurately predict correctness is quite low. Second, as remarked above, it improves the search space by smoothly interpolating between correct and incorrect code sequences.

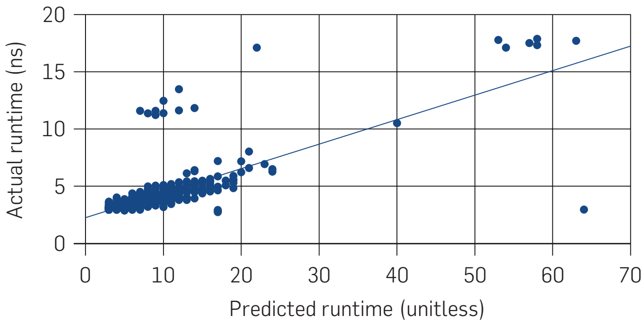
Similar considerations apply to the implementation of the perf(·) term. Although it seems natural to JIT compile both target and rewrite and compare runtimes, the amount of time required to execute a code sequence sufficiently many times to eliminate transient effects is prohibitively expensive. To account for this, we define a simple heuristic for approximating runtime which is based on a static approximation of the *average latencies* (lat(·)), of instructions

$$\mathrm{perf}(\mathcal{R})=\sum_{i\in \mathrm{inst}(\mathcal{R})}\mathrm{lat}(i)\tag{11}$$

Figure 2 shows a reasonable correlation between this heuristic and actual runtimes for a representative corpus of code sequences. Outliers are characterized either by disproportionately high instruction level parallelism at the micro-op level or inconsistent memory access times. A more accurate model of the higher order performance effects introduced by a modern CISC processor is feasible if tedious to construct and would likely be necessary for more complex code sequences.

Regardless, this approximation is sufficient for the benchmarks that we consider (Section 8). Errors that result from this approach are addressed by recomputing perf(·) using the JIT compilation method as a postprocessing step. During search, we record the n lowest cost programs produced by MCMC sampling, rerank each based on their actual runtimes, and return the best result.

Figure 2. Predicted versus observed runtimes for selected code sequences. Outliers are characterized by instruction level parallelism and memory effects.



Finally, there is the implementation of MCMC sampling for x86_64 optimization. Rewrites are represented as loop-free sequences of instructions of length ℓ, where a distinguished token (UNUSED) is used to represent unused instruction slots. This simplification to sequences of bounded length is crucial, as it places a constant value on the dimensionality of the resulting search space.[1] The proposal distribution $q(\cdot)$ chooses from four possible moves: the first two minor and the last two major:

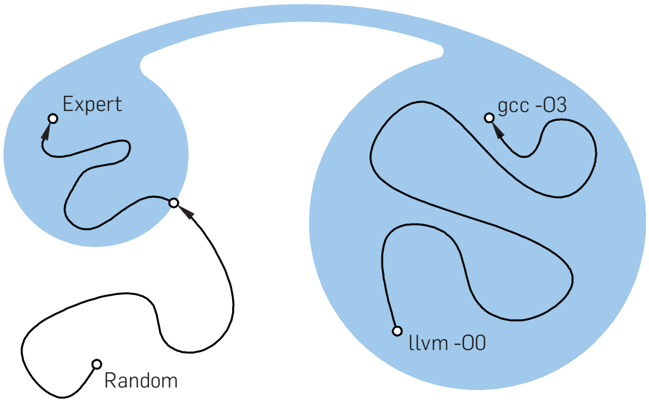
- **Opcode.** An instruction is randomly selected, and its opcode is replaced by a random opcode.
- **Operand.** An instruction is randomly selected and one of its operands is replaced by a random operand.
- **Swap.** Two lines of code are randomly selected and interchanged.
- **Instruction.** An instruction is randomly selected and replaced either by a random instruction or the UNUSED token. Proposing UNUSED corresponds to deleting an instruction, and replacing UNUSED by an instruction corresponds to inserting an instruction.

These moves satisfy the ergodicity property described in Section 3: any code sequence can be transformed into any other through repeated application of instruction moves. These moves also satisfy the symmetry property, and allow the use of Equation (5). To see why, note that the probabilities of performing all four moves are equal to the probabilities of undoing the transformations they produce using a move of the same type: opcode and operand moves are constrained to sample from identical equivalence classes before and after acceptance, and swap and instruction moves are unconstrained in either direction.

5. SEARCH STRATEGIES

An early version of the implementation described above was able to transform `llvm -O0` code into the equivalent of `gcc -O3` code, but was unable to produce results

Figure 3. Search space for the Montgomery multiplication benchmark: `O0` and `O3` codes are densely connected, whereas expert code is reachable only by an extremely low probability path.



that were competitive with expert hand-written code. The reason is suggested by Figure 3, which abstractly depicts the search space for the Montgomery multiplication benchmark shown in Figure 1. For loop-free code sequences, `llvm -O0` and `gcc -O3` differ primarily in stack use and instruction selection, but otherwise produce algorithmically similar results. Compilers are generally designed to compose many small local transformations: dead code elimination deletes an instruction, constant propagation changes a register to an immediate, and strength reduction replaces a multiplication by an add. Sequences of local optimizations such as these correspond to regions of equivalent code sequences that are densely connected by very short sequences of moves (often just one) and easily traversed by local search methods. Beginning from `llvm -O0` code, MCMC sampling will quickly improve local inefficiencies one by one and hill climb its way to the equivalent of `gcc -O3` code.

The code discovered by STOKE occupies an entirely different region of the search space. As remarked earlier, it represents a completely distinct algorithm for implementing the kernel at the assembly level. The only method for a local search procedure to produce code of this form from compiler generated code is to traverse the extremely low probability path that builds the code in place next to the original (all the while increasing its cost) only to delete the original code at the very end.

Although MCMC sampling is guaranteed to traverse this path in the limit, the likelihood of it doing so in any reasonable amount of time is so low as to be useless in practice. This observation motivates the division of cost minimization into two phases:

- A **synthesis** phase focused solely on correctness, which attempts to locate regions of equivalent code sequences that are distinct from the region occupied by the target.
- An **optimization** phase focused on performance, which searches for the fastest sequence within each of those regions.

These phases share the same implementation and differ only in starting point and acceptance function. Synthesis begins with a random code sequence, while optimization begins from a code sequence that is known to be equivalent to the target. Synthesis ignores the perf(·) term and uses Equation (10) as its cost function, whereas optimization uses both terms, which allows it to improve performance while also experimenting with "shortcuts" that (temporarily) violate correctness.

It is perhaps unintuitive that synthesis should be able to produce a correct rewrite from such an enormous search space in a tractable amount of time. In our experience, synthesis is effective precisely when it is possible to discover portions of a correct rewrite incrementally, rather than all at once. Figure 4 compares cost over time against the percentage of instructions that appear in the final rewrite for the Montgomery multiplication benchmark. As synthesis proceeds, the percentage of correct code increases in inverse proportion to the value of the cost function.

While this is encouraging and there are many code sequences that can be synthesized in pieces, there are many that cannot. Fortunately, even when synthesis fails, optimization is still possible. It must simply proceed only from the region occupied by the target as a starting point.

6. SEARCH OPTIMIZATIONS

Equation (10) is sufficiently fast for MCMC sampling, however its performance can be further improved. As described above, the eq* (·) term is computed by executing a proposal on test cases, noting the ratio in total cost with the current rewrite, and then sampling a random variable to decide whether to accept the proposal. However, by first sampling p, and then computing the maximum ratio that the algorithm will accept for that value, it is possible to terminate the evaluation of test cases as soon as that bound is exceeded and the proposal is guaranteed to be rejected

$$\begin{aligned} p &< \alpha(\mathcal{R} \to \mathcal{R}^*; \mathcal{T}) \\ &< \min(1, \exp(k)) \\ \text{where } k &= -\beta \cdot (\text{cost}(\mathcal{R}^*; \mathcal{T}) - \text{cost}(\mathcal{R}; \mathcal{T})) \qquad (12) \\ \text{eq}^*(\mathcal{R}^*; \mathcal{T}, \tau) &\leq \text{cost}(\mathcal{R}; \mathcal{T}, \tau) - \text{perf}(\mathcal{R}^*) - \frac{\log(p)}{\beta} \end{aligned}$$

Figure 4. Cost over time versus percentage of instructions that appear in the final zero-cost rewrite for the Montgomery multiplication synthesis benchmark.

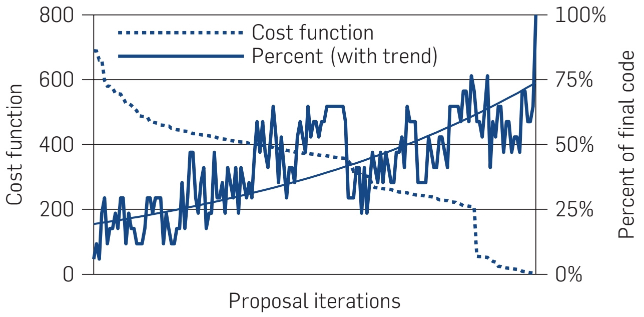


Figure 5. Proposals evaluated per second versus test cases evaluated prior to early termination, for the Montgomery multiplication synthesis benchmark. Cost function shown for reference.

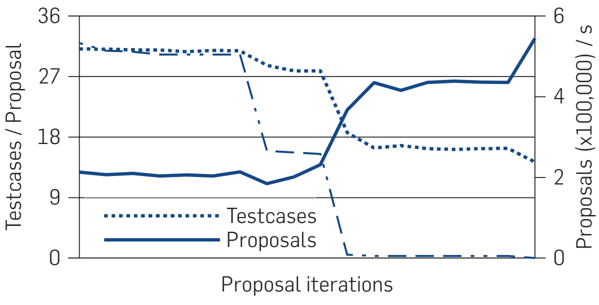


Figure 5 shows the result of applying this optimization during synthesis for the Montgomery multiplication benchmark. As the value of the cost function decreases, so too does the average number of test cases that must be evaluated prior to early termination. This produces a considerable increase in the number of proposals evaluated per second, which at peak falls between 100,000 and 1 million.

A more important improvement stems from the observation that the definition of reg(·) in Equation (8) is unnecessarily strict. An example is shown in Figure 6 for a target in which register al is live out. A rewrite that produces the inverse of the desired value in al is assigned the maximum possible cost in spite of the fact that it produces the correct value, only in the incorrect location: dl. We can improve this term by rewarding rewrites that produce correct (or nearly correct) values in incorrect locations. The relaxed definition shown below examines all registers of equivalent *bit-width* (bw(·)), selects the one that most closely matches the value of the target register, and assigns a small *misalignment penalty* (w_m) if the selected register differs from the original. Using this definition, the rewrite is assigned a cost of just w_m

$$\mathrm{reg}'(\mathcal{R};\mathcal{T},\tau)=\sum_{r\in\rho}\min_{r'\in\mathrm{bw}(r)}\mathrm{R}(r,r';\tau)$$
$$\mathrm{R}(r,r';\tau)=\mathrm{pop}(\mathrm{eval}(\mathcal{T},r)\oplus\mathrm{val}(\mathcal{R},r'))+w_m\cdot\mathbf{1}(r\neq r') \qquad (13)$$

Although it is possible to relax the definition of memory equality analogously, the time required to compute this term grows quadratically with the size of the target's memory footprint. Although this approach suffices for our experiments, a more efficient implementation is necessary for more complex code sequences.

Figure 7 shows the result of using these improved definitions during synthesis for the Montgomery multiplication benchmark. In the amount of time required for the relaxed cost function to converge, the original strict version obtains a minimum cost that is only slightly superior to a purely random search. The dramatic improvement can be explained as an implicit parallelization of the search procedure. Accepting correct values in arbitrary locations allows a rewrite to simultaneously explore as many alternate computations as can fit within a code sequence of length ℓ.

7. STOKE

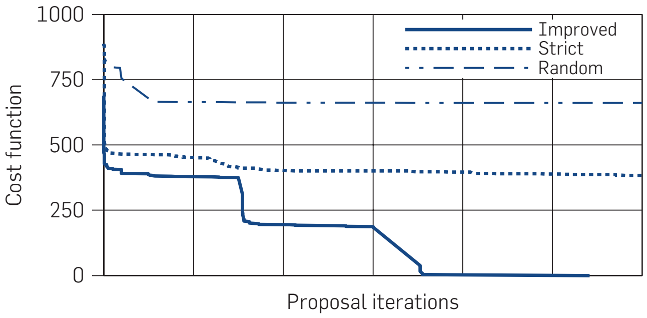
STOKE is a prototype implementation of the ideas described above.[a] STOKE runs a binary created by a standard compiler (in our experiments, `llvm -O0`) under instrumentation to generate test cases for a loop-free target of interest and broadcasts both target and test cases to a set of synthesis threads. After a fixed time, those threads report back with a set of validated rewrites, and a new set of optimization threads are used to improve those rewrites. Of the results, the top 20% most performant are re-ranked based on actual runtime, and the best is returned to the user. STOKE generates 32 test cases for each target and synthesis and optimization are executed in parallel on an 8 core 3.5 GHz Intel i7-4770K with a computational budget of 15 min.

Figure 6. Strict versus relaxed equality functions for a target in which ax is live out and the correct result appears in an incorrect location.

	al	bl	cl	dl
$\mathrm{eval}(\mathcal{T},\cdot)$	1111	0000	0000	0000
$\mathrm{eval}(\mathcal{R},\cdot)$	0000	1000	1100	1111
$\delta=\mathrm{eval}(\mathcal{T},\mathrm{al})\oplus\mathrm{eval}(\mathcal{R},\cdot)$	1111	0111	0011	0000
$\mathrm{pop}(\delta)$	4	3	2	0
$w_m\cdot\mathbf{1}(\mathrm{al}\neq\cdot)$	0	1	1	1

$$\mathrm{reg}(\mathcal{T},\mathcal{R},\tau)=4$$
$$\mathrm{reg}'(\mathcal{T},\mathcal{R},\tau)=\min(4,3+1,2+1,1)$$
$$=1$$

Figure 7. Strict versus relaxed cost functions for the Montgomery multiplication synthesis benchmark. Random search results shown for reference.



STOKE generates test cases using Intel's PinTool.[11] It executes the binary provided by the user, and for every invocation of the target records both initial register state and the set of values dereferenced from memory. For each test case, the set of addresses dereferenced by the target are used to define the sandbox in which candidate rewrites are executed. Dereferences of invalid addresses are trapped and replaced by instructions that produce a constant zero value; reads from undefined locations and floating-point exceptions are handled analogously.

STOKE uses a sound procedure to validate the equality of loop-free code sequences.[2] Both target and rewrite are converted into SMT formulae in the quantifier free theory of bit-vector arithmetic used by Z3,[5] producing a query that asks whether both sequences produce the same side effects on live outputs when executed from the same initial machine state. Depending on type, registers are modeled as between 8- and 256-bit vectors, and memory is modeled as two vectors: a 64-bit address and an 8-bit value (x86_64 is byte addressable).

STOKE asserts the constraint that both sequences agree on the initial machine state with respect to the live inputs defined by the target. For each instruction in the target, it also asserts a constraint that encodes the transformation it represents on the machine state and chains these together to

[a] See https://stoke.stanford.edu.

produce a constraint that describes the state of the live outputs defined by the target. An analogous set of constraints are asserted for the rewrite, and for all pairs of memory accesses at addresses $\texttt{addr}_1$ and $\texttt{addr}_2$, STOKE adds an additional constraint that relates their values: $\texttt{addr}_1 = \texttt{addr}_2 \Rightarrow \texttt{val}_1 = \texttt{val}_2$.

Using these constraints, STOKE asks Z3 if there exists an initial machine state that causes the two sequences to produce different results. If the answer is "no," then the sequences are determined to be equal. If the answer is "yes," then Z3 produces a witness that is converted to a new test case.

STOKE makes two simplifying assumptions to keep runtimes tractable. It assumes that stack addresses are represented exclusively as constant offsets from rsp. This allows stack locations to be treated as named variables in `llvm -O0` code, which exhibits heavy stack traffic. Additionally, it treats 64-bit multiplication and division as uninterpreted functions which are constrained by a small set of special-purpose axioms. Whereas Z3 diverges when reasoning about two or more such operations, our benchmarks contain up to four per sequence.

8. EVALUATION

In addition to the Montgomery multiplication kernel, we evaluate STOKE on benchmarks drawn from both the literature and high-performance codes. Performance improvements and runtimes are summarized in Figure 8. Beginning from binaries compiled using `llvm -O0`, STOKE consistently produces rewrites that match the performance of code produced by `gcc` and `icc` (the two compilers produce essentially identical results). In several cases, the rewrites are comparable in performance to handwritten assembly.

Gulwani et al.[7] identifies Hacker's Delight[18]—a collection of techniques for encoding complex algorithms as small loop-free code sequences—as a source of benchmarks (p01–p25) for program synthesis and optimization. For brevity, we focus only on a kernel for which STOKE discovers an algorithmically distinct rewrite. Figure 9 shows the "Cycle Through 3 Values" benchmark, which takes an input x and transforms it to the next value in the sequence ⟨a, b, c⟩. The most natural implementation of this function is a sequence of conditional assignments, but for ISAs without the required intrinsics, the implementation shown is cheaper than one that uses branches. For x86_64, which has conditional move intrinsics, this is an instance of premature optimization. While STOKE is able to rediscover the optimal implementation, `gcc` and `icc` transcribe the code as written.

Single-precision Alpha X Plus Y (SAXPY) is a level 1 vector operation in the Basic Linear Algebra Subsystems Library.[3] It makes heavy use of heap accesses and presents the opportunity to use vector intrinsics. To expose this property, our integer implementation is unrolled four times by hand, as shown in Figure 10. Despite annotation to indicate that *x* and *y* are aligned and do not alias, neither production compiler is able to produce vectorized code. STOKE on the other hand, is able to discover the ideal implementation.

Figure 8. Speedups over `llvm -O0` versus STOKE runtimes. Benchmarks for which an algorithmically distinct rewrite was discovered are shown in bold; synthesis timeouts are annotated with a –.

	Speedup (×100%)		Runtime (s)	
	gcc/icc -O3	STOKE	Synth.	Opt.
p01	1.60	1.60	0.15	3.05
p02	1.60	1.60	0.16	3.14
p03	1.60	1.60	0.34	3.45
p04	1.60	1.60	2.33	3.55
p05	1.60	1.60	0.47	3.24
p06	1.60	1.60	1.57	6.26
p07	2.00	2.00	1.34	3.10
p08	2.20	2.20	0.63	3.24
p09	1.20	1.20	0.26	3.21
p10	1.80	1.80	7.49	3.61
p11	1.50	1.50	0.87	3.05
p12	1.50	1.50	5.29	3.34
p13	3.25	3.25	0.22	3.08
p14	1.86	1.86	1.43	3.07
p15	2.14	2.14	2.83	3.17
p16	1.80	1.80	6.86	4.62
p17	2.60	2.60	10.65	4.45
p18	2.44	2.50	0.30	4.04
p19	1.93	1.97	-	18.37
p20	1.78	1.78	-	36.72
p21	1.62	1.65	6.97	4.96
p22	3.38	3.41	0.02	4.02
p23	5.53	6.32	0.13	4.36
p24	4.67	4.47	-	48.90
p25	2.17	2.34	3.29	4.43
mont mul	2.84	4.54	319.03	111.64
linked list	1.10	1.09	3.94	8.08
SAXPY	1.82	2.46	10.35	6.66

Figure 9.Cycling Through 3 Values benchmark.

```
int p21(int x, int a, int b, int c) {
  return ((-(x == c)) & (a ^ c)) ^
         ((-(x == a)) & (b ^ c)) ^ c;
}
```

```
# gcc -O3               # STOKE

movl edx, eax           cmpl edi, ecx
xorl edx, edx           cmovel esi, ecx
xorl ecx, eax           xorl edi, esi
cmpl esi, edi           cmovel edx, ecx
sete dl                 movq rcx, rax
negl edx
andl edx, eax
xorl edx, edx
xorl ecx, eax
cmpl ecx, edi
sete dl
xorl ecx, esi
negl edx
andl esi, edx
xorl edx, eax
```

In closing, we note that STOKE is not without its limitations. Figure 11 shows the Linked List Traversal benchmark of Bansal and Aiken.[2] The code iterates over a list of integers and doubles each element. Because STOKE is only able to

Figure 10. SAXPY benchmark.

```
void SAXPY(int* x, int* y, int a) {
  x[i]   = a * x[i]   + y[i];
  x[i+1] = a * x[i+1] + y[i+1];
  x[i+2] = a * x[i+2] + y[i+2];
  x[i+3] = a * x[i+3] + y[i+3];
}
```

```
# gcc -O3                  # STOKE

movslq ecx,rcx             movd edi,xmm0
leaq (rsi,rcx,4),r8        shufps 0,xmm0,xmm0
leaq 1(rcx),r9             movups (rsi,rcx,4),xmm1
movl (r8),eax              pmullw xmm1,xmm0
imull edi,eax              movups (rdx,rcx,4),xmm1
addl (rdx,rcx,4),eax       paddw xmm1,xmm0
movl eax,(r8)              movups xmm0,(rsi,rcx,4)
leaq (rsi,r9,4),r8
movl (r8),eax
imull edi,eax
addl (rdx,r9,4),eax
leaq 2(rcx),r9
addq 3,rcx
movl eax,(r8)
leaq (rsi,r9,4),r8
movl (r8),eax
imull edi,eax
addl (rdx,r9,4),eax
movl eax,(r8)
leaq (rsi,rcx,4),rax
imull (rax),edi
addl (rdx,rcx,4),edi
movl edi,(rax)
```

Figure 11. Linked List Traversal benchmark.

```
while (head != 0) {
  head->val *= 2;
  head = head->next;
}
```

```
# gcc -O3                  # STOKE

movq -8(rsp), rdi          .L1:
.L1:                       movq -8(rsp), rdi
sall (rdi)                 sall (rdi)
movq 8(rdi), rdi           movq 8(rdi), rdi
.L2:                       movq rdi, -8(rsp)
testq rdi, rdi             .L2:
jne .L1                    movq -8(rsp), rdi
                           testq rdi, rdi
                           jne .L1
```

reason about loop-free code—recent work has explored solutions to this problem[15]—it fails to eliminate the stack movement at the beginning of each iteration. STOKE is also unable to synthesize a rewrite for three of the Hacker's Delight benchmarks. Nonetheless, using its optimization phase alone it is able to discover rewrites that perform comparably to the production compiler code.

9. CONCLUSION

We have shown a new approach to program optimization based on stochastic search. Compared to a traditional compiler, which factors optimization into a sequence of small independently solvable subproblems, our approach uses cost minimization and considers the competing constraints of transformation correctness and performance improvement simultaneously. Although the method sacrifices completeness, it is competitive with production compilers and has been demonstrated capable of producing code that can out-perform expert handwritten assembly.

This article is based on work that was originally published in 2013. Since then, STOKE has undergone substantial improvement; the updated results that appear here were produced using the current implementation and improve on the original by over an order of magnitude. Interested readers are encouraged to consult the original text *Stochastic Superoptimization*[13] and those that followed.

Data-Driven Equivalence Checking[15] describes extensions to STOKE that enable the optimization of code sequences with non-trivial control flow. It defines a sound method for guaranteeing that the optimizations produced by STOKE are correct for all possible inputs even in the presence of loops. The method is based on a data-driven algorithm that observes test case executions and automatically infers invariants for producing inductive proofs of equivalence. The prototype implementation is the first sound equivalence checker for loops written in x86_64 assembly.

Stochastic Optimization of Floating-Point Programs with Tunable Precision[14] describes extensions to STOKE that enable the optimization of floating-point code sequences. By modifying the definition of the eq(·) term to account for relaxed constraints on floating-point equality STOKE is able to generate reduced precision implementations of Intel's handwritten C numeric library that are up to six times faster than the original, and achieve end-to-end speedups of over 30% on high-performance applications that can tolerate a loss of precision while still remaining correct. Because these optimizations are mostly not amenable to formal verification using the current state of the art, the paper describes a search technique for characterizing maximum error.

References

1. Andrieu, C., de Freitas, N., Doucet, A., Jordan, M.I. An introduction to MCMC for machine learning. *Machine Learning 50*, 1–2 (2003), 5–43.
2. Bansal, S., Aiken, A. Binary translation using peephole superoptimizers. In *Proceedings of the 8th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2008.* R. Draves and R. van Renesse, eds. (San Diego, CA, USA, December 8–10, 2008). USENIX Association, 177–192.
3. Blackford, L.S., Demmel, J., Dongarra, J., Duff, I., Hammarling, S., Henry, G., Heroux, M., Kaufman, L., Lumsdaine, A., Petitet, A., Pozo, R., Remington, K., Whaley, R.C. An updated set of basic linear algebra subprograms (BLAS). *ACM Trans. Math. Softw. 28*, 2 (2002), 135–151.
4. Cadar, C., Dunbar, D., Engler, D.R. Klee: Unassisted and automatic generation of high-coverage tests for complex systems programs. In *Proceedings of the 8th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2008.* R. Draves and R. van Renesse, eds. (San Diego, CA, USA, December 8–10, 2008). USENIX Association, 209–224.
5. Ganesh, V., Dill, D.L. A decision procedure for bit-vectors and arrays. In *Proceedings of the 19th International Conference on Computer Aided Verification (CAV) 2007.* W. Damm and H. Hermanns, eds. Volume of 4590 *Lecture Notes in Computer Science* (Berlin, Germany, July 3–7, 2007), Springer, 519–531.
6. Gilks, W.R. *Markov Chain Monte Carlo in Practice.* Chapman and Hall/CRC, 1999.
7. Gulwani, S., Jha, S., Tiwari, A., Venkatesan, R. Synthesis of loop-free programs. In *Proceedings of the 32nd ACM SIGPLAN*

Conference on Programming Language Design and Implementation (PLDI) 2011. M. W. Hall and D. A. Padua, eds. (San Jose, CA, USA, June 4–8, 2011). ACM, 62–73.

8. Hastings, W.K. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika 57*, 1 (1970), 97–109.
9. Joshi, R., Nelson, G., Randall, K.H. Denali: A goal-directed superoptimizer. *In Proceedings of the ACM SIGPLAN 2002 Conference on Programming Language Design and Implementation* (Berlin, Germany, 2002). ACM, New York, NY, USA, 304–314.
10. Liang, P., Jordan, M.I., Klein, D. Learning programs: A hierarchical Bayesian approach. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10).* J. Fürnkranz and T. Joachims, eds. (Haifa, Israel, June 21–24, 2010). Omnipress, 639–646.
11. Luk, C.-K., Cohn, R.S., Muth, R., Patil, H., Klauser, A., Lowney, P.G., Wallace, S., Reddi, V.J., Hazelwood, K.M. Pin: Building customized program analysis tools with dynamic instrumentation. In *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation* (Chicago, IL, USA, 2005). ACM, New York, NY, USA, 190–200.
12. Massalin, H. Superoptimizer – A look at the smallest program. In *Proceedings of the Second International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS II).* R. H. Katz, ed. (Palo Alto, CA, USA, October 5–8, 1987). ACM Press, 122–126.
13. Schkufza, E., Sharma, R., Aiken, A. Stochastic superoptimization. In *Architectural Support for Programming Languages and Operating Systems, ASPLOS'13.* V. Sarkar and R. Bodík, eds. (Houston, TX, USA, March 16–20, 2013). ACM, 305–316.
14. Schkufza, E., Sharma, R., Aiken, A. Stochastic optimization of floating-point programs with tunable precision. In *ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI'14.* M. F. P. O'Boyle and K. Pingali, eds. (Edinburgh, United Kingdom, June 09–11, 2014). ACM.
15. Sharma, R., Schkufza, E., Churchill, B.R., Aiken, A. Data-driven equivalence checking. In *Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages & Applications, OOPSLA 2013, Part of SPLASH 2013.* A. L. Hosking, P. Th. Eugster, and C. V. Lopes, eds. (Indianapolis, IN, USA, October 26–31, 2013). ACM, 391–406.
16. Solar-Lezama, A., Tancau, L., Bodík, R., Seshia, S.A., Saraswat, V.A. Combinatorial sketching for finite programs. In *Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2006.* J. P. Shen and M. Martonosi, eds. (San Jose, CA, USA, October 21–25, 2006). ACM, 404–415.
17. Tate, R., Stepp, M., Tatlock, Z., Lerner, S. Equality saturation: A new approach to optimization. *Logical Methods Comput. Sci. 7*, 1 (2011).
18. Warren, H.S. *Hacker's Delight.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

Eric Schkufza, Rahul Sharma, and Alex Aiken ([eschkufza, sharmar, aiken]@cs.stanford.edu), Stanford University, Stanford, CA.

CAREERS

Boise State University

Department of Computer Science

Three Open Rank, Tenured/Tenure-Track Faculty Positions

The **Department of Computer Science** at **Boise State University** invites applications for **three** open rank, tenured/tenure-track faculty positions. Seeking applicants in the areas of data science (and associated areas in machine learning, databases and information systems, data mining, distributed systems, natural language processing, visualization), cybersecurity, and computer science education research. Strong applicants from other areas of computer science will also be considered.

Applicants should have a commitment to excellence in teaching, a desire to make significant contributions in research (supported by a substantially reduced teaching assignment), and experience in collaborating with faculty and local industry to develop and sustain funded research programs. A PhD in Computer Science or a closely related field is required by the date of hire. For additional information, please visit http://coen.boisestate.edu/cs/jobs

California State University, Chico

CSUC Computer Science Dept.

Assistant Professor

Dept. of Computer Science has one fulltime tenure track Asst. Prof position, starting 8/2016. EOE Employer. Please see the full Announcement at: http://jobs.csuchico.edu/postings/3255

George Mason University

Department of Computer Science

Tenure-track Faculty Positions

The Department of Computer Science in the Volgenau School of Engineering at George Mason University invites applications for tenure-track faculty positions beginning Fall 2016. Candidates must have (or be close to completing before the start date of the position) a PhD in Computer Science or a related field, demonstrated potential for excellence and productivity in research, and a commitment to high quality teaching. Exceptionally strong senior candidates may also be considered, and must have an established record of outstanding research and excellent teaching. Such candidates will be eligible for tenured Associate Professor or Professor positions.

While applicants in all areas of computer science will be given serious consideration, we are particularly interested in candidates in the areas of cyber-physical systems, mobile and ubiquitous computing, data-intensive computing, distributed systems and cloud computing, and software engineering.

The department has over 40 faculty members with wide-ranging research interests including artificial intelligence, algorithms, autonomic computing, computational biology, computer graphics, computer vision, databases, data mining, parallel and distributed systems, real-time systems, robotics, security, software engineering, and wireless and mobile computing. The CS department has over $5 Million in annual research funding and has 11 recipients of NSF's prestigious CAREER awards.

In addition to BS, MS and PhD programs in Computer Science, the department offers MS programs in Information Systems, Information Security and Assurance, and Software Engineering. The department also participates in an inter-disciplinary MS in Data Analytics Engineering offered by the Volgenau School of Engineering. For more information on the department, visit our Web site: http://cs.gmu.edu/. George Mason University is located in Fairfax in the Northern Virginia suburbs of Washington, DC. Northern Virginia is home to one of the largest concentrations of high-tech firms in the nation, providing excellent opportunities for interaction with government agencies and industry. Fairfax is consistently rated as being among the best places to live in the country, and has an outstanding local public school system. George Mason has grown in leaps and bounds since its inception in 1972 and is consistently ranked among the top up-and-coming universities in the USA by US News and World Report.

For full consideration, qualified applicants must apply online at http://jobs.gmu.edu for position F9995Z. The review of applications will begin February 15, 2016 and will continue until the position is filled.

George Mason University

Department of Computer Science

Computer Science Department – Term Instructor/Term Assistant Professor Position

The Department of Computer Science at George Mason University invites applications for renewable, term, non tenure-track Instructor and Assistant Professor positions beginning Fall 2016. Responsibilities include teaching undergraduate computer science courses as well as service duties associated with the department's undergraduate degree programs. Minimum qualifications for an Assistant Professor position include a Ph.D. in computer science, software engineering or related field. Applicants with an MS degree in Computer Science or a related field will be considered for Instructor positions. Applicants should possess a strong commitment to and demonstrated excellence in teaching.

The department has over 40 faculty members with wide-ranging research interests including artificial intelligence, algorithms, computational biology, computer graphics, computer vision, databases, data mining, parallel and distributed systems, real-time systems, robotics, security, software engineering, and wireless and mobile computing. The CS department has over $5 Million in annual research funding and has 11 recipients of NSF's prestigious CAREER awards. For more information on the department, visit our Web site: http://cs.gmu.edu/.

George Mason University is located in Fairfax in the Northern Virginia suburbs of Washington, DC. Northern Virginia is home to one of the largest concentrations of high-tech firms in the nation, providing excellent opportunities for interaction with government agencies and industry. Fairfax is consistently rated as being among the best places to live in the country, and has an outstanding local public school system. George Mason has grown in leaps and bounds since its inception in 1972 and is consistently ranked among the top up-and-coming universities in the USA by US News and World Report.

For full consideration, please complete the online application at http://jobs.gmu.edu for position F9796z. The review of applications will begin on February 1, 2016 and will continue until the positions are filled.

George Mason University is an Equal Opportunity/Affirmative Action/Veterans/Disability Employer encouraging diversity.

Princeton University

Computer Science

Tenure Track Position(s)

The Department of Computer Science at Princeton University invites applications for faculty positions at the Assistant Professor level. We are accepting applications in all areas of Computer Science. Applicants must demonstrate superior research and scholarship potential as well as teaching ability.

A PhD in Computer Science or a related area is required. Candidates should expect to receive their PhD before Fall, 2016. More senior appointments may be considered for extraordinary candidates at the associate and full professor levels. Successful candidates are expected to pursue an active research program and to contribute significantly to the teaching programs of the department. Applicants should include a CV and contact information for at least three people who can comment on the applicant's professional qualifications.

For full consideration we recommend that applicants apply by December 1, 2015, though we will continue to review applications past that date.

Princeton University is an equal opportunity employer. All qualified applicants will receive consideration for employment without regard to race, color, religion, sex, national origin, disability status, protected veteran status, or any other characteristic protected by law. This position is subject to the University's background check policy.

You may apply online at:
http://jobs.cs.princeton.edu/
Requisition Number: 1500928

Siena College

Assistant Professors

The CS Dept at Siena College invites applications for 2 tenure-track positions beginning Sept 2016. The positions require a PhD in CS, IS, or a related area. Applicants must be committed to teaching and research. Candidates with any areas of specialization are encouraged to apply. See www.siena.edu/cssearch for complete details. The department will begin reviewing applications on Feb 1, 2016.

Swarthmore College

Computer Science Department

Lab Lecturer

The Department of Computer Science is currently accepting applications for a Lab Lecturer. The Lab Lecturer position is full time during the academic year (Fall and Spring semesters) with summers off. The start date is January 11, 2016.

Swarthmore College has a strong institutional commitment to excellence through diversity and in its educational program and employment practices. The College actively seeks and welcomes applications from candidates with exceptional qualifications, particularly those with demonstrated commitments to a more inclusive society and world.

Swarthmore College is a small, selective, liberal arts college located 10 miles outside of Philadelphia. The Computer Science Department offers majors and minors at the undergraduate level.

The Lab Lecturer position is an Instructional Staff position at the college. The responsibilities of the position include, but are not limited to: teaching lab sections of the introductory courses in the Computer Science Department; working with faculty to develop lab assignments for the introductory courses; creating lab assignment write-ups and documentation on tools used in introductory labs; supporting faculty in creating and setting up lab code examples, documentation, and software tools for lab work; lab grading and coordinating student graders; and holding regular office hours and helping students in the lab during open lab hours. More information about the Computer Science Department can be found on our website at www.cs.swarthmore.edu.

A master's degree or Ph.D. in computer science or a related field with extensive computer science background is required. Prior teaching experience at the college level is preferred.

Applications should include a vita, teaching statement, and two letters of reference that speak to the candidate's teaching ability.

Applications are being accepted online at https://academicjobsonline.org/ajo/jobs/6465. We will begin reviewing applications on November 9. Applications will continue to be accepted until the position is filled.

Swarthmore College

Computer Science Department

Tenure Track and Visiting Positions

The Computer Science Department invites applications for one tenure-track position and multiple visiting positions at the rank of Assistant Professor to begin Fall semester 2016.

Swarthmore College has a strong institutional commitment to excellence through diversity and inclusivity in its educational program and employment practices. The College actively seeks and welcomes applications from candidates with exceptional qualifications, particularly those with demonstrated commitments to a more inclusive society and world.

Swarthmore College is a small, selective, liberal arts college located 10 miles outside of Philadelphia. The Computer Science Department offers majors and minors at the undergraduate level.

Applicants must have teaching experience and should be comfortable teaching a wide range of courses at the introductory and intermediate level. Candidates should additionally have a strong commitment to involving undergraduates in their research. A Ph.D. in Computer Science at or near the time of appointment is required.

For the tenure-track position, we are particularly interested in applicants whose areas will complement and broaden our program, including theory and algorithms, programming languages, and systems areas. Strong applicants in other areas will also be considered.

For the visiting position, strong applicants in any area will be considered.

For the tenure-track position, priority will be given to complete applications received by December 15. For the visiting position, priority will be given to complete applications received by February 15. Applications for both positions will

NATIONAL RESEARCH FOUNDATION
PRIME MINISTER'S OFFICE
SINGAPORE

Singapore NRF Fellowship 2017

The Singapore National Research Foundation (NRF) invites outstanding researchers in their early stage of research careers to apply for the Singapore NRF Fellowship.

The Singapore NRF Fellowship offers:

- **A five-year research grant of up to SGD 3 million (approximately USD 2.4 million).**
- **Freedom to lead ground-breaking research in any discipline of science and technology in a host institution of choice in Singapore.**

The NRF Fellowship is open to all researchers (PhD holders) who are ready to lead independent research in Singapore.

The Call is now open!

Apply online at **https://rita.nrf.gov.sg**

Closing Date: **23 February 2015, 3pm (GMT+8)**

For further queries, email us at **NRF_Fellowship@nrf.gov.sg**

About the National Research Foundation, Prime Minister's Office, Singapore

The National Research Foundation (NRF), set up on 1 January 2006, is a department within the Prime Minister's Office. The NRF sets the national direction for research and development (R&D) by developing policies, plans and strategies for research, innovation and enterprise. It also funds strategic initiatives and builds up R&D capabilities by nurturing research talent. The NRF aims to transform Singapore into a vibrant R&D hub that contributes towards a knowledge-intensive, innovative and entrepreneurial economy; and make Singapore a magnet for excellence in science and innovation. For more information, please visit ***www.research.gov.sg***.

continue to be accepted after these dates until the positions are filled.

Applications should include a cover letter, vita, teaching statement, research statement, and three letters of reference, at least one (preferably two) of which should speak to the candidate's teaching ability. In your cover letter, please briefly describe your current research agenda; what would be attractive to you about teaching in a liberal arts college environment; and what background, experience, or interests are likely to make you a strong teacher of Swarthmore College students.

Tenure-track applications are being accepted online at https://academicjobsonline.org/ajo/jobs/6161.

Visiting applications are being accepted online at https://academicjobsonline.org/ajo/jobs/6173. Candidates may apply for both positions.

Texas State University

Department of Computer Science

Assistant Professor

Applications are invited for a tenure-track **Assistant Professor** position in any field of computer science to start on September 1, 2016. Consult the department's recruiting page at www.cs.txstate.edu/employment/faculty/ for job duties, qualifications, application procedure, and information about the department and the university.

Texas State University will not discriminate against any person in employment or exclude any person from participating in or receiving the benefits of any of its activities or programs on any basis prohibited by law, including race, color, age, national origin, religion, sex, disability, veterans' status, sexual orientation, gender identity, or gender expression. Texas State is committed to increasing the number of women and minorities in faculty and senior administrative positions. Texas State is a member of The Texas State University System.

Truckee Meadows Community College

Computer Technologies Instructor,
Tenure Track

Truckee Meadows Community College (TMCC), located in Reno, Nevada is seeking applicants for a full-time, tenure track, Computer Technologies Instructor. TMCC is an EEO/AA

Apply for this Job: Apply URL:
http://jobs.tmcc.edu/postings/730
Contact: Human Resources
Email: humanresources@tmcc.edu
Phone: (775) 673 - 7168

University of Illinois at Chicago

Department of Computer Science

Non-Tenure Track Teaching Faculty – Computer Science

The Computer Science Department at the University of Illinois at Chicago is seeking a full-time teaching faculty member beginning fall 2016. This is a long-term, career-oriented position with the possibility of advancement through the university's Clinical track. The department is committed to effective teaching, and candidates would be working alongside seven full-time teaching faculty with over 100 years of combined teaching experience and 11 awards for excellence in teaching. The department is looking for candidates dedicated to teaching; candidates must have evidence of effective teaching, or present a convincing case of future dedication and success in the art of teaching. Content areas of interest include introductory programming/data structures, theory/algorithms, computer systems, databases, software design, and web development. The teaching load is three undergraduate courses per semester, with a possibility of teaching at the graduate level if desired.

The University of Illinois at Chicago (UIC) is ranked in the top-5 best US universities under 50 years old (Times Higher Education), and one of the top-10 most diverse universities in the US (US News and World Report). UIC's hometown of Chicago epitomizes the modern, livable, vibrant city. Located on the shore of Lake Michigan, Chicago offers an outstanding array of cultural and culinary experiences. As the birthplace of the modern skyscraper, Chicago boasts one of the world's tallest and densest skylines, combined with an 8100-acre park system and extensive public transit and biking networks. Its airport is the second busiest in the world, with frequent non-stop flights to most major cities. Yet the cost of living is surprisingly low, whether in a high-rise downtown or a house on a tree-lined street in one of the nation's finest school districts.

Minimum qualifications include a Master's degree or higher in Computer Science or a related field, and either (a) demonstrated evidence of effective teaching, or (b) convincing argument of future dedication and success in the art of teaching. Applications are submitted online at https://jobs.uic.edu/. In the online application, please include your curriculum vitae, the names and addresses of at least three references, a statement providing evidence of effective teaching, and a separate statement describing your past experience in activities that promote diversity and inclusion (or plans to make future contributions). Applicants needing additional information may contact Professor Joe Hummel, Search Committee Chair, jhummel2@uic.edu.

For fullest consideration, please apply by February 15, 2016. We will continue to accept until the position is filled. The University of Illinois is an Equal Opportunity, Affirmative Action employer. Minorities, women, veterans and individuals with disabilities are encouraged to apply.

Washington State University

School of Electrical Engineering & Computer Science

Clinical Asst/Assc/Full Professor in Software Engineering at Everett

Software Engineering faculty - WSU Sch of Electrical Engineering & Computer Science is hiring FT non-TT faculty for Everett WA campus. For information and to apply, please visit https://www.wsujobs.com/postings/22475. WSU is EO/AA Educator & Employer. Apply URL: https://www.wsujobs.com/postings/22475

[CONTINUED FROM P. 128] are auditioning to be characters.

The circles overlap. Old Malcolm the poet was sitting down with Donald and Sheena, a couple who work the langoustine fishery on the other side of the headland. Donald waved me over. Their conversation was, you might say, heated.

“It’s the long-term temperature anomaly you have to look at,” said Sheena. She shot a glance at my dripping parka and damp-patch knees. “We all know it’s cold outside.”

“Ah, but,” said Malcolm, after a moment’s hesitation, “the problem with that is the data corrections, which may distort the record.”

“That’s all well and good,” said Donald. He paused, as if to think. “But there is a lot of evidence that has nothing to do with weather stations or even satellite data. And we can see that with our own eyes. Take the population distribution of sessile mollusca—that’s shellfish, to you and me—and jellyfish. They’re migrating north. Two kilometres a year, on average.”

Sheena nodded. “Aye, and I’ve seen that myself on what comes up in trawls. The mussel beds are gone. And the jellyfish are all through the water. It’s good for us, mind—keeps the seabed clear for the langoustines and other edible arthropoda. But you can’t tell me nothing’s happening.”

“Oh, I’m not saying nothing’s happening,” said Malcolm, hands around a warming glass of Springbank. “I’m saying the case for anthropogenic forcing is by no means as firmly established as … ” His thick white eyebrows lifted. “What are you staring at?”

“All of you,” I said. “What’s got into you? You’ve always been a bit of a scoffer, Mal, and Sheena and Donald the opposite, but usually you just yell at each other. Now you’re all talking like you’ve swallowed Wikipedia.”

“Och, no,” said Donald, as they all laughed. “We’re on Chatter.”

“Something like Twitter?”

“Not a bit,” said Sheena. “This is *informed* opinion. It’s … ” She turned to Malcolm. “You tell him.”

The old poet leaned back and gazed at the ceiling. “Chatter,” he said, “is an entirely automated social media app which aggregates comment from a wide range of sources and converges to clusters of widely held views and agreed-on facts. It then sends visual and verbal prompts to users, enabling them to converse confidently on any topic at any level. Other uses include fluent ripostes in social interactions of all kinds.” He blinked then smiled wryly. “It says here.”

“Aye,” she said, “but that sliding tackle the Kilmarnock left half did was a foul, no doubt about that.”

“I don’t see the point,” I said. “Why would anyone want to just spout received opinions?”

“Like they don’t already?” said Donald. “But … don’t turn around, just use your glasses … there’s Neil gone up to the bar now, take a look.”

Neil works in the supermarket. Bright lad, but shy and inarticulate. Normally he’d barely meet Moira’s eye. Now he was downright flirting. Moira was flirting back.

“It’s called the Cyrano de Bergerac effect,” said Sheena. “The funny thing is, it works even if all concerned know what’s going on.”

“People can’t be that shallow,” I said.

Malcolm snorted. “You try it.”

He flicked, and there it was: Chatter, on my glasses and whispering away in my ear. The first suggested use was celebrity gossip. Sheena, Donald, and Malcolm were looking at me expectantly. I let the app stay where it was.

“So, that Alma Stevenson?” I said. “Do you think she’s really breaking up with Maxc-D, even after her new album? And her pregnant and everything!”

A moment’s pause. “Sure the baby’s his?” said Donald.

Sheena looked indignant. “Ah, come on, that’s going too … ”

Photos and captions flashed before my eyes. “She was seen leaving an IVF clinic two weeks ago,” I pointed out.

“Nah,” said Malcolm. “She’s long had problems in that area, and,” he leaned forward, speaking quietly, “some say she’s a good 10 years older than she claims.”

“Ooh!” I said.

Before long we were squealing and our drinks were sunk. We blinked out of it.

“My round,” said Malcolm.

“No, mine,” I said.

We had a brief verbal tussle that ended in us sharing the round. I went to the bar with him. Malcolm ordered, then nudged me and glanced at Moira.

I’d blinked away the celebrity gossip layer. The app went to the next default topic: football.

“Another bad week for Morton,” I said.

Moira batted her eyelashes.

“Aye,” she said, “but that sliding tackle the Kilmarnock left half did was a foul, no doubt about that.”

“Ah come on,” I said. “The ref saw it and let it go, so McClafferty was … ”

This well-informed-fan chit-chat continued as Moira set up three pints and a gin and tonic. Then she laughed.

“You don’t follow football, do you?”

I shook my head.

“Well, I do,” said Moira, shoving the drinks across the counter. “But I’ve fallen behind a bit since I got interested in politics.”

“So what you said about Belgium wasn’t from Chatter?”

“Of course not.”

I gave her a shamefaced look. “Sorry.”

“That’s what Neil said,” she told me. She laughed again. “But I’m still seeing him tomorrow.”

Quick as a flash, I came back with: “In the library?”

She laughed. “You’ve got the hang of Chatter all right.”

“No, no,” I said, picking up two pints. “That was just me.”

“That’s what they all say,” she said.

I deleted the app and went back to the table with my friends.

Ken MacLeod (kenneth.m.macleod@gmail.com) is the author of 15 novels, from *The Star Fraction* (Orbit Books, London, 1995) to *The Corporation Wars: Dissidence* (Orbit Books, London, 2016). He blogs at The Early Days of a Better Nation (http://kenmacleod.blogspot.com) and tweets as @amendlocke.

From the intersection of computational science and technological speculation, with boundaries limited only by our ability to imagine what could be.

DOI:10.1145/2859012

Ken MacLeod

Future Tense
Chatterbox

A social network can sometimes make more of us than we ought to be.

I'M NOT WHAT you'd call an early adopter. I clung so long to screens that my late-model eyewear is to me shiny and new. Social media? Don't get me started. My Plodar account is still live. PhySkan went from hype to tumbleweed without my noticing. I prefer my social life real. So every so often I go down the steep cobbled street from my flat and turn right along Harbour Road to the Magnus. Last week, I hadn't been out for a drink for months. But I'd finished a draft, and I owed myself a pint.

A dark December evening. Sleet in my face from the wind that curls around the headland and moans through the ruined castle and makes the rigging of the sailing boats in the harbor chime against their masts. Out in the Sound a last ferry of the day chugged, its flat prow butting spray. I hurried past the newsagent, the pottery, and the art gallery, and ducked into the Magnus, throwing back my hood and shrugging my parka.

The King Magnus Crown: old pub, small town. Oak beams and yellowed walls. Ropes, glass floats, plaques of ships' crests, the odd rusty sword and tattered flag. A silent, dusty television screen.

IMAGE BY ANDRIJ BORYS ASSOCIATES, USING PHOTO BY STUART MONK

Behind the bar: bottles of spirits; cask ale; Moira. A graphics student, she works in the gallery on Saturdays and the bar on Wednesday evenings. She knows me well enough to draw a pint of Best at a nod. While the head was settling she did that fast blink people do with contacts, gazed off into space and shook her head with a heavy sigh.

"What's the matter?" I asked.

She flickered her fingers in front of her eyes, flipping the image to my glasses. A soldier in powered armor punched though an interior wall in Brussels. Screams, a pop of gas grenades, commentary in French. I waved the news clip away.

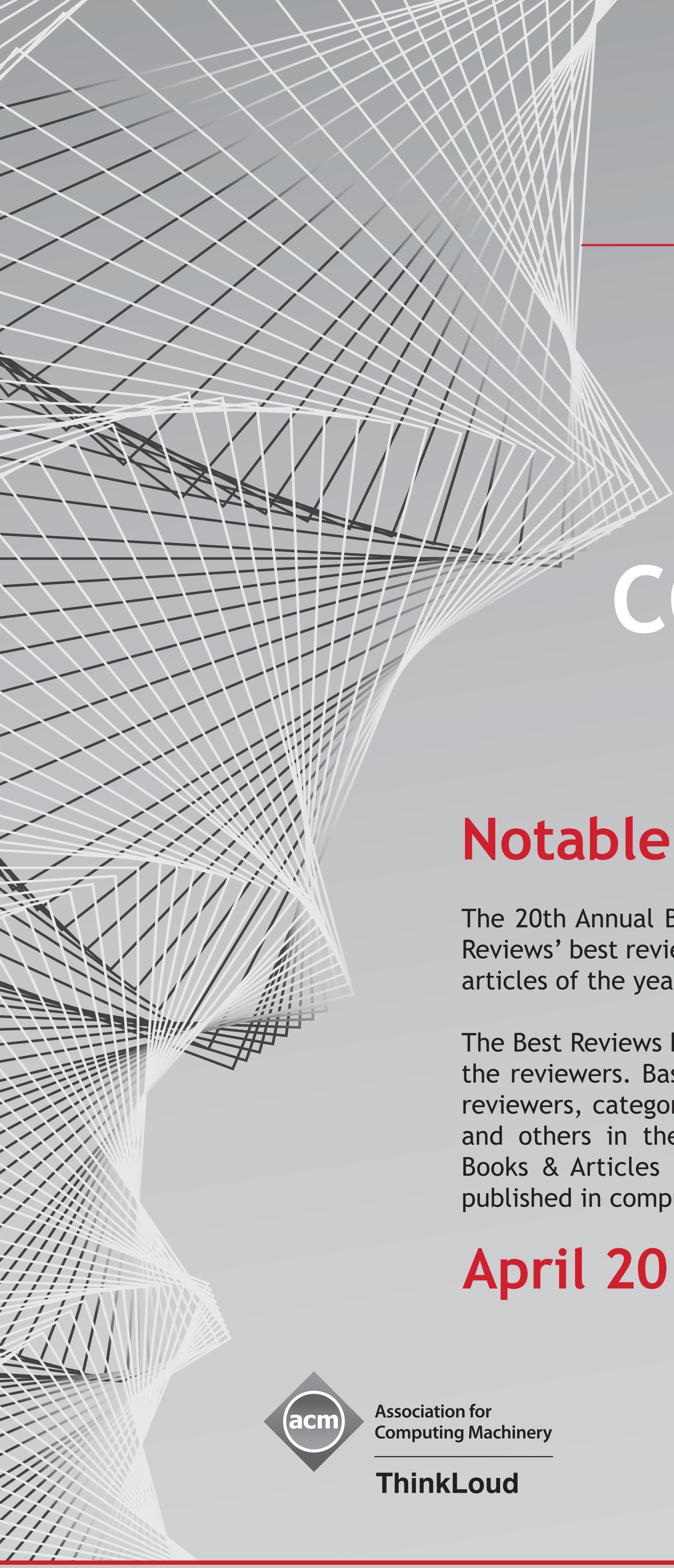
"Yeah, it's grim over there," I said, carefully noncommittal.

"Still, doesn't mean the French should go in," she said. "It could push the two Walloon factions together, and lead to cross-border trouble."

"Good point," I said. Moira had never before shown the slightest interest in European politics. I wasn't inclined to say more. She pushed the pint across.

I thanked her, waved my paycard and sipped, looking around. It was busy for a mid-week night. Maybe everyone else had finished a first draft, too. At local arts events you can get the impression that half the people in the town are aspiring writers and the other half

[CONTINUED ON P. 127]



fuse
DIS 2016

Designing Interactive Systems

June 4 – 8

Brisbane Australia

dis2016.org

bit.ly/dis16

@DIS2016

Association for Computing Machinery

SIGCHI

QUT
Queensland University of Technology
Brisbane Australia