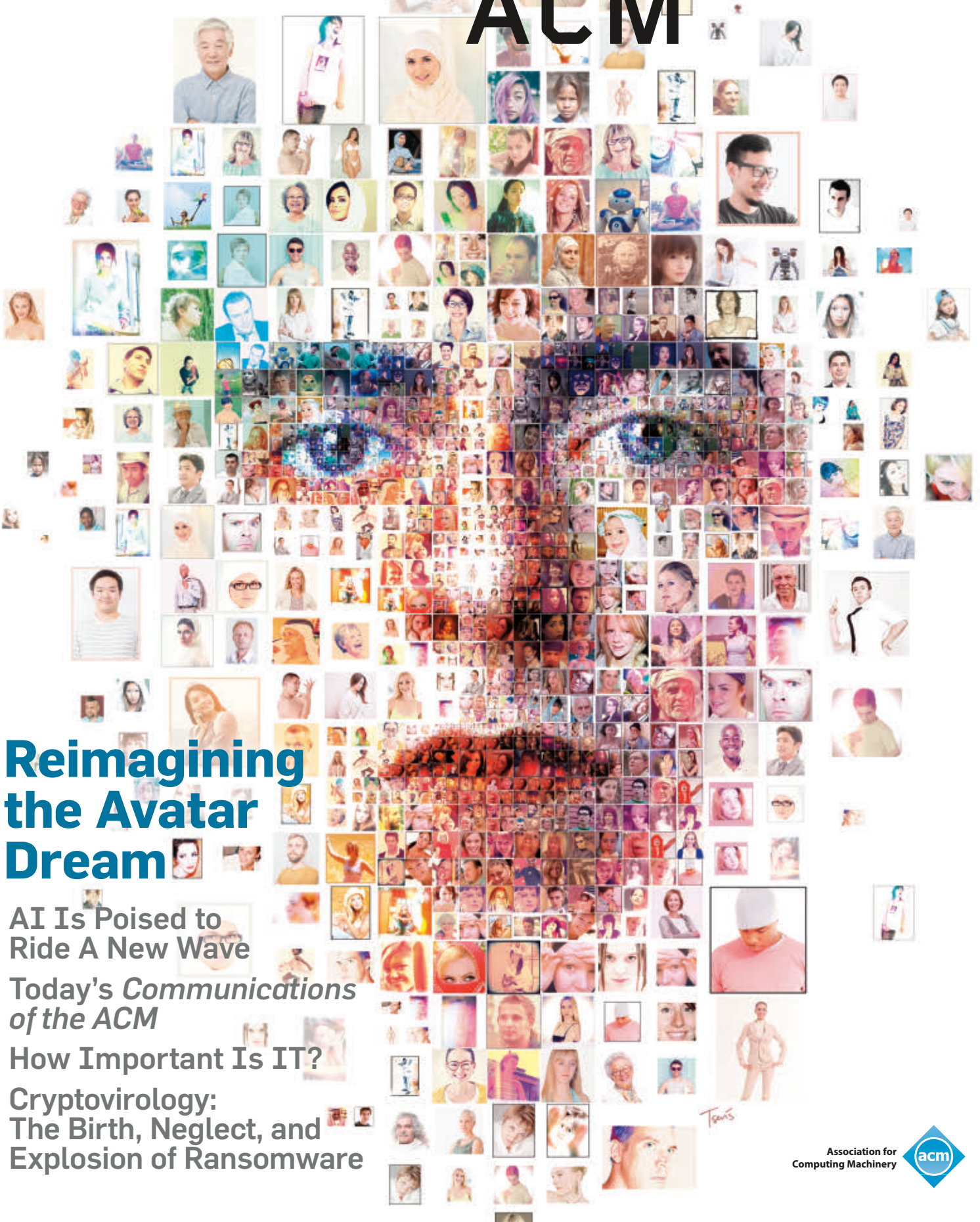
COMMUNICATIONS OF THE ACM

CACM.ACM.ORG

07/2017 VOL.60 NO.07



Reimagining the Avatar Dream

AI Is Poised to Ride A New Wave

Today's *Communications of the ACM*

How Important Is IT?

Cryptovirology: The Birth, Neglect, and Explosion of Ransomware

Association for Computing Machinery

ACM NanoCom 2017

4TH ACM INTERNATIONAL CONFERENCE ON NANOSCALE COMPUTING AND COMMUNICATION

STUDENT TRAVEL GRANT

SEPTEMBER 27-29, 2017
WASHINGTON DC, USA

Don't miss the opportunity to participate at this exciting, engaging and multi-disciplinary event on innovations and technological creativity at the nanoscale!

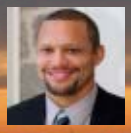
ACM NanoCom 2017 aims at fostering and reinforcing the research community contributing to new nanoscale communications and computing paradigms. The conference will highlight research potentials, stimulate novel and breakthrough ideas, identify the short and medium term exploitation, and feature keynote addresses, poster sessions and tutorials in this emerging inter-disciplinary field and provide an amazing networking opportunity.

The conference will cover topics ranging from electromagnetic and molecular communications to applications in biomedical engineering and consumer electronics at the nanoscale. It will be a great networking opportunity to meet world leading experts, active researchers and young innovators at the frontiers of nanoscale communication whether through molecular techniques or using radio propagation in and out of the body or in the surrounding environment.

KEYNOTES

Prof. Andrea Ferrari, Director, Cambridge Graphene Centre, UK
"Graphene and Related Materials for Photonics and Optoelectronics"

Prof. Gregory F. Payne, University of Maryland, USA
"Redox: A Modality to Bridge Biological and Electronic Communication"



Prof. Douglas Densmore, Boston University, USA
"The Living Computing Project - How Can I Make A Cell Compute?"

/ACMNANOCOM NANOCOM.ACM.ORG /ACMNANOCOM

SIGGRAPH ASIA 2017 BANGKOK

CONFERENCE 27 – 30 November 2017
EXHIBITION 28 – 30 November 2017
BITEC, Bangkok, Thailand

10th Edition

THE CELEBRATION OF LIFE & TECHNOLOGY

CALL FOR SUBMISSIONS

Submit your works & be a presenter at SIGGRAPH Asia!

SIGGRAPH Asia 2017 invites you to submit your works and showcase your outstanding creative ideas and innovations at the 10th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia, taking place from **27 – 30 November**, in Bangkok, Thailand.

Log-on to **sa2017.siggraph.org/submitters** to submit your works.

CONFERENCE PROGRAMS' SUBMISSION DEADLINES*:

DEADLINES	PROGRAMS
27 April 2017	Workshops' Proposals
23 May 2017	Technical Papers
30 May 2017	Emerging Technologies
1 June 2017	Art Gallery
13 June 2017	Symposium on Education
21 June 2017	Symposium on Mobile Graphics and Interactive Applications
28 June 2017	Courses
29 June 2017	Symposium on Visualization
15 July 2017	Student Volunteers - Team Leaders Application
19 July 2017	Computer Animation Festival
30 July 2017	VR Showcase
12 August 2017	Student Volunteers Application
15 August 2017	Posters Technical Briefs Workshops' Papers

**The submission time for all dates is 23:59 UTC/GMT*

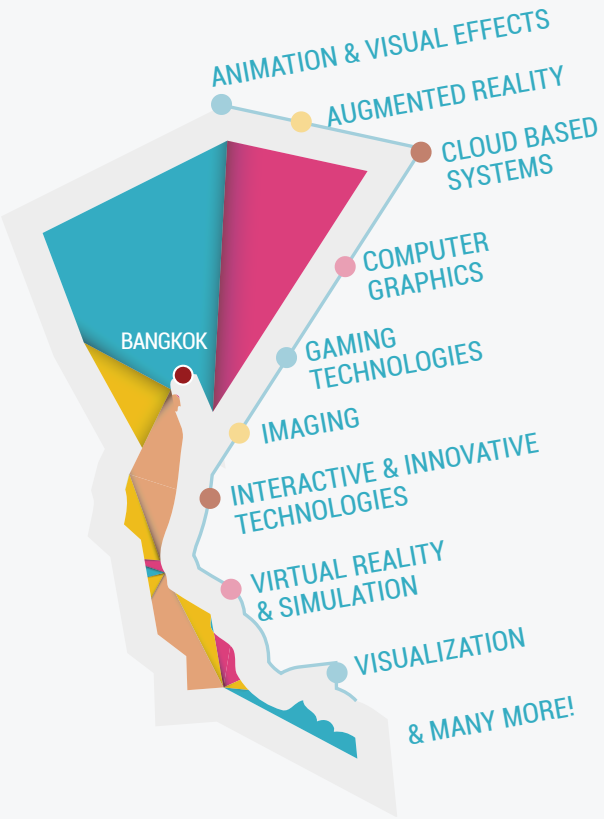
Visit sa2017.siggraph.org for more details.

CALL FOR EXHIBITORS & SPONSORS

Be a part of the SIGGRAPH Asia Exhibition – Asia's Digital Media Marketplace

Meet close to 7,000 technical and creative industry experts and individuals from over 60 countries and regions face-to-face to explore business opportunities, partnerships, and to strengthen existing relations – all in person at SIGGRAPH Asia 2017. Book your stand now to secure your preferred location.

Contact Clariss Chin at **+65 6500 6722** or **clariss.chin@siggraph.org** for more information on the exhibit space options and fees, as well as sponsorship packages.



Sponsored by acm

SA2017.SIGGRAPH.ORG

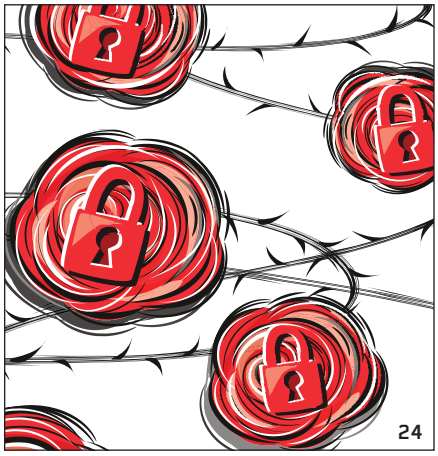
Organized by koelnmesse
we energize your business | since 1924

Departments

Last Byte

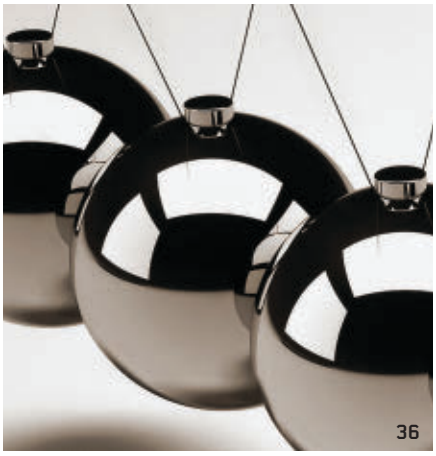
News

Viewpoints



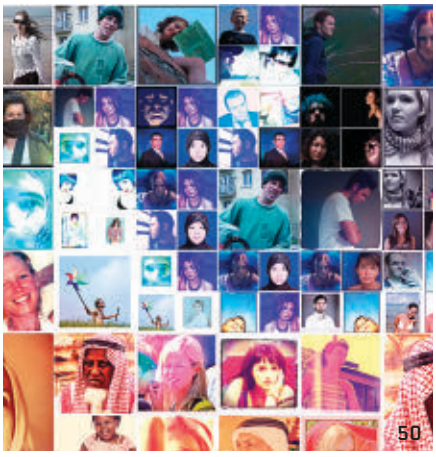
IMAGES BY KATERYNA KON; ALICIA KUBISTA/ANDRIJ BORYS ASSOCIATES

Practice



Articles' development led by ACM queue
queue.acm.org

Contributed Articles



Watch the authors discuss their work in this exclusive *Communications* video. https://cacm.acm.org/videos/reimagining-the-avatar-dream

Review Articles

Research Highlights

Watch the authors discuss their work in this exclusive *Communications* video. https://cacm.acm.org/videos/ironfleet

About the Cover:
The Avatar Dream, as depicted by D. Fox Harrell and Chong-U Lim on p. 50, allows people to imagine using the computer to become whomever they want to be. Their journey goes beyond the typical cartoon-like avatar characters by celebrating diversity and experimenting with different races, cultures, and ethnicities. Cover illustration by Charis Tsevis.

IMAGES BY ORLA/SHUTTERSTOCK; CHARIS TSEVIS

Association for Computing Machinery
Advancing Computing as a Science & Profession

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.

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

DOI:10.1145/3101111

Andrew A. Chien

Today's *Communications of the ACM*

Greetings! It is with great pleasure that I take the helm as the ninth Editor-in-Chief of *Communications*, the flagship publication and ACM's vessel for the most important and interesting happenings across the field of computing. *Communications* is *the* indispensible source of information and insights for the well-educated computing professional around the world.

Computing is unique. In many fields, fundamental breakthroughs take years to reach the market and even longer to achieve full impact. Computing's "clock speed" is far faster;[2] set by exponential hardware performance advances and now driven by rapid and exponential advances in algorithms and cloud services. Clockspeed and computing's extraordinary leverage enables tiny teams to translate insights into powerful change with global reach and impact (for example, this year's Turing Laureate Sir Tim Berners-Lee and the WWW, but also public-key cryptography, social networks, and blockchain, deep learning, and many more). These advances arise from academe, startups, practitioners, and even from garages and basements ... and their worldwide impact inevitably frames profound new problems for deep research. Computing's rapid disruptive change is so powerful and transformative that the term "Internet speed" is deeply embedded in the popular vernacular.

Communications core elements reflect computing's extraordinary dynamism:

- **News:** Updates on hot topics in technology, practice, and public policy
- **Viewpoints:** Reasoned, thought-provoking perspectives that bring out the balance of concerns in critical technology and policy issues
- **Practice:** Frames cutting-edge software challenges and disruptive technologies that are breaking through
- **Contributed Articles:** Peer-reviewed articles of compelling interest spanning computing's breadth; framed to be approachable to the broad computing community
- **Reviews:** New significant developments as seen through the lens of their future impact
- **Research Highlights:** Outstanding research, drawn from ACM's leading SIG conferences, put in context, and made accessible to the educated computing professional
- **Editorial Elements:** Letters to the Editor, blogs, and columns that connect to the community

Each month *Communications* brings these elements together to inform your professional perspective, framing rapid change, new fundamental problems, and disruptive developments in perspective. The magazine's structure is the product of a radical editorial transformation.[1,3]

Each issue is the product of the extraordinary efforts of contributors (authors, columnists, reviewers, bloggers), the editorial board (co-chairs, associate editors), and an extraordinary production team. These volunteer efforts reflect great passion and dedication for the field of computing. Some are reflected on the masthead and bylines, but others remain anonymous. Let me take a moment to thank all of the volunteers who generously contribute their time and leadership each month! And of course, I would be remiss to not include the members of *Communications*' amazing production team who render each issue a polished work of beauty.

In the history of *Communications*, each Editor-in-Chief has faced significant challenges, but by any standard the past 10 years have been an extraordinary period of change and renewal. Today's *Communications* is a dynamic, vibrant reflection and beacon of the computing profession, and that is in no small part due to Moshe Vardi's dynamic, thoughtful, and tireless leadership. So on behalf of the editorial board, the members of the ACM, and the computing profession, thank you for a decade of extraordinary service!

Looking forward, I am acutely aware of computing's relentless advance and rapid "creative destruction;" our challenge is to invent *Communications*' future—thereby ensuring its vitality, relevance, and impact. I hope many of you will join in and enable our success. Next month, I will describe a few new challenges and opportunities that lie directly in our sights!

Andrew A. Chien, EDITOR-IN-CHIEF

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

References

1. Aho, A. and Gottlob, G. A front row seat to *Communications*' editorial transformation. *Commun. ACM 57*, 4 (Apr. 2014), 5; https://doi.org/10.1145/2582611.
2. Fine, C. *Clockspeed: Winning Industry Control in the Age of Temporary Advantage.* MIT Sloan School, Basic Books, 1999.
3. Vardi, M.Y. CACM: Past, present, and future. *Commun. ACM 51*, 1 (Jan. 2008), 44–48; http://dx.doi.org/10.1145/1327452.1327474

ACM Europe Conference

Barcelona, Spain | 7 – 8 September 2017

The ACM Europe Conference, hosted in Barcelona by the Barcelona Supercomputing Center, aims to bring together computer scientists and practitioners interested in exascale high performance computing and cybersecurity.

The High Performance Computing track includes a panel discussion of top world experts in HPC to review progress and current plans for the worldwide roadmap toward exascale computing. The Cybersecurity track will review the latest trends in this very hot field. High-level European Commission officials and representatives of funding agencies are participating.

Keynote Talk by ACM 2012 Turing Award Laureate Silvio Micali, "ALGORAND: A New Distributed Ledger"

Co-located events:

- ACM Europe Celebration of Women in Computing: WomENcourage 2017 (Requires registration, https://womencourage.acm.org/)
- EXCDI, the European Extreme Data & Computing Initiative (https://exdci.eu/)
- Eurolab-4-HPC (https://www.eurolab4hpc.eu/)
- HiPEAC, the European Network on High Performance and Embedded Architecture and Compilation (https://www.hipeac.net/)

Conference Chair: Mateo Valero, Director of the Barcelona Supercomputing Center

Registration to the ACM Europe Conference is free of charge for ACM members and attendees of the co-located events.

acm Europe Council

http://acmeurope-conference.acm.org

DOI:10.1145/3102112

Vinton G. Cerf

A Brittle and Fragile Future

While this is not intended to be a dystopian rant, I feel strongly motivated to draw attention to the fragile and interdependent future we are creating through the use of

programmable devices and systems. Some of you are, no doubt, rather tired of this theme, but as we equip cyber-physical and virtual systems with programs that animate their functions, it seems inescapable that over time they will become increasingly interdependent and that may produce vulnerabilities and fragilities that will be exploited by inimical parties or will simply create difficult and even unrecoverable failures.

Consider systems that use passwords and two-factor authentication to identify users. It is often advised to have alternative means for authentication: a mobile device, a distinct email account, a phone number, or an alternative means of identification. These kinds of interdependencies can lead to cascade failures where loss of access to one system initiates failures in others until a complex of authentication failures render a user unable to use any of them. Loss or cancellation of an email account or a mobile phone number may have later consequences if users do not remember to revise all accounts dependent on these alternative means of identification. They may discover the oversight just when the alternatives are vitally needed.

Multiple platforms that support common services such as Alexa or Google Assistant may be concurrently invoked, leading to confusion as to which is "in charge" at the moment. The situation is exacerbated when multiple users are interacting with the same set of platforms or when the platforms are distant from one another. Conflicting commands from authorized but uncoordinated parties could easily lead to instability or even damage physical and virtual systems.

An analogy might be apt. Personal computers were designed initially to be exactly that: isolated computers for personal use. But before long, they became valuable avenues to access and use of the Internet. Not much thought had been put into the security of these systems when they were stand-alone devices and viruses and worms were already propagating by Sneakernet via floppy-disk drives. The Internet and its predecessors including bulletin board systems were new vectors through which malware could travel and various attacks could be executed. A great deal of effort had to be expended to improve the resistance of personal computers to various forms of attack and failure.

Concerns for safety, security, privacy, and control must be assuaged by systematic analysis of increasingly complex use scenarios.

Many of the devices that are considered cyber-physical systems may suffer from a similar oversight. Often the designers see them as a single-user device controlled from an application running, for example, in the user's mobile smartphone. What is emerging, however, is a highly connected ecosystem of devices and networks with emergent properties derived from the rich, diverse, and distributed connectivity they exhibit. Concerns for safety, security, privacy, and control must be assuaged by systematic analysis of increasingly complex use scenarios. It might even be argued that these analyses will need to be carried out automatically just to keep up with the non-linear growth in potential use cases and device interactions as the devices proliferate.

The designers of devices that populate the Internet of Things have an ethical responsibility to be attentive to the hazards their interactions may create and the companies that market the devices and their services may ultimately be charged by society with liability for their failures or the abuses they invite. It is not too early to begin thinking about these kinds of problems and how they might be addressed technically, legally, and ethically by the engineers and scientists whose advances make new capabilities possible, but which may have unknown consequences as their use proliferates. C

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

ACM
ON A MISSION TO SOLVE TOMORROW.

Dear Colleague,

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

For over 60 years, ACM has helped computing professionals be their most creative, connect to peers, and see what's next, and inspired them to advance the profession and make a positive impact.

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

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

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

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

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

Sincerely,

Vicki L. Hanson
President
Association for Computing Machinery

Association for Computing Machinery

Advancing Computing as a Science & Profession

SHAPE THE FUTURE OF COMPUTING.
JOIN ACM TODAY.

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

SELECT ONE MEMBERSHIP OPTION

ACM PROFESSIONAL MEMBERSHIP:

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

ACM STUDENT MEMBERSHIP:

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

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

Priority Code: CAPP

Payment Information

Name

ACM Member #

Mailing Address

City/State/Province

ZIP/Postal Code/Country

Email

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

❑ AMEX ❑ VISA/MasterCard ❑ Check/money order

Total Amount Due

Credit Card #

Exp. Date

Signature

Purposes of ACM

ACM is dedicated to:

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

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

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

Satisfaction Guaranteed!

BE CREATIVE. STAY CONNECTED. KEEP INVENTING.

Association for Computing Machinery

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

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

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

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:10.1145/3088245 http://cacm.acm.org/blogs/blog-cacm

'Generation CS' Drives Growth in Enrollments

Undergraduates who understand the importance of computer science have been expanding the CS student cohort for more than a decade.

Generation CS: When Undergraduates Realized They Needed Computing
By Mark Guzdial
http://bit.ly/2qiMahP
March 28, 2017

The new Computing Research Association (CRA) report "Generation CS: Computer Science Undergraduate Enrollments Surge Since 2006" (http://cra.org/data/Generation-CS/) describes the dramatic increase in enrollments in computer science (CS) over the last 11 years, with an especially rapid increase since 2009. Sixty percent of academic units surveyed more than doubled their enrollment in that time. The report describes a new generation of undergraduate students who realize the importance of computing education.

The CRA committee that assembled the report carefully analyzed the data in terms of size of the department (for example, number of tenure-track faculty), type of department (for example, Ph.D.-granting or not), and where it characterizes growth in terms of majors vs. non-majors. The bottom line is reflected in this quote:

The current surge of CS majors is pervasive. Large and small academic units, in public and private institutions, have been affected similarly. Doctoral granting and non-doctoral granting units are affected, though doctoral granting units to date have seen larger increases. While academic units are taking a range of actions to handle the increased enrollment, percentage increases in tenure-track faculty are about 1/10th of the increase in the number of majors.

I found several surprises in the report:

▸ Non-major enrollment is also increasing, and at all levels. One might expect the number of non-CS majors to increase at the intro level, but there are also huge increases at the mid and upper levels of the undergraduate curriculum. For units that track the data, the growth in CS minors is similarly dramatic.

▸ Efforts to diversify computing are failing in the face of the enrollment increase. A recent report from Code.org ("University computer science finally surpasses its 2003 peak!," https://medium.com/anybody-can-learn/university-computer-science-finally-surpasses-its-2003-peak-ecefa4c8d77d) shows that the number of CS graduates has finally surpassed the number from 2003, the peak of CS graduate production. Unfortunately, the number of female CS graduates is even less than in 2003. The evidence in "Generation CS" suggests that there are women in the introductory classes, but we are not retaining them into the mid and upper levels of the undergraduate curriculum. The evidence suggests the percentage of Black/African-American students in CS is declining, while Hispanic/Latino percentage share is increasing. A positive sign is that the departments that report taking actions to increase diversity are more diverse.

▸ Departments are having to tighten their belts in response to the increase in enrollment. Schools are not really helping yet. As you can see in the accompanying table, faculty increases are nowhere near the enrollment increases. CRA is offering to share the data from the report with any department that would like to use this data to argue for resources. Departments are canceling low-enrollment classes, increasing class sizes, using more adjunct faculty, using more undergraduate students as teaching assistants, and using more graduate students as instructors.

Google has funded several efforts to respond to the enrollment increases without sacrificing diversity gains. Chris Stephenson has a blog post describing these efforts (https://research.googleblog.com/2017/02/the-cs-capacity-program-new-tools-and.html), with links to more information. Until we can convince schools to increase resources to departments, developing strategies like these and sharing them are our best chances to manage "Generation CS" without losing ground on our efforts to provide CS education to all students.

Cumulative percent growth of CS majors and instructional faculty since 2006.

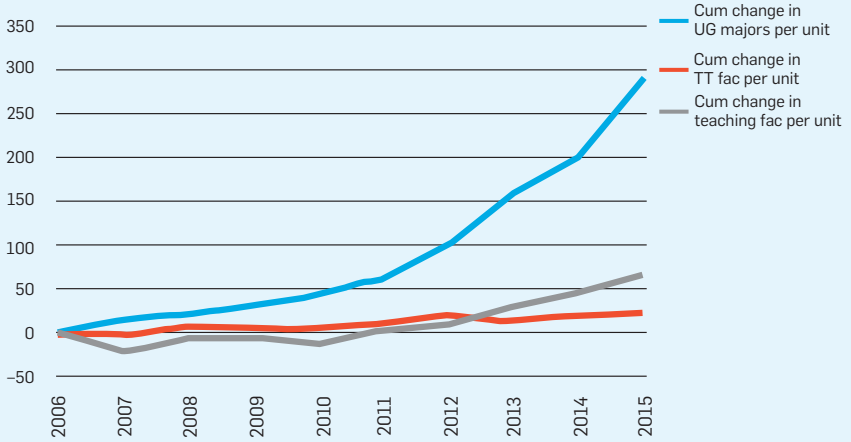


Figure B.4: Cumulative percent growth of CS majors and instructional faculty since 2006.

Source: CRA Taulbee Survey

Comments

If the number of female CS graduates is decreasing, or the number of male CS graduates is decreasing, you cannot assume that the only reason is that universities are not doing enough to recruit them, or encourage them, into the field.

There are many possible reasons outside of the purview of computing, and beyond the competence of the science and practice of computing. Not all of them are even addressable by policies in academia.

There is, however, one factor that is not even mentioned here. There are computing "boot camps" popping up all over the place in the U.S. and outside the U.S. These students are not being counted I'll bet because this article only talks about majors and minors and "diversity" numbers. I personally know several women who have been through these. And the cost is a LOT less in both money and time-to-jobs.

In other words, computing as a practical skill is gaining ground, and private-sector computing (and even many government entities) are interested more in proper results than in credentials.

Academia has had it pretty good with all the federally guaranteed loans pouring into its treasuries after World War II, but the burden has been put upon the over-taxed middle class and even worse on college graduates. Who can blame the victims for seeking alternatives?

The decline after the peak in 2003 mentioned here, we all know why that happened in the U.S. at least. I'll bet it was different in India. Starting with the Y2K projects just before the century digit turned over, the rush to finish and the new Internet infrastructure began the use of remote resources. And there began a decline that continued with some of my coding colleagues training H1B's to replace them. (Illegally but never prosecuted at the time). Philosophically I am a culture-aware libertarian, but hate it when the public is lied to.

It may be that smaller companies that do not have resources to get computing help from overseas are expanding the market for computing skills. It doesn't matter. Getting government out of the picture altogether would remove the warp.

The freak-out about diversity numbers suggests that young girls already know they have no barrier to computing fields and are opting for other studies. I have three daughters who fit this description, and I hate this narrative that stereotypes my daughters based on the warped thinking that something is wrong if females don't make the same choices as males. That is biological nonsense, and studies have shown the differences are innate even from the womb. If it is clear that there are no barriers and that females even get the advantage of extra attention for being a minority in the field, why not take the win and run with it instead of seeking blowback against all the pressure?

—Cassidy Alan

I don't assume that universities are not doing enough to recruit, encourage, or keep women in computing. I know that because there is a large body of computing education research showing that it's true.

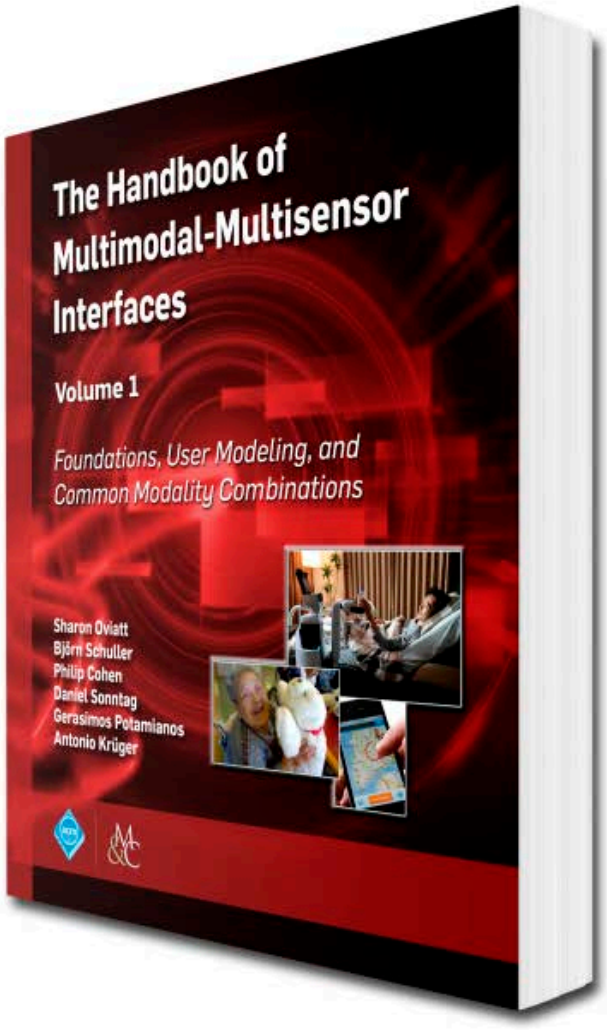
I encourage you to look at the excellent books about the work at Carnegie Mellon University where they successfully have recruited women so that over 40% of their CS class is female. Or, check out the articles on Harvey Mudd College where they are over 50% female. There is a project called BRAID (https://www.hmc.edu/about-hmc/2014/09/24/harvey-mudd-launches-initiative-increase-diversity-computer-science/) to teach other CS departments what Harvey Mudd figured out. It's within the CS departments' control to improve their gender diversity.

The research on "boot camps" is devastating. Many "boot camp" students take repeated boot camps because they don't learn enough CS and can't get jobs.

The reason why few women pursue computing in the U.S. has nothing to do with biology. At Qatar University, computer science is 75% female and computer engineering is 100% female (https://computinged.wordpress.com/2010/10/18/latest-enrollment-numbers-at-qatar-university-big-gender-imbalance/). The gender balance in CS is much more about culture than it is about biology.

I have two daughters myself. Both have tried computer science classes and been quite successful in them. Neither are choosing to get CS degrees. There is nothing wrong with them for pursuing other subjects. As a computing education researcher who studies broadening participation issues, I can list for you all the things that their CS departments did wrong—not recruiting, not encouraging, not keeping women. The problem is with the CS departments, and the data and research studies back me up.

—Mark Guzdial



The FIRST authoritative resource.

EDITED BY
Sharon Oviatt, *Incaa Designs*
Björn Schuller, *Univ. of Passau, Imperial College London*
Philip Cohen, *VoiceBox Technologies*
Daniel Sonntag, *German Research Center for AI*
Gerasimos Potamianos, *University of Thessaly*
Antonio Krüger, *German Research Center for AI*

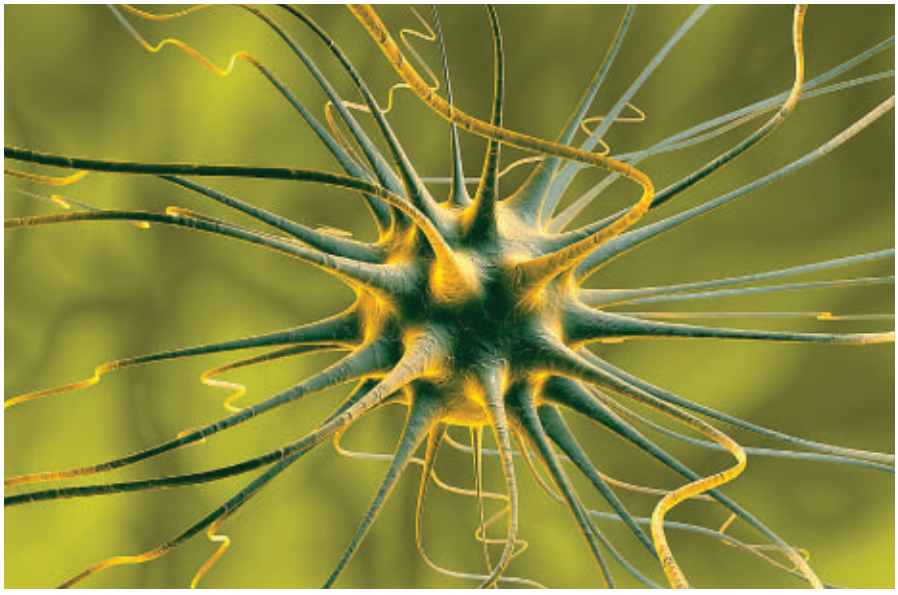
The *Handbook of Multimodal-Multisensor Interfaces, Volume 1* provides the first authoritative resource on what has become the dominant paradigm for new computer interfaces–user input involving new media (speech, multi-touch, gestures, writing) embedded in multimodal-multisensor interfaces. These interfaces support smart phones, wearables, in-vehicle and robotic applications, and many other areas that are now highly competitive commercially. This edited collection is written by international experts and pioneers in the field. It provides a textbook, reference, and technology roadmap for professionals working in this and related areas.

ISBN: 978-1-970001-64-8 DOI: 10.1145/3015783
http://books.acm.org
http://www.morganclaypoolpublishers.com/acm

Science | DOI:10.1145/3088315 Neil Savage

Building a Brain May Mean Going Analog

Analog circuits consume less power per operation than CMOS technologies, and so should prove more efficient.



DIGITAL SUPERCOMPUTING CAN be expensive and energy-hungry, yet still it struggles with problems that the human brain tackles easily, such as understanding speech or viewing a photograph and recognizing what it shows. Even though artificial neural networks that apply deep learning have made much headway over the last few years, some computer scientists think they can do better with systems that even more closely resemble a living brain. Such neuromorphic computing, as this brain emulation is known, might not only accomplish tasks that current computers cannot, it could also lead to a clearer understanding of how human memory and cognition work. Also, if researchers can figure out how to build the machines out of analog circuits, they could run them with a fraction of the energy needed by modern computers.

"The real driver for neuromorphic computing is energy efficiency, and the current design space on CMOS isn't particularly energy efficient," says Mark Stiles, a physicist who is a project leader in the Center for Nanoscale Science and Technology at the U.S. National Institutes for Standards and Technology (NIST) in Gaithersburg, MD. Analog circuits consume less power per operation than existing complementary metal oxide semiconductor (CMOS) technologies, and so should prove more efficient. On the other hand, analog circuits are vulnerable to noise, and the technologies for building them are not as advanced as those for CMOS chips.

One group working on such analog components is Hideo Ohno's Laboratory for Nanoelectronics and Spintronics at Tohoku University in Sendai, Japan. They have built a device that could work as an artificial synapse by relying on spintronics, a quantum property of electrons that gives rise to magnetism. Their device consists of a strip of cobalt/nickel, which is ferromagnetic, meaning that its spins are all aligned. They cross it with a strip of platinum manganese, which is antiferromagnetic, so spins in successive

IMAGE BY KATERYNA KON

atomic layers of the material are perpendicular to each other. Applying a current across the antiferromagnetic layer affects its spins, which in turn applies torque to the spins in the magnetic layer, switching the magnetization from up to down. Unlike in a digital system, the switching is not limited to a 0 or a 1, but can be some fraction that depends on how much current is applied.

A reading current, lower than the switching current, does not switch the magnetization, but its voltage depends on the level of magnetization. Sending multiple signals causes the spin device to adjust its own resistance, strengthening or weakening the connections between neurons, just as synapses in the brain do when they are forming memories, a process brain scientists call plasticity. "The more signals we send in this learning process—the more times we increase or decrease these weights depending on what we want it to understand—the better chance we have of it remembering it when we really want it to," says William Borders, a Ph.D. student in Ohno's lab who works on the project. Train the system to associate one pattern of magnetization with the letter C and another with the letter T, he said, and it will be able to recognize those letters again by measuring resistance.

Whereas Ohno's group is simulating individual synapses, Stiles' group is using spintronics to model collections of neurons and synapses. Their device consists of two ferromagnetic layers separated by a spacer. When a current flows across the junction, it creates a torque on the spins in the material, which sets up a nonlinear oscillation of the magnetization, which in turn creates an oscillation in voltage. Living neurons also behave as nonlinear oscillators, sending out electrical spikes and synchronizing with each other. The spintronic oscillators, therefore, emulate the activity in the brain.

Another approach to building analog circuits with synapses uses memristors, a recently discovered fourth fundamental circuit element, in which flowing current alters resistance, providing the device with memory. Dmitri Strukov, a professor of electrical and computer engineering at the University of California, Santa Barbara, is using memristors to create a multilayer-perceptron network with a simplified version of a synapse's functionality. "Biological networks are much more complicated, in the sense that it is known for example that information is encoded in timing of the spikes, and the shapes of the spikes matter. Here we neglect all of that," he says.

Memristors are promising because it should be possible to make them very small, and they can be stacked, allowing for the high density a neuromorphic system would require.

In the learning part of the process, a large voltage tunes the state of the memristors. The inference part, such as trying to match a new visual pattern with a learned one, uses a smaller voltage. Memristors are promising, Strukov says, because it should be possible to make them very small, and they can be easily stacked, allowing for the high density a neuromorphic system will require. On the other hand, the process for making them has not been refined to the point where it can reliably produce billions of working devices.

Strukov is also using flash memory for synapses. Because it is based on CMOS technology, it works reliably, but it would be hard to shrink the devices or to pack them more densely, so scaling up to a high-density network with billions of devices could be challenging.

A different way to build synapses relies on Josephson junctions, made by separating two pieces of superconducting material with a thin layer of an insulating material. Stephen Russek, a physicist at NIST in Boulder, CO, puts magnetic nanoparticles of manganese in the insulating layer of amorphous silicon between two layers of niobium. Applying a pulse current to the system changes the magnetic structure in the insulating layer—the equivalent of synaptic memory—which affects how current flows from one superconductor to the other. At a certain point, the superconductors start sending out voltage spikes, just as neurons in the brain do. "It's very analogous to what happens in a real neural system where you have a synapse, which is basically a gap between a dendrite and an axon, and that nanostructure in that gap determines the coupling and the spiking probability of that membrane," Russek says.

The advantage of this approach over both CMOS and spintronics, Russek argues, is that the Josephson junction is a naturally spiking element, so it is already more similar to the brain than other devices. The NIST system also resembles a living brain in that signaling activity continues even when the machine is in a resting state. "Just like the brain, you don't turn it off. It's always doing stuff," Russek says. He'd like to emulate that continuous activity, which may play a role in creativity. "We want it because that's what the brain does and we think that's an important part of learning."

Karlheinz Meier, a physicist who headed up BrainScaleS (Brain-inspired multiscale computation in neuromorphic hybrid systems), a project that, until its end in 2015, aimed to develop a mixed-signal neuromorphic system as part of the European Union's Human Brain Project, says neuromorphic computing will have to incorporate such random activity, and also come to grips with analog circuits' susceptibility to noise. "We have to learn how to do what I call 'dirty computing,' how you live with non-perfect components," he says.

So far, these analog systems are very small. Tohoku University, for instance, built a chip with 36 spintronic synapses. Strukov's most recent memristor device holds the equivalent of approximately 400 synapses, while his flash-based system has 100,000. By contrast, the TrueNorth chip, IBM's digital implementation of neural computing, simulates 100 million spiking neurons and 256 million synapses and consumes just 70mW, though Strukov argues that his flash chip is more efficient, with energy use and latency three orders of magnitude better than TrueNorth, when the IBM chip is configured to perform the same task. Even True-

North is not close to the brain, which has about 100 billion neurons connected by perhaps 100 trillion synapses, operating on about 20W.

Going digital was important for TrueNorth, says Dharmendra Modha, chief scientist for IBM's Brain Inspired Research group, which developed the chip. That's because the learning that shapes the neural network is done on a separate system, and the network is then transferred to the chip where it can be run with low energy requirements. Doing it that way, Modha says, requires a digital approach because that helps guarantee one-to-one equivalence between the software and the hardware, which is harder to achieve with analog circuits that are prone to signal leakage.

In the short term, the digital approach allowed IBM to build a neuromorphic chip in the time-frame of a government-funded project, says Modha. The project allowed the company to experiment with new architectures that could yield practical applications soon, and may also lead the way to future systems that use analog circuits in materials other than silicon. "Our view is not analog versus digital," Modha says. Instead, the company is pursuing various architectures, materials, and approaches to see which pan out and which prove useful on the way to the ultimate goal of a brain-like computer.

Similarly, the digital Spiking Neural Network Architecture (SpiNNaker) chip, a massively parallel neural network completed last year at the U.K.'s University of Manchester (UManchester) as part of the Human Brain Project, may find commercial applications for neuromorphic computing in a relatively short time. It is much more flexible than an analog set-up, where the network is configured in hardware, or TrueNorth, in which training of the network takes place off-chip, says Steve Furber, ICL Professor of Computer Engineering at the School of Computer Science at UManchester, who runs the project. "If you have a slightly wacky idea for a potential application for a spiking network, then the easiest machine to prototype it on would be SpiNNaker. When you knew exactly what you wanted, you'd probably want to go and reengineer it, either in this very efficient TrueNorth digital form, or possibly in an analog form," he says.

Rick Stevens, professor of computing at the University of Chicago, says analog neuromorphic computing is still in its early days. "We don't have any neuromorphic hardware that's sufficiently interesting to make the case that it's a plausible computing platform," he says. "If you're trying to do serious deep learning work, you're going be running not on analog hardware." Still, he says, it is worth pursuing, in the same way another far-off technology, quantum computing, is.

Russek agrees an analog neuromorphic computer with an equivalent neuron count to a dog or a monkey brain is still a good 25 years away, depending on how non-silicon technologies develop, but he believes the field has the potential to change computing. "Mother Nature is a good model," he says. "She took five billion years. Hopefully we'll take less." C

Further Reading

Borders, W.A., Akimai, H., Fukami, S., Morya, S., Kurihara, S., Horio, Y., Sat, S., and Ohno, H.
Analogue spin–orbit torque device for artificial-neural-network-based associative memory operation, *Applied Physics Express*, 10, 2016
http://iopscience.iop.org/article/10.7567/APEX.10.013007

Russek, S.E., Donnelly, C.A., Schneider, M.L., Baek, B., Pufall, M.R., Rippard, W.H., Hopkins, P.F., Dresselhaus, P.D., and Benz, S.P.
Stochastic Single Flux Quantum Neuromorphic Computing using Magnetically Tunable Josephson Junctions, IEEE Int'l Conf. on Rebooting Computing, October 2016, San Diego, CA
http://ieeexplore.ieee.org/document/7738712/

Prezioso, M., Merrikh-Bayat, F., Hoskins, B.D., Adam, G.C., Likharev, K.K., and Strukov, D.B.
Training and operation of an integrated neuromorphic network based on metal-oxide memristors, *Nature*, 512, May 2015
http://www.nature.com/nature/journal/v521/n7550/full/nature14441.html

Grollier, J., Querlioz, D., and Stiles, M.D.
Spintronic nano-devices for bio-inspired computing, *Proceedings of the IEEE*, 104, October 2016
http://ieeexplore.ieee.org/document/7563364/

Stanford engineer creates circuit board that mimics the human brain
https://www.youtube.com/watch?v=D3T1tiVcRDs

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

ACM Member News

HELPING CS STUDY CROSS DISCIPLINES

"My deep interest is bringing computer science to everyone," says Carla Brodley, dean of the College of Computer and Information Science at Northeastern University. "I think interesting research problems arise from putting different fields together," reinforcing that interdisciplinary studies have been a constant theme for her.

After Brodley switched majors several times during her undergraduate years at McGill University, a roommate suggested she try taking some computer science courses. "I wrote Newton's method in Fortran as my second assignment, and I fell in love," she recalls. When she asked her mother for advice about which major to pursue, her mother asked if there anything that made her forget to eat. "Yes, when I am coding," she replied, and decided to become a computer science major. Brodley v went on to earn her bachelor's degree in mathematics and computer science at McGill, and then pursued her M.S. and Ph.D. degrees, both in computer science, from the University of Massachusetts at Amherst.

After graduating, Brodley spent 10 years teaching in the School of Electrical Engineering at Purdue University, before departing for Tufts University, where she was a professor in the Department of Computer Science and the Clinical and Translational Science Institute, working primarily on predictive medicine. She joined Northeastern as Dean in August 2014.

Initiatives she is working on at Northeastern include a program called Meaningful Minors, where programs for students minoring in computer science are customized to complement their majors. She is also working to create combined majors relevant to computer science, such as Criminal Justice and Cybersecurity.

—John Delaney

Technology | DOI:10.1145/3088493 Samuel Greengard

Cracking the Code on DNA Storage

Researchers are tapping DNA to create a new and different type of storage media. The technology could prove revolutionary.

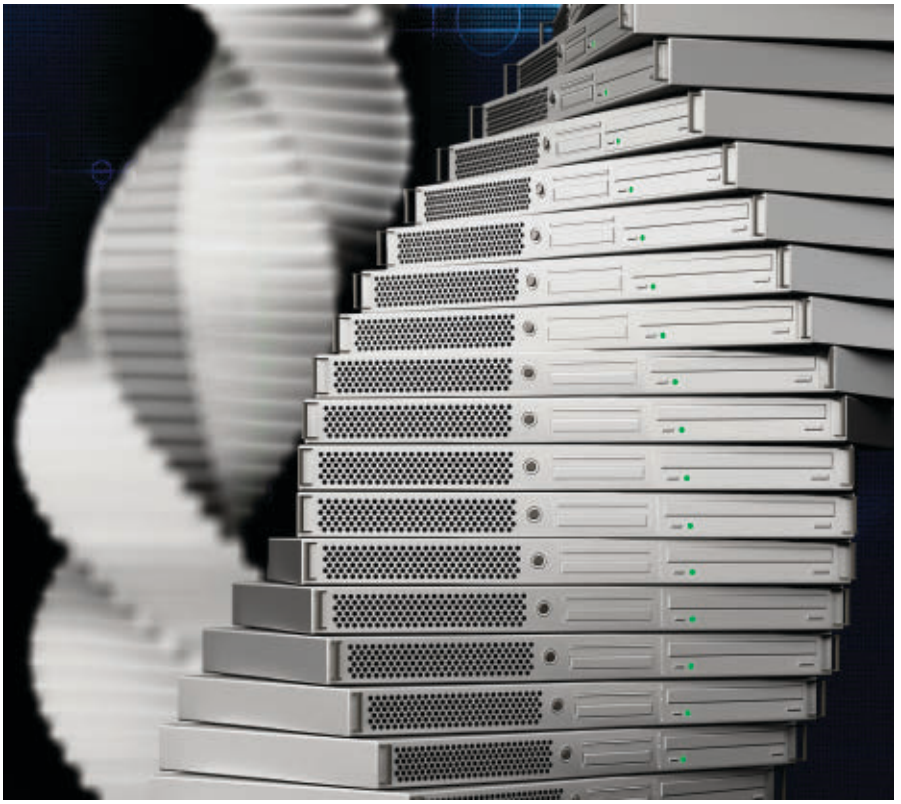
One of the remarkable ironies of digital technology is that every step forward creates new challenges for storing and managing data. In the analog world, a piece of paper or a photograph never becomes obsolete, but it deteriorates and eventually disintegrates. In the digital world, bits and bytes theoretically last forever, but the underlying media—floppy disks, tapes, and drive formats, as well as the codecs used to play audio and video files—become obsolete, usually within a few decades. Once the machine or media is outdated, it is difficult, if not impossible, to access, retrieve, or view the file.

"Digital obsolescence is a very real problem," observes Yaniv Erlich, assistant professor of computer science at Columbia University and a core member of the New York Genome Center. "There is a constant need to migrate to new technologies that don't always support the old technologies."

The challenges don't stop there. Current storage technologies—even state-of-the-art flash drives—require significant space and devour incredible amounts of energy to operate. Ultimately, there's a need to "find a better way to store data" while addressing these and other issues, Erlich says.

As a result, researchers are looking for more efficient ways to store data from books, movies, and myriad other digital file formats. One of the most promising emerging candidates: DNA storage. The technology uses synthetically produced DNA and a "printing" or electrochemical assay process to capture data in strings of synthetically produced genetic code.

Unlike existing media and even other emerging technologies such as holographic and three-dimensional (3D) storage, DNA can withstand huge variations in temperature, along with some variation in moisture. This makes it theoretically possible for the data to last tens of thousands of years, and even to withstand a global disaster.



The scale of the technology also introduces remarkable possibilities. At present, a gram of DNA holds about 730 million megabytes of data. "You could potentially fit a datacenter in DNA material the size of a sugar cube," states Georg Seelig, an associate professor in the department of electrical engineering at the University of Washington. Already, Microsoft Research and Technicolor are eyeing DNA storage, while researchers are inching closer to developing a commercially viable storage technology based on DNA.

Observes George Church, professor of genetics at Harvard Medical School and a pioneer in the field of DNA storage: "It's a fast-moving space full of uncertainty, but there is a great deal of promise in the technology."

Beyond Biology

Although the idea of using DNA to store data extends back to the mid-1960s, it wasn't until 2012 that the concept began to take shape in any tangible way. Then, Church and a team at Harvard University figured out how to convert digital 1s and 0s into long strings on four different nucleotides, also referred to as bases, comprised of As, Gs, Cs, and Ts (for adenine, guanine, cytosine, and thymine). They ultimately encoded 70 billion copies of a 53,400-word book, as well as JPG images and a JavaScript program. The team made multiple copies of all the data to test the accuracy and capacity of the stor-

IMAGE BY CIGDEM

age medium. In the end, they managed to store about six petabits, or about a million gigabytes of data, within each cubic millimeter of DNA. The project demonstrated the validity and potential of the technology.

Over the last few years, various other researchers have worked to advance DNA storage and they have made important breakthroughs. For example, in 2016, a research team from Microsoft and the University of Washington reported that it had written approximately 200 megabytes of data, including *War and Peace* and 99 other literary classics, into DNA. Early this year, Erlich and a research team pushed the boundaries further by encoding six separate files into a single DNA file: a French film (the 49-second *Arrival of a train at La Ciotat*, from 1895, considered the first motion picture in modern history); a complete computer operating system called KolibriOS; a $50 Amazon gift card; a computer virus known as Zip Bomb; the contents of the plaque carried on the *Pioneer* spacecraft; and a research paper. They chose the mix of files because they were more prone to "highly sensitive errors," he says.

What was remarkable about Erlich's research was the ability to push closer to the limit of the so-called Shannon Information Capacity—which determines how much information can be fit into a unit of a system. Previously, researchers had been able to place about 0.9 bits per nucleotide; the Columbia University researchers pushed the figure to about 1.6. Altogether, the team achieved a storage density of about 215 petabytes per gram of DNA, while improving the speed to read data and reducing the cost by roughly 60%. "We were able to achieve far greater robustness and reliability within the storage mechanism," he explains.

One of the keys to success was that the team tapped an advanced algorithm that significantly improved error correction. A problem with DNA storage is that the enzymes used for copying data aren't perfect; the DNA deteriorates during the copying process—and errors wind up in the code. "If you think about regular storage devices, such as a hard drive or a flash drive, this is completely unacceptable," Erlich says. "You want to be able to read the file tens of thousands of times." However, after copying the original sample file 25 times and then copying the copy nine more times (essentially reaching a factor of 10 to the power of 15 copies), the team found that while the results were noisier and more error-prone, they could use an error correction method to view the data accurately.

The key technical challenge is that not all DNA sequences are created equal, and the efficiency and accuracy of the coding process can vary based on the sequencing pattern.

The group conducted one last experiment that further supported the viability of the technology. Researchers took the DNA molecules that contained the data and used a dilution method to damage the DNA sample. The result? Using the algorithm, which works like a Sudoku puzzle by tapping a few cells to create hints about the content of the file without actually sending the file, "We could still perfectly retrieve the information, and we showed that we can get to an information density of 215 petabytes per 1 gram of DNA. To the best of our knowledge, this is probably the densest human-made storage device ever created."

Forward Thinking

Although researchers are pushing DNA storage forward, numerous challenges remain. The biggest obstacle right now is the price tag, which works out to about to about $3,500 per megabyte of storage. Even with researchers at Columbia University reducing DNA storage costs by about 60%, the figure would probably have to reach about a 600% improvement to tip the scale to a commercially viable solution.

"Right now, cost is an enormous roadblock," states Sri Kosuri, an assistant professor of chemistry and biochemistry at the University of California, Los Angeles (UCLA), and a member of Church's 2012 team. Currently, Kosuri notes, most DNA research and funding are focused on genome and biomedical research, which has enormous and growing commercial viability. Yet, "There's no reason the price couldn't drop. It's a question of scaling and advancing the technology to make it viable."

The key technical challenge, which factors directly into the cost issue, is that not all DNA sequences are created equal, and the efficiency and accuracy of the encoding process can vary based on the sequencing pattern. For example, DNA sequences with high GC content or long homopolymer runs (such as AAAAAA... coding) are difficult to synthesize and more prone to sequencing errors. At present, this requires rewriting the DNA code in a different way. One way for researchers to potentially reduce latency and costs that might result from this problem is to produce DNA material faster, but at lower quality. Seelig describes this as "quantity over quality."

Already, algorithms such as the one Erlich used can sidestep many potential problems through error correction. In the future, better DNA printing systems, along with improved error-correction algorithms, will almost certainly drive down the cost further. Ultimately, "The more information you can store in fewer copies of a DNA molecule without having to add much logic redundancy, the better your storage becomes," Seelig points out. In the future, this could also mean that the capacity of a gram of DNA could be reduced from 200 petabytes of material to, say, two petabytes per gram. "At that point, it becomes an economical feasible technology," Erlich says.

Larger questions also loom about how the technology might play out in a practical sense. Reinhard Heckel, a postdoctoral researcher at U.C. Berkeley, says current DNA storage appears to be limited to archival solutions. "Because it doesn't look like it will write or read very fast, it's more likely to serve as a form of archival storage," he says. For the foreseeable future, "It will probably be best suited for information that must be stored for decades and for preserving the important information of the world, sort of like a global seed vault." There are

also questions about whether future computing frameworks—which could delve into organic or molecular models—would be fully compatible with DNA storage systems.

Still, the technology is marching forward. DNA storage could introduce new and intriguing possibilities over the next decade, including radical advances in cloud storage and far more versatile ways to store data, experts say. "Microsoft and other companies are interested in this space simply because the volume of data is growing exponentially. There's a need to develop better ways to store massive amounts of data, particularly over long time periods," Seelig says.

The use of DNA to store data might also unleash new and radically different possibilities that extend beyond conventional storage arrays. This includes loading data into food and medicine, and possibly combining DNA storage with molecular computing and synthetic biology. Because synthetic DNA molecules are extraordinary small, some intriguing possibilities emerge through combining DNA storage with computational systems.

In fact, no one is exactly sure where the research in DNA storage technology will lead. Concludes Seelig: "The intersection of molecular science and computing models is remarkable because DNA molecules can be programmed to act like simple computers that can sense and respond to their environment. They also open up the possibility of quickly and efficiently searching for information stored in a vast DNA archive." C

Further Reading

Bornholt, J., Ceze, L., Carmean, D.M., Lopez, R., Seelig, G., and Strauss, K.
A DNA-Based Archival Storage System, *ASPLOS 2016*, April 1, 2016. https://www.microsoft.com/en-us/research/publication/dna-based-archival-storage-system/.

Blawat, M., Gaedki, K., Hütter, I., Chen, X.M., Turczyk, B., Inverso, S., Pruitt, B.W., and Church, G.M.
Forward Error Correction for DNA Data Storage, *Procedia Computer Science*, Volume 80, 2016, Pages 1011–1022.

Zhirnov, V., Zadegan, R.M., Sandhu, G.S., Church, G.M., and Hughes, W.L.
Nucleic acid memory, *Nature Materials*, Vol. 15, April 2016. www.nature.com/naturematerials.

Erlich, Y. and Zielinski, D.
DNA Fountain enables a robust and efficient storage architecture, *Science*, 03 Mar 2017: Vol. 355, Issue 6328, pp. 950-954. DOI: 10.1126/science.aaj2038. http://science.sciencemag.org/content/355/6328/950.

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

Milestones

NAE's Newest Include 17 Computer Scientists

Seventeen computer scientists were among the 84 new members and 22 new foreign members recently announced by The National Academy of Engineering (NAE).

NAE's newest members in computer science include:

- **Jingsheng Jason Cong, Chancellor's Professor and Director, Center for Domain-Specific Computing, computer science department, at the University of California, Los Angeles, for "pioneering contributions to application-specific programmable logic via innovations in field-programmable gate array synthesis."**
- **Mark David Dankberg, chairman and CEO of ViaSat Inc., for "contributions to broadband Internet communications via satellite."**
- **Whitfield Diffie, adviser, Black Ridge Technology, "for the invention of public key cryptography and for broader contributions to privacy." Diffie shared the 2015 ACM A.M. Turing Award with Martin Hellman for critical contributions to modern cryptography.**
- **Ali H. Dogru, chief technologist and fellow, computational modeling technology, EXPEC ARC, Saudi Aramco/Saudi Arabian Oil Co., "for the development of high-performance computing in hydrocarbon reservoir simulation."**
- **Leonidas J. Guibas, Paul Pigott Professor of Computer Science and Electrical Engineering, computer science department, Stanford University, "for contributions to data structures, algorithm analysis, and computational geometry."**
- **Julia Hirschberg, Percy K. and Vida L.W. Hudson Professor of Computer Science, and chair, department of computer science, Columbia University, "for contributions to the use of prosody in text-to-speech and spoken dialogue systems, and to audio browsing and retrieval."**
- **Dina Katabi, professor, computer science and artificial intelligence laboratory, Massachusetts Institute of Technology, "for contributions to network congestion control and to wireless communications." In 2012, the Katabi was awarded the ACM Grace Murray Hopper Award for contributions to the theory and practice of network congestion control and bandwidth allocation.**
- **Tsu-Jae King Liu, TSMC Distinguished Professor in Microelectronics and chair, department of electrical engineering and computer sciences, University of California, Berkeley, "for contributions to the fin field effect transistor (FinFET) and its application to nanometer complementary metal-oxide-semiconductor (CMOS) technology."**
- **Yann A. LeCun, director, AI Research, Facebook, "for developing convolutional neural networks and their applications in computer vision and other areas of artificial intelligence."**
- **George T. Ligler, consultant, GTL Associates, "for leadership and engineering innovation in specifying and implementing complex computer-based systems for aviation and the U.S. Census."**
- **Steven B. Lipner, executive director, SAFECode, "for developing and deploying practical methods for engineering secure software and computer systems."**
- **George Varghese, Chancellor's Professor, department of computer science, University of California, Los Angeles, "for network algorithmics that make the Internet faster, more secure, and more reliable."**
- **Katherine A. Yelick, associate laboratory director, computer science, Lawrence Berkeley National Laboratory, and professor, electrical engineering and computer science, University of California, Berkeley, "for software innovation and leadership in high-performance computing." Yelick received the ACM-W Athena Award in 2012 and the ACM-IEEE CS Ken Kennedy Award for 2015.**

Computer scientists among the new Foreign Members of NAE include:

- **Chieko Asakawa, chief technology officer for accessibility research and technology, IBM Research Tokyo, "for developing technologies for the visually impaired to access digital information."**
- **Stéphane Mallat, professor, computer science, École Normale Supérieure, "for contributions to the fast wavelet transform and multiresolution signal processing."**
- **Heung-Yeung Shum, executive vice president, technology and research, Microsoft Corp., "for contributions to computer vision and computer graphics, and for leadership in industrial research and product development."**
- **Joseph Sifakis, professor, School of Computer and Communication Science, École Polytechnique Federale de Lausanne, "for co-inventing model checking and for contributions to the development and verification of real-time and embedded systems." He shared the ACM A.M. Turing Award for 2007 with Edmund Clarke and E. Allen Emerson.**

The new members will be inducted in a ceremony on Oct. 8.

—Lawrence M. Fisher

Society | DOI:10.1145/3088342 Gary Anthes

Artificial Intelligence Poised to Ride a New Wave

Flush with recent successes, and pushed by even newer technology, AI systems could get much smarter.

ARTIFICIAL INTELLIGENCE (AI), once described as a technology with permanent potential, has come of age in the past decade. Propelled by massively parallel computer systems, huge datasets, and better algorithms, AI has brought a number of important applications, such as image- and speech-recognition and autonomous vehicle navigation, to near-human levels of performance.

Now, AI experts say, a wave of even newer technology may enable systems to understand and react to the world in ways that traditionally have been seen as the sole province of human beings. These technologies include algorithms that model human intuition and make predictions in the face of incomplete knowledge, systems that learn without being pre-trained with labeled data, systems that transfer knowledge gained in one domain to another, hybrid systems that combine two or more approaches, and more powerful and energy-efficient hardware specialized for AI.

The term "artificial intelligence" was coined by John McCarthy, a math professor at Dartmouth, in 1955 when he—along with Marvin Minsky of the Massachusetts Institute of Technology (MIT), Claude Shannon of Bell Laboratories, and Nathaniel Rochester of IBM—said they would study "the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it." McCarthy wanted to "find out how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves." That is not a bad description of the goals of AI today.

Chinese professional Go player Ke Jie preparing to make a move during the second game of a match against Google's AlphaGo in May 2017.

PHOTO BY STR/AFP/GETTY IMAGES

The path of AI over the ensuing 62 years has been anything but smooth. There were early successes in areas such as mathematical problem solving, natural language, and robotics. Some of the ideas that are central to modern AI, such as those behind neural networks, made conceptual advances early on. Yet funding for AI research, mostly from the U.S. government, ebbed and flowed, and when it ebbed the private sector did not take up the slack. Enthusiasm for AI waned when the grandiose promises by researchers failed to be met.

AI Comes of Age

A turning point for AI, and for the public's perception of the field, occurred in 1997 when IBM's Deep Blue supercomputer beat world champion Garry Kasparov at chess. Deep Blue could evaluate 200 million chess positions per second, an astonishing display of computer power at the time. Indeed, advances in AI over the past 10 years owe more to Moore's Law than to any other factor. Artificial neural networks, which are patterned after the arrangement of neurons in the brain and the connections between them, are at the heart of much of modern AI, and to do a good job on hard problems they require teraflops of processing power and terabytes of training data.

Michael Witbrock, a manager of Cognitive Systems at IBM Research, says about two-thirds of the advances in AI over the past 10 years have come from increases in computer processing power, much of that from the use of graphics processing units (GPUs). About 20% of the gain came from bigger datasets, and 10% from better algorithms, he estimates. That's changing, he says; "Advances in the fundamental algorithms for learning are now the main driver of progress."

Witbrock points, for example, to a technique called reinforcement learning, "systems which can build models that hypothesize about what the world might look like." In re-

inforcement learning, systems are not trained in advance with huge amounts of labeled data—which is called "supervised learning"—but are simply rewarded when they get the right answer. In a sense, they are self-trained. This psychology-based approach more nearly represents the way humans learn.

Reinforcement learning is useful in games such as chess or poker, which have clearly defined objectives for which system developers don't know in advance how to achieve the desired outcome. The machine plays millions of games against itself, and gradually adapts its behavior to strategies that tend to win.

Animals and Humans Do It

According to Yann LeCun, director of AI research at Facebook, supervised learning is the dominant AI method today, while reinforcement learning occupies a niche mostly in games.

A third type of learning, called predictive learning, is emerging. It is unsupervised, meaning it does not need to be trained with millions of human-labeled examples. "This is the main form of learning that animals and humans use," LeCun says. "It allows us to learn by observation." Babies easily learn that when one object moves in front of another, the hidden object is still there, and that when an object is not supported, it falls. No one teaches the baby those things by giving it rules or labeled examples.

"We don't yet quite know how to reproduce this in machines, and that's a shame," LeCun says. "Until we learn how to do this, we will not go to the next level in AI."

Meanwhile, predictive learning has become one of the hottest topics in AI research today.

Predictive learning is alluring because it represents a step toward human ways of thinking, but also because it reduces or eliminates the expensive curation of big databases of labeled data. "You just show the machine thousands of hours of video, and it will eventually figure out how the world works and how it's organized," LeCun says.

Facebook and others are developing ways to predict the occurrence of a word in a block of text based on the known words around it. A popular technique called Word2Vec, proposed by Tomas Mikolov (then at Google and now at Facebook AI Research), learns to represent words and text in the form of vectors, or lists of numbers, that contain the characteristics of a word, such as syntactic role and meaning. Such unsupervised word-embedding methods are widely used in natural language understanding and language translation applications.

Another promising new approach to unsupervised predictive learning lies in something called generative adversarial networks (GAN), in which two neural nets train themselves by competing in a zero-sum game to produce photorealistic images. Originally proposed by Ph.D. candidate Ian Goodfellow (now at OpenAI), GANs are able to learn reusable image feature representations from large sets of unlabeled data. One experimental application can predict the next frame in a video, given some of the frames that precede it. LeCun calls GANs "the most interesting idea in machine learning in the last 10 years."

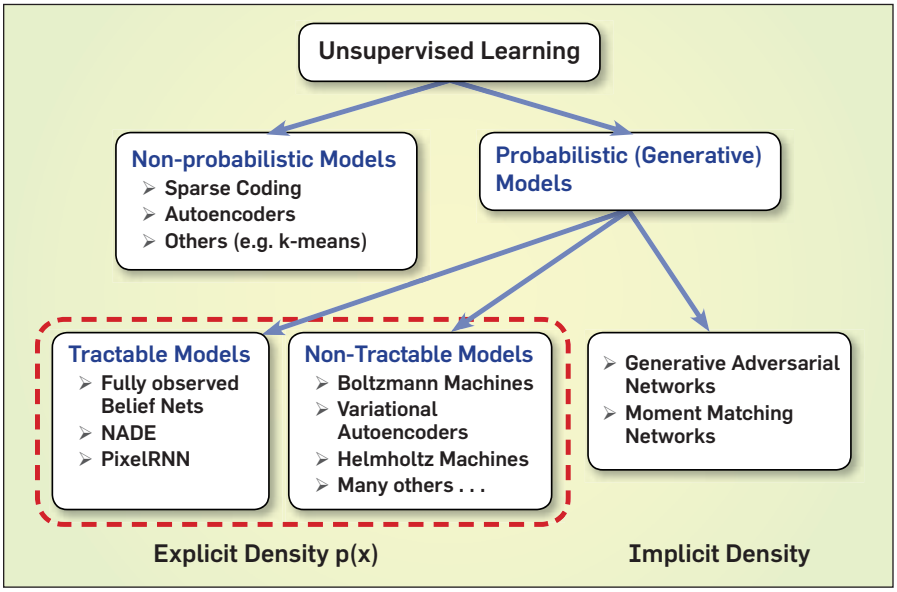
Unsupervised predictive learning systems will be able to show some of the "common sense" that comes so easily to humans, says AI pioneer Geoffrey Hinton, an engineering fellow at Google. For example, in the sentence, "The trophy would not fit in the suitcase because it was too small," humans know immediately that "it" refers to the suitcase, not the trophy. It's important to know that when translating the sentence into French, because the two nouns have different genders in French. "Neural nets can't handle that very well at present," Hinton says, "but they will one day when they are big enough to hold a lot of real-world knowledge."

Last year AlphaGo, developed by Google DeepMind, surprised the world when it became the first AI system to beat a Go professional in a five-game match. Chess programs can evaluate possible moves far ahead in the game, but Go programs are unable to do that because the number of possible moves quickly becomes astronomical. "AlphaGo depends on neural nets having the ability to model intuitive reasoning based on patterns, as opposed to logical reasoning," Hinton says. "That's what Go masters are good at. "

AlphaGo is a hybrid of supervised learning and reinforcement learning. It was initially trained on a database of 30 million moves from historical games; then it was improved, via reinforcement learning, by playing many games against itself. AlphaGo uses one neural net to decide which moves are worth considering in any given position, and another neural net to evaluate the position it would arrive at after a particular sequence of moves. Hinton estimates that AlphaGo uses 10,000 times as much compute power as Deep Blue did 20 years ago.

AI as Pal

Manuela Veloso, head of the Machine Learning Department at Carnegie-



A taxonomy of Unsupervised Learning.

TAXONOMY BY RUSLAN SALAKHUDINOV

Mellon University, predicts that for many years to come AI systems will collaborate with humans, rather than replacing them. "There will be this symbiosis between humans and machines, in the same sense that humans need other humans," she says.

Asked where AI systems are weak today, Veloso says they should be more transparent. "They need to explain themselves: why did they do this, why did they do that, why did they detect this, why did they recommend that? Accountability is absolutely necessary."

IBM's Witbrock echoes the call for humanism in AI. He has three definitions of AI, two of them technical; the third is, "It's an embodiment of a human dream of having a patient, helpful, collaborative kind of companion."

Just such a companion is in the dreams of Facebook engineers. They foresee a day when every Facebook user has his or her own personalized virtual assistant. "The amount of computation required would be enormous," LeCun cautions. "We are counting on progress in hardware to be able to deploy these things." AI developers say progress will come from greater use of specialized processors like GPUs, as well as from entire systems designed specifically for running neural networks. Such systems will be much more powerful and more energy-efficient than those in use today.

Meanwhile, some approaches to AI are so new that even their developers don't know exactly how they work. Google Translate learned to translate among 103 languages by being trained separately in individual language pairs—English to French, German to Japanese, and so on. But all of these unique pairwise combinations entail large development and processing costs, so Google developed a "transfer-learning" system that can drastically cut costs by applying the knowledge learned in one application to another. Remarkably, it has shown it can translate between pairs of languages it has never encountered before.

Called "zero-shot" translation, the Google system extracts the inherent "meaning" of the input language (which does not have to be specified) and uses that to translate into another language without reference to stored pairs of translated words and phrases. "It suggests there's an 'interlingua' the neural net is learning; a high-level representation of the meaning of sentences, beyond any particular language," Hinton says.

"There will be this symbiosis between humans and machines, in the same sense that humans need other humans."

Yet work remains to be done to validate this hypothesis, to understand the system, and to make it work better. "It's still not quite as good as a really good human translator," Hinton says.

Further Reading

Mathieu, M., Couprie, C., and LeCun, Y.
Deep Multi-Scale Video Prediction Beyond Mean Square Error
ICLR 2016 conference paper, Version 6, Feb. 26, 2016
https://arxiv.org/abs/1511.05440

Radford, A., Metz, L., and Chintala, S.
Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks
Version 2, Jan. 7, 2016
https://arxiv.org/abs/1511.06434

LeCunn, Y.
Unsupervised Learning: the Next Frontier in AI [video]
Nov. 28, 2016
https://www.aices.rwth-aachen.de/charlemagne-distinguished-lecture-series

LeCun, Y., Bengio, Y., and Hinton, G.
Deep Learning
Nature, v.521, p.436-444, May 28, 2016
http://www.nature.com/nature/journal/v521/n7553/abs/nature14539.html

Johnson, M., et al
Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation
Nov. 14, 2016
https://arxiv.org/abs/1611.04558

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

Milestones

AAAS Adds New Members

The American Academy of Arts and Sciences (AAAS) recently expanded its membership with the addition of 228 new members, including seven computer scientists.

"It is an honor to welcome this new class of exceptional women and men as part of our distinguished membership," said Don Randel, chair of the Academy's Board of Directors. "Their talents and expertise will enrich the life of the Academy and strengthen our capacity to spread knowledge and understanding in service to the nation."

"In a tradition reaching back to the earliest days of our nation, the honor of election to the American Academy is also a call to service," said Academy president Jonathan F. Fanton. "Through our projects, publications, and events, the Academy provides members with opportunities to make common cause and produce the useful knowledge for which the Academy's 1780 charter calls."

Computer scientists in the new class include:

- **Thomas E. Anderson, University of Washington**
- **Hari Balakrishnan, Massachusetts Institute of Technology**
- **John C. Mitchell, Stanford University**
- **Daniela Rus, Massachusetts Institute of Technology**
- **Fred B. Schneider, Cornell University**
- **Katherine A. Yelick, University of California Berkeley/Lawrence Berkeley National Laboratory**

Also to be inducted as a Foreign Honorary Member is computer scientist Giovanni De Micheli of École Polytechnique Fédérale de Lausanne.

The new class will be inducted at a ceremony on Oct. 7 in Cambridge, MA.

The American Academy of Arts and Sciences is one of the country's oldest learned societies and independent policy research centers, convening leaders from the academic, business, and government sectors to respond to the challenges facing the nation and the world.

In Memoriam | DOI:10.1145/3099559 Lawrence M. Fisher

Jean E. Sammet 1928–2017

Jean E. Sammet, an American computer scientist who served as the first female president of ACM, passed away on May 21 at the age of 89.

Sammet was born March 23, 1928 in New York City. She received a bachelor's degree in mathematics from Mount Holyoke College in 1948, and a master's degree in that discipline from the University of Illinois at Urbana-Champaign in 1949.

In 1951, Sammet joined Metropolitan Life Insurance, but left to pursue a doctorate in mathematics at Columbia University. She worked as a teaching assistant at Barnard College for a year, before deciding to leave academia. (She did not complete doctoral studies, but received an honorary doctor of science degree from Mount Holyoke in 1978).

From 1953 to 1958, Sammet was a mathematician for Sperry Gyroscope (now part of Unisys) in New York, working on mathematical analysis problems.

In 1955, the company asked her to program the Sperry Electronic Digital Automatic Computer. Sammet became leader of an "open shop" of programmers acting as consultants to engineers and working with scientists who assisted them in writing and testing routines.

Sylvania Electric Products hired Sammet in 1958 to oversee software development for the U.S. Army's Mobile Digital Computer. She served as staff consultant for programming research, and was a member of the original COBOL group.

In 1961, she moved to IBM, where she researched the use of restricted English as a programming language and the use of natural language for mathematical programs.

During the mid-1960s, Sammet developed FORMAC (Formula Manipulation Compiler), the first widely used computer language for symbolic manipulation of mathematical formulas.

ACM Activities

Understanding the importance of exchanging information with others

working with languages and software, Sammet contacted ACM's then-president George Forsythe, who named her chairperson of the Special Interest Committee on Symbolic and Algebraic Manipulation (SICSAM, later the Special Interest Group on Symbolic and Algebraic Manipulation, SIGSAM).

In 1966, Sammet was elected ACM's Northeast Regional Representative. She was also a member of the ACM Council, and was named ACM lecturer in 1967, 1968, and 1972. In 1968, Sammet was named chairperson of the ACM Committee on SIGs and SICs.

In 1971, she was elected chair of the ACM Special Interest Group on Programming Languages (SIGPLAN), a post she resigned when elected vice president of ACM in 1972.

While vice president, ACM'S then-president Anthony Ralston asked her to chair the first ACM Long Range Planning Committee (LRPC). In 1974, Sammet was elected the first female president of ACM, and continued to work with the LRPC. A final report from the committee recommended principles to be used as guidelines for ACM activities.

In 1975, Sammet was named an honorary member of Upsilon Pi Epsilon, the International Honor Society for the Computing and Information Sciences.

In 1977, she organized and chaired the first History of Computing Committee for the American Federation of Information Processing Societies, which encouraged the creation of archives for industry professionals' materials. Said Sammet, "From childhood on, I hated to throw papers away. As I became an adult, this characteristic merged with my interest in computing history. As a result, I created important files and documents of my own, and became concerned with having other people publish material on their important work so the facts (rather that the myths) would be known publicly."

ACM celebrated Sammet in 1985 with its Distinguished Service Award "for advancing the art and science of computer programming languages and recording its history. "

Sammet retired from IBM in 1988. She served on the board of directors of the Computer History Museum 1983-1993, and on the board and executive committee of the Software Patent Institute 1991-1998.

In 1989, the Association for Women in Computing presented Sammet its Ada Lovelace Award.

In 1994, Sammet was named a Fellow of the ACM.

In 1997, she shared SIGPLAN's Distinguished Service Award with J.A.N. Lee.

In 2001, Sammet was named a Fellow of the Computer History Museum.

In 2009, IEEE gave her its Computer Pioneer Award "for pioneering work and lifetime achievement as one of the first developers and researchers in programming languages."

In 2013, the National Center for Women & Information Technology (NCWIT) bestowed its Pioneer Award on her.

Sammet maintained an enormous collection of materials on programming languages, which were to be housed at the Charles Babbage Institute Center for the History of Information Technology at the University of Minnesota upon her demise. C

—Lawrence M. Fisher

ACM Books.

In-depth. Innovative. Insightful.

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

For more info please visit
http://books.acm.org
or contact ACM at
ACMbooks-Info@acm.org

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

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

DOI:10.1145/3097347

Adam L. Young and Moti Yung

Privacy and Security

Cryptovirology: The Birth, Neglect, and Explosion of Ransomware

Recent attacks exploiting a known vulnerability continue a downward spiral of ransomware-related incidents.

Cryptovirology was born out of scientific curiosity of what the future may hold for software attacks that merge cryptographic technology with malware. It started at Columbia University as a natural by-product of an unnatural union: a former hacker placed in a room with a cryptographer, both given ample time with which to contemplate the dystopia of tomorrow. Collectively, given our backgrounds, we had amassed a body of highly unconventional scientific problems that hackers face when infiltrating computer systems as well as the foundational cryptography with which to solve those problems.

Our list of problems included the following question: How devastating could the most insidious malicious software attack be against a target? To put things in perspective this was circa 1995. Many people had not heard of the Internet, and among those that did, many were obtaining an email address for the first time. The typical home computer was not online all the time. Users had to use dial-up modems when they wanted to check email. USB technology did not exist. 3.5-inch floppy disks were the norm. Cryptography, for millennia, had been perceived as a purely protective technology, and in particular as a way to hide the content of messages, secure data at rest, and authenticate users.

On the one hand we were aware of the failed AIDS Information Trojan that scrambled the names of the victim's files using a symmetric key and demanded a ransom to unscramble them. From a technological perspective this attack crumbled since the decryption key could be extracted from the code of the Trojan.

In addition, we had in mind the grotesque vision of H.R. Giger in the science-fiction movie *Alien*.[5] Of particular interest to us was the alien facehugger. This creature resembled a cross between an insect and an octopus. It would wrap its legs around the victim's face and insert a tube down the victim's throat. It wrapped its long tail around the victim's neck and squeezed. The victim would enter a form of coma, while the egg the facehugger implanted into the abdomen would incubate into a drone (or queen) and burst through the stomach of the victim, thus completing a phase of the alien life cycle.

There was no way to safely remove the facehugger once attached. Touching the facehugger caused it to tighten its tail and restrict the flow of air to the lungs. Cutting it caused its corrosive alien blood to bleed out and disintegrate everything it seeped through (including the floors of the spaceship). Try as they did the crew's scientists could not find a way to safely remove facehuggers from their victims.

The AIDS Trojan and the facehugger idea defined in our minds the "where we are now" versus where malicious software attacks might evolve

IMAGE BY ALICIA KUBISTA/ANDRIJ BORYS ASSOCIATES

to, respectively. We sought a digital analogue of the facehugger, namely, a forced symbiotic relationship between a computer virus and its host where removing the virus is more damaging than leaving it in place.

But what we discovered was not exactly that which we sought. We discovered the first secure data kidnapping attack. We called it cryptoviral extortion. In cryptoviral extortion, the attacker generates a key pair for a public key cryptosystem and places the "public encryption key" in the cryptovirus. The corresponding "private decryption key" is kept private. The cryptovirus spreads and infects many host systems. It attacks the host system by hybrid encrypting the victim's files: encrypting the files with a locally generated random symmetric key and encrypting that key with the public key. It zeroizes the symmetric key and plaintext and then puts up a ransom note containing the asymmetric ciphertext and a means to contact the attacker. The victim sends the payment and the asymmetric ciphertext to the attacker. The attacker receives the payment, decrypts the asymmetric ciphertext with his private key, and sends the recovered symmetric key to the victim. The victim deciphers his files with the symmetric key.

At no point is the private key revealed to the victims. Only the attacker can decrypt the asymmetric ciphertext. Furthermore, the symmetric key that a victim receives is of no use to other victims since it was randomly generated.

We presented this attack along with the facehugger analogy at the 1996 IEEE Security and Privacy conference.[8] The discovery was perceived as being simultaneously innovative and somewhat vulgar. Years later, the media relabeled the cryptoviral extortion attack as ransomware. In the conference paper we proposed that electronic money could be extorted by the attacker. This is what happens today using bitcoin. We have observed that what we described over 20 years ago is the exact "business model" used today in an estimated $1 billion-a-year criminal industry: the industry of ransomware.

We discovered that public key cryptography holds the power to break the symmetry between the view of an antivirus analyst and the view of the attacker. The view of the antivirus analyst is the malware code and the public key it contains. The view of the attacker is the malware code, the public key it contains, and the corresponding private key. The malware can perform trapdoor one-way operations on the victim's machine that only the attacker can undo. A multitude of cryptovirology attacks, both overt and covert in nature, are based on the unique advantage this gives to the attacker. These methods weaponize cryptography as an attack tool as opposed to the previous uses that were defensive in nature.

In our 2004 book *Malicious Cryptography: Exposing Cryptovirology*[9] we presented the following analogy: cryptovirology is to penetrating computer systems as cryptanalysis is to cracking ciphers. It is a proactive anticipation of the opponent's next move and suggests that certain countermeasures should be developed and put into place. To counter cryptoviral extortion we recommended a diligent backup strategy and searching for crypto code where it

does not belong. We warned the public about these threats and similar ones by publishing our findings, thereby providing a significant head start to develop and deploy defenses.

It has been a long road that we have followed, fraught with skepticism and criticism, ultimately resulting in worldwide recognition that cryptoviral extortion is a severe threat. Over the years we have given numerous lectures on cryptovirology. We have experienced the spectrum of possible reactions. Some concurred that the threat is real. Others insisted that cryptoviral extortion was pointless, that it offered nothing to the attacker beyond deleting the hard drive. Still others professed that no victim would ever pay.

Shortly after we published our book, it was met with harsh criticism. An expert who had written books on computer viruses published a scathing review, concluding that for those seriously involved in the study of malware the book is of "little practical use." This opinion directly translates to telling the public there is no need to worry about ransomware. We attributed such reactions to the inherent resistance many people feel toward new ideas, especially ideas that merge two previously distinct disciplines, in this case, malware and cryptography. It seemed to us that the difficulties known as the "innovator's dilemma"[2] apply also to proactively addressing threats and risks.

Cryptovirology has proven itself to be a formidable threat. Ransomware attacks make the news daily. Victims include individuals, hospitals, police precincts, universities, transportation systems, and government offices. We even saw the development of "ransomware as a service" where cryptovirology tools are sold to criminals that perpetrate cryptoviral extortion (for more details on ransomware, see https://en.wikipedia.org/wiki/Ransomware). This past year we have witnessed a vicious downward spiral: the more organizations that were attacked, the more news coverage there was on ransomware. The more news coverage there was on ransomware, the more criminals got in on the action, prompting ever more news coverage. The media amplified cryptovirology awareness among law-abiding citizens and criminals alike.

Cryptovirology has proven itself to be a formidable threat.

Social and legal reactions to the damage followed. In fact, the trip further down the spiral changed the very definition of a "computer breach." Prior, a computer breach was synonymous with the exfiltration of sensitive data from an organization. This past year the meaning expanded to account for ransomware. A recent fact sheet published by the U.S. Department of Health and Human Services on ransomware and HIPAA states that when electronically protected health information is encrypted by ransomware a breach has occurred and the incident therefore constitutes a disclosure that violates HIPAA.[6] The justification for this definition is that the adversary has taken control of sensitive health information. This is a significant change in the definition of a computer "breach" since now, due to the threat of cryptoviral extortion, a breach can occur even when no sensitive data is exfiltrated!

A highly publicized and effective ransomware attack was carried out against the Hollywood Presbyterian Medical Center, and the hospital paid $17,000 in bitcoin for restoration. This, along with the epidemic levels of similar attacks, prompted the state of California to enact a new law that addresses ransomware.[1] "SB-1137 Computer crimes: ransomware" amends Section 523 of the Penal Code to outlaw the introduction of ransomware into a computer system with the intent of extorting money. Reuters reported that the WannaCry cryptoworm from May 2017 locked up more than 200,000 computers in more than 150 countries.[7] The attack exploited a vulnerability hoarded by the NSA that was exposed by whistle-blowers and later patched. The attack was nonetheless severe since organizations and individuals were not diligent enough in patching.

We finally point out that cryptovirology has influenced popular culture as well, inspiring the plot in Barry Eisler's techno-thriller *Fault Line*.[3]

Over the years we have observed a palpable reluctance by security companies to describe the cryptoviral extortion attack in detail and discuss countermeasures. We view this as being fundamentally flawed; it is the classic phenomenon of "reactive security" (acting after the attack) as opposed to the preventative "proactive security."

We believe ransomware is the tip of the iceberg. Most cryptovirology attacks are covert in nature, allowing the adversary to securely steal information completely unnoticed. These attacks would slip past or stymie the vast majority of computer incident response teams. It took over 20 years for cryptoviral extortion to gain worldwide recognition, and it appears that the bulk of these other attacks, which are fully described in the scientific literature, are heading in the same direction: destined to be overlooked until a large-scale real-world attack is publicized. Santayana's aphorism: "those who cannot remember the past are condemned to repeat it"[4] seems to apply equally well to malicious cryptography. C

References

1. Barth, B. California ransomware bill supported by Hollywood hospital passes committee. *SC Magazine* (Apr. 13, 2016).
2. Christensen, C. *The Innovator's Solution: Creating and Sustaining Successful Growth.* Harvard Business School Press, 2003.
3. Eisler, B. *Fault Line.* Ballantine Books, 2009.
4. Santayana, G. *Reason in Common Sense*, (1905), p. 284, volume 1 of *The Life of Reason.*
5. Scott, R. *Alien.* 20th Century Fox, 1979.
6. U.S. Dept. of Health and Human Services. FACT SHEET: Ransomware and HIPAA; http://bit.ly/29zm57B
7. Volz, D. and Auchard, E. More disruptions feared from cyber attack; Microsoft slams government secrecy. Reuters (May 15, 2017).
8. Young, A. and Yung, M. Cryptovirology: Extortion-based security threats and countermeasures. In *Proceedings of the IEEE Symposium on Security and Privacy*, (1996), 129–140.
9. Young, A. and Yung, M. *Malicious cryptography—Exposing cryptovirology.* Wiley, 2004.

Adam L. Young (ayoung235@gmail.com) is a researcher at Cryptovirology Labs.

Moti Yung (motiyung@gmail.com) is a Security and Privacy Scientist, Snap Inc., and Adjunct Senior Researcher, Computer Science Department, Columbia University.

DOI:10.1145/3097349

Brad Greenwood, Gordon Burtch, and Seth Carnahan

Economic and Business Dimensions

Unknowns of the Gig-Economy

Seeking multidisciplinary research into the rapidly evolving gig-economy.

ALTHOUGH GIG-ECONOMY PLATFORMS like Uber, Postmates, and ThumbTack have captured the attention of policymakers and practitioners, research has only begun to tackle a $26B phenomenon[17] that is estimated to grow dramatically in the coming years. Gig-economy platforms, defined as digital, service based, on-demand platforms that enable flexible work arrangements[19] are a unique recent addition to the broader category of peer-to-peer platform business-models, which previously comprised intermediaries facilitating the exchange of tangible goods (for example, Etsy, eBay, or Alibaba).

New phenomena come with new research questions. To date, such research has examined a variety of important topics, including consumer surplus,[14,16] drunk driving,[12] disruption of industry incumbents,[2,20] entrepreneurial activity,[3] and the evolving nature of the employer-employee relationship.[9,10] Importantly, authors of these works have been careful not to cast changes brought by the gig-economy in beneficial or pejorative terms, working instead towards the collection of a broad base of objective empirical facts that can inform policy. However, the collection of findings amassed thus far has arisen from disparate academic disciplines, conducting work across different levels of analysis, such as markets, firms, and individuals, which hampers their integration toward unified conclusions, and limits the identification of the most promising avenues for future work. Considering the critical implications for individuals, firms, and markets, we call for multidisciplinary research at each of these levels to both accelerate our understanding of the gig-economy, and to inform legislators of the potential benefits and pitfalls of the gig-economy (thereby facilitating the creation of effective policy). We anticipate that *Communications* readers will be heavily involved in research relating to the effective design and functioning of gig-economy markets, which will subsequently inform significant research addressing firm- and individual-level effects. We

PHOTO BY STEVE RHODES

anticipate such effects will be of interest to colleagues in the social sciences, namely scholars of management, economics, and sociology.

Markets

While research has begun to examine the effective design of gig economy markets, as well as the degree to which different industries will be affected by platform emergence, numerous questions remain. Perhaps the most intriguing and important questions relate to improvements in algorithm and market design. Algorithms help to match supply with demand on these platforms, and algorithms also help determine prices. While researchers like Chen and Sheldon[4] highlight the effectiveness of current algorithms, others note the possibilities for more efficient outcomes from improved algorithms.[11] What challenges will arise as gig-economy platforms expand and must integrate with existing markets and infrastructure? In addition to efficiency considerations, researchers need to consider the distributional consequences of algorithm and market design. For example, while providing platform participants with one another's names and photos allows for more efficient and pleasant interactions between buyers and sellers, it also creates the conditions that enable discrimination.[9] What steps can algorithm and market designers take to provide platform participants with enough information about each other while simultaneously mitigating discriminatory behavior?

Further, anecdotal accounts describe the spread of gig-economy models across different industries and geographies. Yet, we know little about which industries and locations are most likely to be affected. What impact might the emergence of P2P sharing platforms have on sales of durable goods in different industries?[1] How will differences in labor laws, regulations, and economic opportunity affect the geographic expansion of gig-economy platforms?[14] Uber, for example, moved its self-driving car operations to Arizona after changes in California law.[a] Legal scholars are also beginning to examine optimal regulatory policies for gig-economy markets,[6] with the debate largely centered on the appropriate administrative unit (such as local, state, national, transnational) for regulating these markets, and with some scholars advocating platform self-regulation.[6] Will regulation and algorithm design become increasingly linked over time? Should regulators police the contents of algorithms as a method for monitoring gig-economy platforms? Addressing these questions requires the combined talents of computer scientists and legal scholars.

Firms

Several important questions arise when we consider the firm-level management challenges posed by growth of the gig-economy.[7,8,13] At the broadest level, it is unclear why gig-economy innovations were primarily developed by startup entrants (including Uber, Airbnb, Prosper), rather than established players (suc as Yellow Cab, Marriott, Bank of America). Were established players simply unaware of the gig-economy approach to organizing, or did they evaluate and reject the idea? Going forward, which incumbents will make the transition to a platform business model,[7,22] that is, from owning resources to renting them, and what will determine their success? Gig-economy platforms also face interesting and unique strategic challenges, such as the management of labor. As is well known, these platforms minimize costs by employing independent contractors. Yet, this gives rise to a host of challenges, such as ensuring there is sufficient labor to meet demand and extending the number of hours laborers work. To date, practice has focused on psychological manipulations of labor in order to extend working hours,[18] which should continue to be of broad interest to behavioral economists and psychologists as means not only of affecting labor supply, but demand as well.[21] Further, researchers should extend this line of work to consider the broader design of incentives structures. The classification of labor as independent contractors similarly poses challenges, inasmuch as gig-economy firms find themselves increasingly buffeted by labor-oriented lawsuits. How do firms manage these legal challenges, and why might some firms decide to convert independent contractors into employees? How do managers decide what level of risk the firm should absorb, versus what level of risk the independent contractor should absorb? As the decision to adopt a gig-economy business model is analogous to outsourcing part of the firm's human and/or physical capital, many of these firm-level questions relate to the classic line of inquiry about how managers demarcate the boundary between their firm and the market.

Gig-economy platforms also have interesting and unique strategic challenges, such as the management of labor.

Individuals

At the individual level, important questions exist for scholars of organizational behavior, sociology, and labor economics. Who participates in the gig-economy? What are the longer-term implications of participation? Little is known about the demographic characteristics of gig-economy participants, such as gender, age, income level, prior occupation, education level, citizenship/immigration status (with a few notable exceptions).[14,15] Does the absence of traditional employment benefits influence behavior in significant ways? While the Contingent Worker Supplement of the Current Population Survey will help shed light on these issues, it is not scheduled to go live until May 2017.[b] Obtaining this information will be critical for policy-

a http://cnnmon.ie/2qocUwb

b The supplement was also included in 1995, 1997, 1999, 2001, and 2005, but was discontinued due to lack of funding.

makers as it will allow them to better understand *whom* the laws and regulations that shape the gig-economy might impact.

The effect of gig-economy participation on long-term career outcomes is particularly unclear. A defining attribute of gig-economy jobs is that opportunities for "advancement" within the firm are limited. These jobs might therefore stagnate workers' career progressions, particularly if the gig-economy job requires the worker to make capital investments, such as the purchase of an automobile, which may require debt-based financing. At the same time, job flexibility may allow the worker to pursue other opportunities outside the gig-economy, such as education, which would allow her to improve career outcomes over the long term. Which of these effects dominates, and for whom, is an important opportunity for future research.

Conclusion

The volume of open questions in this space implies the presence of a substantial blind-spot for practitioners and policymakers alike. It is not yet clear how the gig-economy influences social welfare, or how much total surplus is generated by these platforms. Although consumers appear to benefit from reduced prices,[5] media accounts have repeatedly pointed out that working in these markets can have important drawbacks.[c] Understanding the implications of this new form of organizing is critical for scholars from many academic traditions. We therefore strongly urge researchers to consider these and other research questions at the confluence of business, technology, and society. C

c http://bit.ly/2qTDaNi

Which of these effects dominates, and for whom, is an important opportunity for future research.

References

1. Abhishek, V., Guajardo, J.A., and Zhang, Z. Business models in the sharing economy: Manufacturing durable goods in the presence of peer-to-peer rental markets. 2016; http://bit.ly/2pGF4nv.
2. Brazil, N. and Kirk, D.S. Uber and metropolitan traffic fatalities in the United States. *American Journal of Epidemiology.* (2016).
3. Burtch, G., Carnahan, S., and Greenwood, B.N. Can you gig it? An empirical examination of the gig-economy and entrepreneurial activity. (2017); http://bit.ly/2rmJnk5.
4. Chen, M.K. and Sheldon, M. Dynamic pricing in a labor market: Surge pricing and flexible work on the Uber platform. (Mimeo, UCLA), 2015.
5. Cohen, P. *Using Big Data to Estimate Consumer Surplus: The Case of Uber.* National Bureau of Economic Research, 2016.
6. Cohen, M. and Sundararajan, A. Self-regulation and innovation in the peer-to-peer sharing economy. *U. Chi. L. Rev. Dialogue 82* (2015), 116.
7. Cusumano, M.A. How traditional firms must compete in the sharing economy. *Commun. ACM 58*, 1 (Jan. 2015), 32–34.
8. Cusumano, M.A. Platform wars come to social media. *Commun. ACM 54*, 4 (Apr. 2011), 31–33.
9. Edelman, B. and Luca, M. Digital discrimination: The case of Airbnb.com. Harvard Business School NOM Unit Working Paper (14-054), (2014)
10. Edelman B.G., Luca, M., and Svirsky, D. Racial discrimination in the sharing economy: Evidence from a field experiment. Harvard Business School NOM Unit Working Paper, (16-069), 2015.
11. Greengard, S. Smart transportation networks drive gains. *Commun. ACM 58*, 1 (Jan. 2015), 25–27.
12. Greenwood, B.N. and Wattal, S. Show me the way to go home: An empirical investigation of ride sharing and alcohol-related motor vehicle fatalities. *MIS Quarterly 41*, 1 (Jan. 2017), 163–187.
13. Hagiu, A. Strategic decisions for multisided platforms. *MIT Sloan Management Review 55*, 2 (2014), 71.
14. Hall, J.V. and Krueger, A.B. An analysis of the labor market for Uber's driver-partners in the United States. Mimeo, 2015.
15. Ipeirotis, P.G. Demographics of mechanical turk. 2010; http://bit.ly/2qsuJZD
16. Katz, L.F. and Krueger, A.B. The rise and nature of alternative work arrangements in the United States, 1995–2015 (2016); http://bit.ly/2pSZUuX
17. Malhotra A. and Van Alstyne, M. The dark side of the sharing economy... and how to lighten it. *Commun. ACM 57*, 11 (Nov. 2014), 24–27.
18. Scheiber, N. How Uber uses psychological tricks to push its drivers' buttons. The New York Times (Apr. 2, 2017).
19. Telles, R. Digital matching firms: A new definition in the "sharing economy" space. U.S. Department of Commerce Economics and Statistics Administration, 2016; http://bit.ly/1XBAFwc
20. Zervas, G., Proserpio, D., and Byers, J. The rise of the sharing economy: Estimating the impact of Airbnb on the hotel industry. Boston University School of Management Research Paper (2013–16), 2015.
21. Zhang, S., et al. Professional versus amateur images: Investigating differential impact on Airbnb property demand. In *Proceedings of the Conference on Information Systems and Technology*, 2016.
22. Zhu, F. and Furr, N. Products to platforms: Making the leap. *Harvard Business Review 94*, 4 (2016), 18.

Seth Carnahan (scarnaha@umich.edu) is the Sanford R. Robertson Assistant Professor in Business Administration in the Ross School of Business at the University of Michigan.

Gordon Burtch (gburtch@umn.edu) is an assistant professor of Information and Decision Sciences at the University of Minnesota Carlson School of Management.

Brad N. Greenwood (greenwood@temple.edu) is an assistant professor of Management Information Systems and the Richard J. Fox Faculty Fellow at Temple University's Fox School of Business.

Calendar of Events

March 4–5
I3D '17: Symposium on Interactive 3D Graphics and Games,
San Francisco, CA,
Sponsored: ACM/SIG,
Contact: Kenny Mitchell,
Email: drkennymitchell@yahoo.com

March 6–9
HRI '17: ACM/IEEE International Conference on Human-Robot Interaction,
Vienna, Austria,
Contact: Bilge Dincer Mutlu,
Email: bilge@cs.wisc.edu

March 7–11
CHIIR '17: Conference on Human Information Interaction and Retrieval,
Oslo, Norway,
Sponsored: ACM/SIG,
Contact: Ragnar Nordlie,
Email: ragnar.nordlie@hioa.no

March 13–16
IUI'17: 22nd International Conference on Intelligent User Interfaces,
Limassol, Cyprus,
Co-Sponsored: ACM/SIG,
Contact: George Angelos Papadopoulos,
Email: george@cs.ucy.ac.cy

March 16–17
TAU '17: ACM International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems,
Bay Area, CA,
Sponsored: ACM/SIG,
Contact: Qiuyang Wu,
Email: qiuyang.wu@gmail.com

March 19–22
ISPD '17: International Symposium on Physical Design,
Portland, OR,
Sponsored: ACM/SIG,
Contact: Mustafa Ozdal,
Email: mustafa.ozdal@cs.bilkent.edu.tr

March 20–23
TEI '17: 10th International Conference on Tangible, Embedded, and Embodied Interaction,
Yokohama, Japan,
Sponsored: ACM/SIG,
Email: inakage@kmd.keio.ac.jp
Contact Phone: 81-467-32-7941

DOI:10.1145/3097352

Peter J. Denning

The Profession of IT
The Beginner's Creed

We all need to learn to be expert beginners.

I HAVE TAUGHT operating systems for many years to undergraduate and graduate students. Operating systems are a complex technology difficult to master and it is easy for students to fall into unproductive moods while studying them. More often than I would like, my students were unable to escape their unproductive moods and wound up not learning the technology and being dissatisfied with the course.

We often encounter the same problem in our professional work. New technologies are constantly emerging and some are producing disruptive avalanches of change.[3,4] The emerging technologies are unfamiliar, complex, and difficult to master. We fall into unproductive moods and wind up missing deadlines, getting left behind other colleagues, or being swept away by an avalanche of change. Since we do not often pay attention to our moods, the obstruction to learning seems like a mysterious, unidentifiable force that compounds our frustration.[1]

In her book, Gloria Flores points out that young children tend to dwell in productive moods, for they are ever eager to learn new things.[2] By their teen years, sometimes much earlier, many have changed: they seem to frustrate easily, fear mistakes, and distrust their abilities. In these moods they get defensive and resist learning. Adults fall into the same learning-blocking moods as well. For example, adult experts are confident about their abilities, but when thrust into a situation where they need to learn something new, many quickly become uncomfortable and lose their confidence. They do not welcome the opportunity to learn something new; they want to escape. Their moods of confusion, anxiety, insecurity, embarrassment, and resignation block their professional development and advancement. They are prone to worrying about their reputation if others see them as not competent. Their long expertise left them rusty at the skills of beginners—being okay with not knowing something, allowing themselves time to learn, or asking for help. In reading Flores's account, I realized that one of a mentor's greatest contributions is to help their protégés recognize their moods and learn to shift to moods productive for learning.

These insights helped recently with a cohort of adult (age 30–35) graduate students in my operating systems class. They were enjoying the class until their first quiz. Many got worse grades they expected and fell into various bad moods including discouragement, anger, and even resentment. Inspired by Flores's insights, I decided to talk to them about their moods. I composed a poetic page that walked through all the moods a beginner is likely to experience. I called it "The Beginner's Creed," a copy of which appears here.

I asked my students, "How many of you are an expert in some area?" Every hand went up. Then I asked, "How many of you feel like a beginner in operating systems?" Every hand went up. Then I asked, "How many of you like being a beginner?" Only two hands went up. I said, "We need to have a conversation about that."

I handed out the Beginner's Creed and asked them to read it. When all were done, I read it aloud to them so that I could intonate its moods. I asked them to read it to themselves every day for a week. For the rest of the course, the students were much more relaxed about their roles as beginners and were much more engaged in the work of the course. At the end of the course, when the project teams stood up to make their final presentations to the class, one team said proudly, "We are beginners! And look at what we have accomplished!" In my concluding remarks of the course, I said, "Congratulations. You are no longer beginners. You are now advanced beginners. You are prepared to learn to be competent with operating systems." Some smiled with pride.

What is even more interesting is the Beginner's Creed resonated with concerns my students had outside of class and in other departments of the university. After reading the Creed, one student immediately asked, "May I give a copy of this to my son?" Another asked the same question regarding his boss. A senior military officer on campus showed it to one of the students who was having a particularly difficult time acclimating to his studies; the student said, "I wish you had given me this when I first arrived! I see that I have been resisting too much." The campus librarian framed a copy and hung it on the wall of the library. Flores's insight is powerful indeed and speaks to a yearning that many people have been unable to articulate.

Even though we computing professionals are the authors of disruptive technologies, we are often threatened by our own creations. Our own tools and ways of doing business can suddenly become obsolete. When a disruption comes, we need to reinvent ourselves by learning new things in new emerging domains. To be ahead of disruption, we need to antici-

pate emergences and do the learning early before it becomes a critical necessity. Our moods will be a problem if we do not learn to recognize them, especially the unproductive moods, and shift to ones that are productive for learning.

An essential part of learning is to allow ourselves to be a beginner in the new domain. That means we come to the domain knowing nothing or next to nothing about the domain. Beginner is the first stage of a progression of skill in a domain that can take us next to advanced beginner, competent, proficient, expert, and master. As we get older, we find that we are experts at some things—and we like being an expert. We like when people look up to us and ask us for guidance and wisdom. But when we come into a new domain in which we are just a beginner, we cannot expect our expertness from other domains to help us. In fact, it is likely to draw us into unproductive moods such as frustration over not learning fast enough or discouragement that we are not treated as an expert.

We all marvel at how easy it seems to be for your young children to be beginners. Perhaps that is because they are experts at nothing and do not have their minds clouded with any expectations about being an expert. They just approach the learning as an opportunity to play, adventure, and experiment. Older children often acquire self-assessments that block them from adventure and play, whereupon they may have more trouble learning.

So here is the Beginner's Creed, a one-page declaration that you can recite to yourself to help you when you find yourself in a new domain as a beginner, whether by your own choice or by circumstance. Make a copy and read it every day for a week, especially when you are a beginner. Return to it as needed. C

References

1. Denning, P. and Flores, G. Learning to learn. *Commun. ACM 59*, 12 (Dec. 2016), 32–36.
2. Flores, G. *Learning to Learn and the Navigation of Moods.* Pluralistic Networks Publishing, 2016.
3. Friedman, T. *Thank You for Being Late.* Farrar, Straus, and Giroux, 2016.
4. Kelly, K. *The Inevitable.* Viking, 2016.

Peter J. Denning (pjd@nps.edu) is Distinguished Professor of Computer Science and Director of the Cebrowski Institute for information innovation at the Naval Postgraduate School in Monterey, CA, is Editor of ACM *Ubiquity*, and is a past president of ACM.
The author's views expressed here are not necessarily those of his employer or the U.S. federal government.

The Beginner's Creed

I am a beginner.
I am entering a new game about which I know nothing.
I do not yet know how to move in this game.
I see many other people playing in this game now.
This game has gone on for many years prior to my arrival.
I am a new recruit arriving here for the first time.
I see value to me in learning to navigate in this domain.

There is much for me to learn:

- The basic terminology
- The basic rules
- The basic moves of action
- The basic strategies

While I am learning these things I may feel various negative reactions:

- Overwhelmed at how much there is to learn
- Insecure that I do not know what to do
- Inadequate that I lack the capacity to do this
- Frustrated and discouraged that my progress is so slow
- Angry that I have been given insufficient guidance
- Anxious that I will never perform up to expectations on which my career depends
- Embarrassed that everyone can see my mistakes

But these moods are part of being a beginner. It does not serve my goal and ambition to dwell in them. Instead,

- If I make a mistake, I will ask what lesson does this teach.
- If I make a discovery, I will celebrate my aha! moment.
- If I feel alone, I will remember that I have many friends ready to help.
- If I am stuck, I will ask for help from my teachers.

Over time, I will make fewer mistakes.
I will gain confidence in my abilities.
I will need less guidance from my teachers and friends.
I will gain familiarity with the game.
I will be able to have intelligent conversations with others in the game.
I will not cause breakdowns for promises that I lack the competence to keep.
I have an ambition to become competent, perhaps even proficient or expert in this game. But for now,
I am a beginner.

—By Peter J. Denning

DOI:10.1145/3097354

Marc Snir

Viewpoint
The Informal Guide to ACM Fellow Nominations

Recommendations for a successful nomination process.

IMAGE BY ANDRIJ BORYS ASSOCIATES/SHUTTERSTOCK

THE ACM FELLOWS program recognizes the top 1% of the ACM membership that has shown excellence in technical, professional, and leadership contributions. The ACM web site (http://awards.acm.org/fellow/) provides detailed information about the criteria of the program and detailed instructions about the process and requirements for nominations. This Viewpoint aims to complement these formal instructions with informal advice about writing good nominations and endorsements. It is based on the personal experience of the author and of other current and past members of the ACM Fellows Award committee. It is not an official ACM document, therefore presentation as a *Communications* Viewpoint.

The success of a nomination depends first and foremost on the quality of the candidate. Usually, the candidate will not be familiar to most or all the committee members; a committee member that knows the candidate well could have a conflict and not be able to participate in the discussion of that candidate. Further, few committee members are likely to be thoroughly versed in the candidate's subfield. Therefore, decisions on candidates will be based almost uniquely on the information provided by the short nomination and endorsements. Hence, the quality of these documents is paramount: While a good nomination may not help a weak candidate, a lousy one may sink a good candidate. Most nominators and endorsers understand this and write well-considered nominations and endorsements.

Unfortunately, we still see some poorly written nominations and we see a substantial number of poorly written endorsements—I hope the following suggestions will reduce the frequency of those. Similar and additional suggestions were made in a blog by Jim Horning more than a decade ago (http://horning.blogspot.com/2006/10/making-case-for-acm-fellow.html).

I assume the process starts with a decision that a candidate is ripe for nomination and with the selection of a nominator. Many institutions have an award committee that is responsible for this first step. In other places, the candidate will start the process. The nominator writes the nomination and

selects the endorsers. The endorsers then submit their endorsements. The recommendations here cover the stages of this process: Selecting a nominator, writing a nomination, selecting endorsers, and writing endorsements; they address the people involved in these various activities.

Choose an experienced and willing nominator. Writing good nominations is a skill that improves with experience; it is also a time-consuming activity. If no experienced and willing nominator is to be found, then consider having a more experienced person read the nomination and suggest improvements.

Involve the candidate in the nomination process. A nominator might be tempted to nominate a candidate without her knowledge, to avoid disappointment in the case the nomination fails. This is a bad idea, for a variety of reasons: The candidate is best placed to provide accurate information on her achievements and for selecting plausible endorsers. Besides, since each nomination is a bet that carries a risk (a two-year waiting time before the next attempt) it is best to consult the candidate before making the bet on her behalf.

Do not let the candidate write the nomination on her own. A candidate may want to write her own nomination; such a write-up could be a useful draft, but should not be the final nomination. For one thing, the nomination is supposed to be contributed by the nominator and express his views, not the views of the candidate. A nominator will have a more objective view of the importance of various contributions and a better understanding of how the nomination will be read by a committee that is not necessarily familiar with the candidate. The nominator should have more experience writing this type of document.

Start creating the nomination early. An earlier start means more time to iterate on the nomination text. It means that endorsers are more likely to agree to endorse, since they have not yet been approached multiple times, and they have time to write a quality endorsement. It means endorsements are likely to be submitted well ahead of the deadline, thus ensuring that unforeseen events will not prevent a submission. ACM will accept only the first five submitted endorsements. An endorser is unlikely to be pleased when his endorsement is rejected as being superfluous. Rather than soliciting more endorsements than needed and creating superfluous work, it may be advisable to solicit five endorsements from the preferred endorsers and ensure an additional person will be willing to write an endorsement on short notice, if needed.

While a good nomination may not help a weak candidate, a lousy one may sink a good candidate.

Ensure the formal requirements for Fellowship are satisfied. ACM requires five years of continuous membership. People might be well known in the field and have a long association with ACM, but could have dropped their membership for a period during the last five years though negligence or frugality. Check that the five-year requirement is satisfied before starting the process.

Focus the nomination write-up on the formal requirements for an ACM Fellow.

"The title of ACM Fellow denotes professional excellence, as evidenced by technical, professional and leadership contributions that:

- *advance the arts, sciences and practices of computing,*
- *promote the free interchange of ideas and information in the field,*
- *develop and maintain the integrity and competence of individuals in the field, and advance the objectives of ACM."*

The text speaks quite specifically to contributions in the field of computing. Contributions outside the field are not relevant to the nomination. The committee usually takes a broad view of computing to include cognate areas. But success in an executive position unrelated to computing (for example, as Provost at a University or manager of a non-IT unit in industry), or voluntary work unrelated to computing will not carry much weight. Think carefully before devoting precious nomination text to such activities.

The large majority of the successful candidates are selected mostly on the strength of their scientific and technical contributions. A very small number is selected mostly on the strength of outstanding service to the ACM community. However, candidates are expected to have contributed both.

Avoid platitudes and do not spend your word budget on evident claims or meta-discussions. One annoying example is *"This nomination is a no-brainer."* It may be a no-brainer for the nominator (or endorser), but no nomination is a no-brainer for the award committee. Let the evidence show that the nomination is a no-brainer.

Another common example is: *"It is my opinion that the candidate is in the top 1% of ACM members."* The nominator is very unlikely to be acquainted with a representative sample of the entire ACM population, so such a statement weakens the credibility of the nomination.

Don't spend space reporting Google citation counts. Committee members are given reports from ACM on citation rates (both Google and ACM Digital library) and similar metrics. Given the large diversity of computing the numbers are relevant only when compared to numbers of people of a similar age working in the same area. Do not paraphrase the CV of the candidate.

Do not assume the award committee is familiar with the candidate's research area. An award committee of 10 people cannot possibly represent all research areas in computing, and the committee member most familiar with the candidate may have a conflict. Therefore, it is essential to explain why an achievement is important. "She proved theorem xxx" is not useful without an explanation of why people care about xxx; "she developed protocol yyy" is not useful without an explanation of how broadly the protocol is used.

Do not assume the committee is familiar with the candidate's country. ACM strives to have an international representation on the committee, but not every country can be represented.

Committee members may not be familiar with the importance of national awards, national academic societies or national leadership positions. Please explain their importance.

P**rovide evidence of accomplishment that is most relevant to the type of accomplishment.** Accomplishments are meaningful if they have a visible impact. Impact will be of different natures for different types of impact. If the achievements are in theoretical computer science, the impact is intellectual advance in our understanding of computing; the evidence could be subsequent research that builds on this advance as evidenced by citations. If the achievements are in applied research, the impact would be in the use of the developed technology; tangible artifacts could be more important than citations. If the achievement is to the computer industry, then the impact would be industrial success, with products as evidence of impact. Of course, these are not hard rules, and many caveats apply: The last inventor of a new concept is often more cited than the first one; and commercial success of a product is only weakly correlated to its technical quality.

Speak of the past, not of the future. Fellows are selected for their actual accomplishments, not for their potential accomplishments in the future. The nomination and the endorsements should focus on the impact the research has had so far, not on the impact it is likely to have in the future.

Provide all the required information. The nomination is required to include:

- *Candidate's most significant professional accomplishments and their foundational, technical, commercial, or other achievements (limited to 750 words).*
- *Up to 8 specific contributions epitomizing the significance and lasting impact of those accomplishments (limited to 300 words).*
- *Candidate's most significant leadership roles in ACM or other service activities (limited to 300 words).*
- *Formal professional recognition the candidate has received for his/her contributions, such as awards or other honors (limited to 300 words)*

Don't skip any of these sections. Please remember that "contributions" need not be publications. Also please explain why the contributions, roles, or recognition are significant. As noted earlier, do not assume the committee members will just know.

Select the endorsers carefully. One is naturally tempted to pick the most famous ACM Fellows that are willing to write an endorsement. Most will be diligent in doing so. However, some will write an endorsement that sounds like the "Model of a Letter of Recommendation of a person you are unacquainted with" that Benjamin Franklin once composed (http://sites.sas.upenn.edu/bfranklin/pages/letter-recommendation. Famous people are busy people and, with the best intentions in the world, time pressure may result in proforma, weak endorsements. This is particularly likely if the famous person is not already deeply familiar with the candidate's work.

Endorsements are more convincing when they come from people who work in the candidate's field of specialty and have made use of the candidate's work. If the candidate co-created a key result in their field, having at least one of the collaborators as an endorser is recommended. Such a collaborator could be in the same organization as the candidate. On the other hand, endorsers from the same organization that are not closely connected to the candidate's work are discouraged, as are endorsers who have an obligation to the candidate (for example, former grad student or current supervisor). Having only collaborators as endorsers is a bad idea. Having all endorsers chosen from a narrow community (a small sub-specialty or a small national community) is a bad idea. Carefully weigh the trade-off between the familiarity of an endorser with the candidate and his perceived objectivity; and between the familiarity of the endorsers with the candidate and the breadth and diversity of the community they represent collectively. If one cannot find five endorsers that are ACM Fellows or have equal standing, and balance well these conflicting requirements, then it is likely the candidacy is premature.

For a candidate coming from a smaller country, or a country with a smaller community of ACM Fellows, it is important to have endorsers of another nationality. The committee wants to know how famous the candidate is in her field; not how famous she is in her country. Good nominations often use endorsers that can testify to the importance of different contributions; one might focus on service, others on different aspects of the technical contributions. The right mix will depend on the types of contributions and their relative importance.

While the candidate's advice is important in selecting the endorsers, it is better that the endorsers be approached by the nominator: It will be easier for a potential endorser to say no if he is approached by the nominator, rather than by the candidate; a straight no is preferable to a tepid endorsement.

Write meaningful endorsements. A one-sentence endorsement such as "I believe this candidate merits fellow status" is poison, even if it comes from a very illustrious computer scientist. Don't agree to provide an endorsement if there is a risk you might not be able to say more; don't choose an endorser you suspect may be content with a one-sentence endorsement. Substantive, thoughtful, convincing endorsements will provide enough detail for credibility. This generally uses most of the space allotted.

There is no point repeating text that appears in the nomination—this is not new information. The endorsement is a "personal assessment of the candidate's impact on the computing field." The endorser should explain why he believes the impact is important enough to merit recognition with fellow status. Ideally, this explanation is distinct from or expands upon the explanation provided in the nomination. If it is not obvious why the endorser is able to assess the quality of the candidate's contribution, then a short explanation to that effect will be useful.

Please remember: An endorsement of the form "I am famous and trust me on this one" is likely to do more harm than good. C

Marc Snir (snir@illinois.edu) is Richard Faiman Professor in the Department of Computer Science at the University of Illinois at Urbana-Champaign.

I wish to thank Laura Haas, Joseph Konstan, Craig Partridge and Mary Shaw for their comments on a draft of this Viewpoint.

Blog Ubiquity

INFORMATION EVERYWHERE

The newest ACM forum.

Contributions that cover the vast information-rich world where computing is embedded everywhere.

ACM's *Ubiquity* is the online magazine oriented toward the future of computing and the people who are creating it.

We invite you to participate: leave comments, vote for your favorites, or submit your own contributions.

Captivating topics.

Net Neutrality and the Regulated Internet

The End of Life As We Know It

A Shortage of Technicians

The Fractal Software Hypothesis

Your Grandfather's Oldsmobile–NOT!

Superscalar Smart Cities

Association for Computing Machinery

Visit us at http://ubiquity.acm.org/blog/

DOI:10.1145/3080010

Article development led by acmqueue
queue.acm.org

One system's side effect is another's meat and potatoes.

BY PAT HELLAND

Side Effects, Front and Center

WE THINK OF computation in terms of its consequences. The big MapReduce job returns a large result. Web interactions display information. Enterprise applications update the database and return an answer. These are the reasons we do our work.

What we rarely discuss are the *side effects* of doing the work we intend. Side effects may be unwanted, or they may actually cause desired behavior at different layers of the system. This article points out some fun patterns to keep in mind as we build and use our systems.

As we build systems, we come across a bunch of layers of abstractions. The datacenter provides power, networking, cooling, and protection from rain. The server provides DRAM (dynamic random-access memory), SSD (solid-state drive), network, computation, HDD (hard-disk drive), and more. The

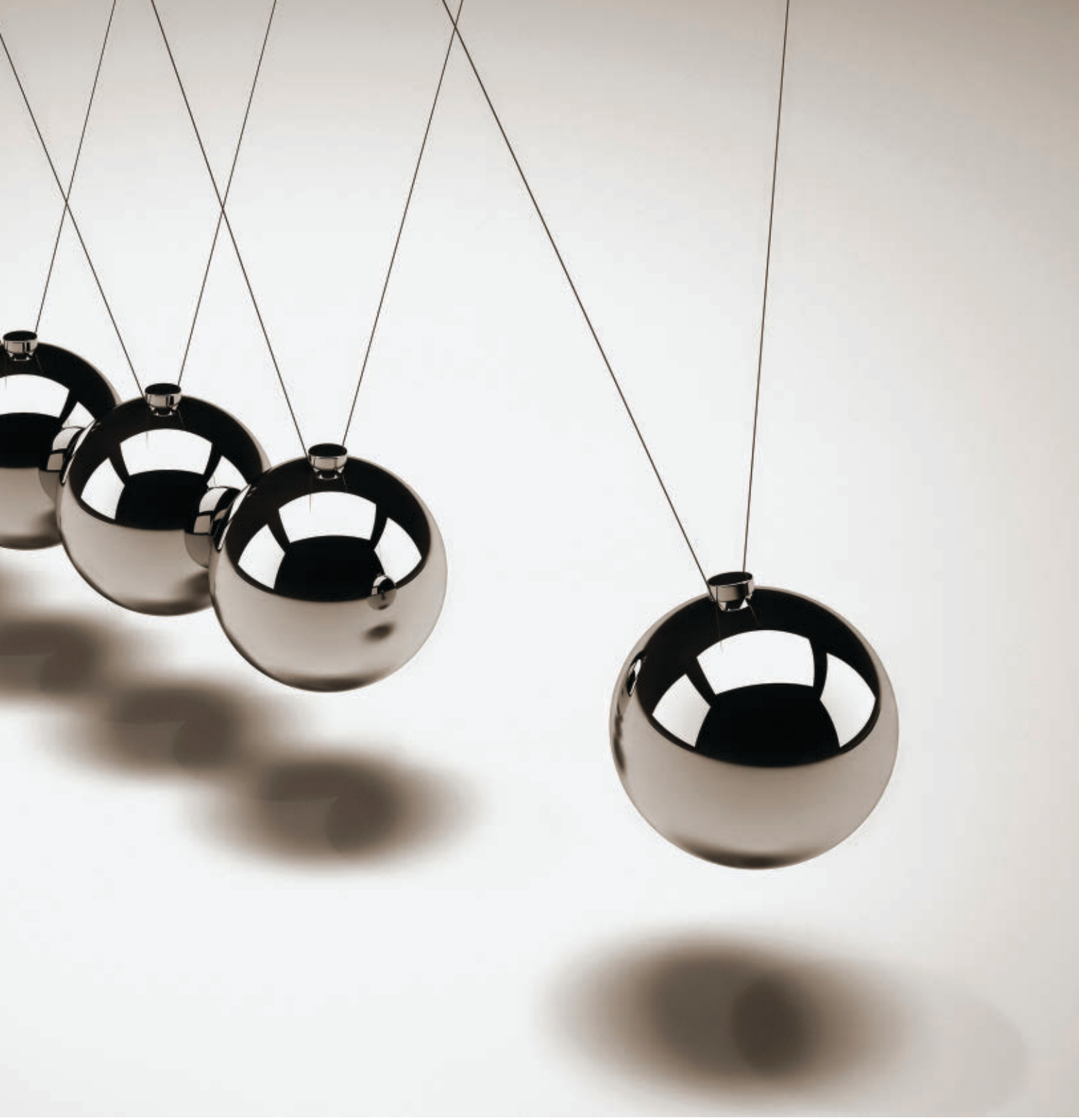


IMAGE BY ORLA/SHUTTERSTOCK

operating system provides processes, virtual memory, file systems, and more. *Application* and *platform* are subjective terms: Application is the stuff that runs on top of me; platform is the stuff I run on top of.

As an example, memory management resides in a layer of abstraction below most application code. When memory is allocated from a heap, the application worries about *malloc* and *free* or some equivalent. It doesn't give a darn how the memory is managed or even where it resides. The application certainly doesn't care about fragmentation of the heap.

TMI. In the past few decades, the phrase *TMI*, meaning *too much information*, has entered the lexicon. It generally refers to knowledge about someone's personal life or hygiene that you have heard and wish you could un-hear. When your Great Uncle tells you about his digestive problems, that's TMI!

TMI can also refer to stuff you really don't want to know about that other subsystem you call from your application.

Side effect is a fancy computer science term for TMI.

Side Effects In Lots of Places

We see side effects in many places at many levels of abstraction. We even see

side effects in life outside of computers. Here are a few to contemplate:

- Messages into and out of a microservice are typically logged for monitoring purposes.
- Competition for any resource may cause congestion and delay for other competing work. This is very much like the bad luck you experience as you try to drive on the freeway just as the ballgame is getting out.
- Traffic into a microservice may cause heap fragmentation, impacting the responsiveness of the next request as the garbage in the heap is collected.
- Writing to the disk may cause the file system to get full. The next request is impacted.
- I may reserve a seat on an airplane, causing the next request by someone else to fail. It doesn't matter if I later cancel and don't use the seat. The other flyer still loses and won't be on that flight.

Each of these examples can be driven by work that is subsequently undone or aborted at the higher layer of abstraction. Logically, the work is undone *from the perspective of the higher layer*. Still, there are persistent changes visible at the lower layers and TMI for the upper layers to handle.

Transactions Are in the Eye of the Beholder

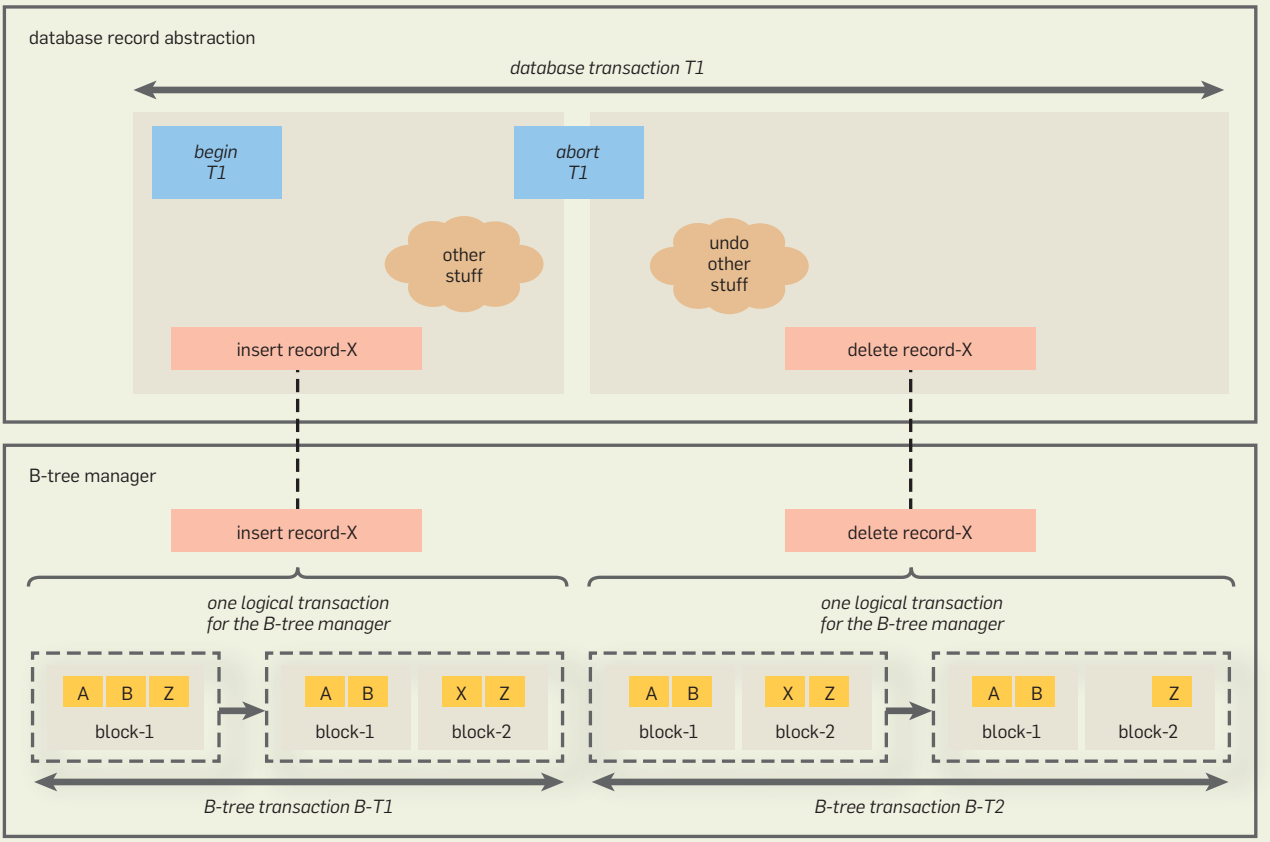
The word *transaction* is used to describe some changes that are all or nothing. *ACID transactions*[1,2] refer to those that are atomic, consistent, isolated, and durable. These attributes ensure a reliable sense of one change at a time and are most commonly associated with databases and database transactions. Transactions are a fascinating tool—and one I've spent a large part of my 38-year career working on.

It turns out transactions are frequently composed of other transactions at different layers of abstraction. This is called an *open-nested transaction*.[4] In an open-nested transaction, a higher-layer transaction consists of multiple lower-layer transactions. To abort the higher-layer transaction, the system may need to issue compensated lower-level transactions that undo the effect of the upper one.

Example 1. The trip to Europe. Now let's consider some side effects that may result from a simple business trip to Europe.

- I make a reservation for a Wednesday night at a hotel in Paris. This is part of a set of reservations for airplanes, cars, and hotels for my weeklong trip to Europe.
- The reservation causes the occupancy of the hotel to cross a threshold so more staff and food for the restaurant are needed.
- The hotel restaurant orders a new shipment from the grocer for Tuesday.
- The grocer calls the shipping company for a delivery on Tuesday.
- The shipping company notices a projected shortfall in its petrol fuel supplies and orders more fuel for Monday.
- Then I cancel my trip to Europe.

Layered abstractions.



My reservation caused a cascading set of effects that I don't see. Indeed, telling me about them would truly be TMI, causing me a great deal of confusion. Furthermore, these side effects persist even if my initial work is canceled.

Side effects persist even if the stimulating activity is canceled or aborted.

Example 2. The B-tree split. Database management systems typically store records in a B-tree. Consider the following scenario:

- Record X is inserted by the user at the record-oriented layer of abstraction as a part of transaction T1.
- The database system calls its B-tree manager, which climbs down the B-tree to insert Record X. Upon discovering that the leaf of the B-tree is too full to receive Record X, it splits the leaf into two and stores Record X in one of them.
- Transaction T1 does some more stuff.
- Transaction T1 is aborted at the record-oriented layer. As a consequence, the B-tree manager is called to delete Record X from the B-tree, which it does. *The split of the leaf of the B-tree is not undone.*

When transaction T1 is aborted, all effects of T1 are eliminated from the set of records making up the database. Still, the leaf of the B-tree has been split and remains split. The accompanying figure shows layered abstractions with database records on top and B-tree implementations below. A database transaction inserts into a B-tree, causing a block splt. Later, the database transaction aborts, causing a delete to the B-tree. While Record X is deleted from the B-tree, the block split is not necessarily undone.

The record-oriented database is correct with T1 removed. The B-tree as a B-tree is correct with the proper leaves, indices, and pointers. Still, the B-tree is different because the transaction inserted and later deleted Record X.

The split of the B-tree is a side effect of the aborted transaction T1. From the perspective of the set of records in the database, that's TMI.

Idempotence and Side Effects

Personally, I think all distributed computing depends on timeouts, retries, and idempotence.[3] *Idempotence* is the property of certain operations that you can do more than once but get the same result as if you did it once. Timeouts, retries, and idempotence allow the distribution of work with very high probabilities of success.

Now, what does idempotence mean if there are side effects? Is an operation idempotent if it causes monitoring of the call? That yields two monitoring records and is, hence, not an identical result. An operation is idempotent if it is repeatable at the desired layer of abstraction. It is typically considered OK if logging and monitoring record both attempts.

Idempotence is in the eye of the beholder.

Side effects to an idempotent operation are always OK. After all, they are side effects and, hence, not semantically important.

I'll Get Around to Hysteresis

It is quite common for one layer of the system to be slow in undoing stuff it recently did. This avoids the overall system flopping and jittering too aggressively.

For example, when the hotel reservation is canceled because I chose not to go to Europe, it probably didn't change the order for groceries. Perhaps my reservation pushed the occupancy to 200 rooms and a new level of demand for the restaurant. Most likely, the expected occupancy will need to drop to 180 or so before the hotel will fiddle with the grocery order. Repeatedly calling the grocer to schedule, then cancel, then schedule deliveries is likely to drive the grocer to remove the hotel from its list of customers.

Similarly, most B-tree managers are not anxious to merge two adjacent blocks when they fall below 50% each. The cost of rejiggering their contents repeatedly is too high.

Side effects from canceled work will sometimes leave the system in a different state from what it was before. That may, in turn, impact subsequent requests.

Conclusion

Our systems compose in fascinating ways that have interesting interactions. To cope with this, many times we need to ignore the complications inside of the systems we use and just pretend life is simpler than it really is. That's great! We live in a higher level of abstraction and don't sweat the details.

One system's side effect is another's meat and potatoes.

Still, the system providing the lower level of abstraction sees its job as its reason for existence. An order of groceries is the main purpose of the restaurant-scheduling application. Similarly, the B-tree manager has to keep records, fit them into the B-tree, and split when necessary. That's not a side effect but rather part of the job.

Side effects are only side effects to busybodies not minding their own business!

Just Look Past the TMI

If every system pays attention to its own layer of abstraction and ignores the TMI of other layers of abstraction, all of this composition makes sense. Good design involves knowing when stuff is relevant and when stuff is TMI. After all, your Great Uncle's digestive problems are relevant to his doctor! C

Related articles on queue.acm.org

A Conversation with Erik Meijer and José Blakeley
http://queue.acm.org/detail.cfm?id=1394137

Abstraction in Hardware System Design
Rishiyur S. Nikhil
http://queue.acm.org/detail.cfm?id=2020861

Bridging the Object-Relational Divide
Craig Russell
http://queue.acm.org/detail.cfm?id=1394139

References

1. Gray, J. and Reuter, A. *Distributed Transaction Processing: Concepts and Techniques.* Morgan Kaufmann, 1993.
2. Haerder, T. and Reuter, A. Principles of transaction-oriented database recovery. *ACM Computing Surveys 15*, 4 (1983), 287.
3. Helland, P. Idempotence is not a medical condition. *acmqueue 10*, 4 (2012).
4. Weikum, G. and Schek, H.-J. Multi-level transactions and open nested transactions. *Data Engineering 14*, 1 (1991), 60–64.

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

DOI:10.1145/3079970

Article development led by acmqueue
queue.acm.org

An improvement over UML.

BY MARK A. OVERTON

The IDAR Graph

UNIFIED MODELING LANGUAGE (UML)[6] is the de facto standard for representing object-oriented designs. It does a fine job of recording designs, but it has a severe problem: its diagrams don't convey what humans need to know, making the diagrams difficult to understand. This is why most software developers use UML only when forced to.[1]

For example, the UML diagrams in Figures 1 and 2 portray the embedded software in a fax machine. While these diagrams are attractive, they do not even tell you which objects control which others. Which object is the topmost controller over this fax machine? You don't know. Which object(s) control the Modem object? You don't know.

People understand an organization, such as a corporation, in terms of a control hierarchy. When faced with an organization of people or objects, the first question usually is: "What's controlling all this?" Surprisingly, UML has no concept of one object controlling another. Consequently, in every type of UML diagram, no object appears to have greater or lesser control than its neighbors. This absence of a control hierarchy in software design does much harm in the following ways:

- Designs are difficult to understand. Showing no hierarchy is like portraying a corporation by drawing a line between every pair of employees who interact with each other. Such a chart would rapidly become incomprehensible spaghetti. An organizational chart is drawn as a control hierarchy for good reason: people can readily understand them, regardless of the corporation's size.
- Because any object can interact with any other object in any way desired, code structure slides into disorganization as people add features and interactions to objects during design and implementation.
- Maintenance becomes slower and more error-prone because learning curves are steeper. In addition, maintainers can and do insert hacks anywhere, causing code to decay.

These problems mean designs tend to become messy during both initial implementation and maintenance, resulting in more bugs and delays.

The Basics of an IDAR Graph

To be useful, a graph that portrays software design must communicate in a way that humans understand. An organization of objects in software is analogous to a human organization, and almost without exception, an organization of people is portrayed as a control hierarchy, with the topmost person having the broadest span of control. Based on this idea, Figure 3 is a simple IDAR graph that portrays part of the same fax machine design shown in Figures 1 and 2, but expressed as a control hierarchy.

PHOTO BY FABRIKA SIMF

In an IDAR graph, boxes represent objects. If a class has only one instance (the most common case), then the box is labeled with the class name. An arrow connecting two boxes means that the upper object commands (and thus controls) the lower object. Such command arrows always point down. In Figure 3, the `Fax` object is the topmost controller, which commands the `Receive` and `Send` objects, which in turn control `ImageProc` (image processing). `Command` arrows may be labeled with the names of commands that are sent. For example, `Fax` commands `Send` to `sendFax`. Note that `ImageProc` has two bosses. Having multiple bosses is uncommon in human organizations but is common (and encouraged) in software to prevent redundant implementations.

Objects need to communicate in more ways than commands. For example, they often need to exchange data and inform each other about events and results. In an IDAR graph, such non-command communications are called *notices* and are shown as floating arrows. For example, in Figure 3, `Send` tells `Fax` that transmission is done via the `done` notice.

Both commands and notices are merely method calls. This means that the public methods in each object are divided into two groups: commands and notices. Software designers must give careful thought to which methods will be commands, because they determine the hierarchy. Commands and notices have constraints, which are precisely defined later in this article.

A note about terminology: when you call a command (method) in an object, you are said to be *commanding* (or *sending a command to*) that object; when you call a notice in an object, you are *notifying* (or *sending a notice to*) that object.

More Features of IDAR Graphs

A graph of a design should portray other salient features, such as threading, data flows, and the use of indirection. The complete IDAR graph of the fax machine in Figure 4 exemplifies some of these additional features.

The horizontal line above the `Connect` and `Negotiate` boxes is analogous to a horizontal line in an organizational chart: it groups subordinates under their manager. In an IDAR graph, such a *rail* (as it's known) is more general, as it indicates that all objects above the line (called *superiors* or *bosses*) command all objects below it (called *subordinates* or *workers*). In this fax machine, two superiors (`Receive` and `Send`) command three subordinates (`Connect`, `Negotiate`, and `ImageProc`).

An object containing a thread is said to be *active* and is denoted with double vertical lines on each side of its box. This notation was taken from UML. In the fax machine design in Figure 4, `Fax` and `ImageProc` are active.

Indirect method-calls are indicated by a bubble (circle) placed on the tail of the appropriate arrow. Such indirection can be explicit in the source code or implicit using polymorphic inheritance. Indirection is commonly used for notices sent from a subordinate to one of multiple superiors, such as the `connected` notice in Figure 4 that is sent from `Connect` to `Receive` or `Send`.

A subsystem is a separate hierarchy (a separate graph) with a *subman* (subsystem manager) as its topmost object. A subman controls its subsystem and is portrayed as an elongated hexagon. In a graph that commands a subsystem, only the subman is drawn. For example, in Figure 4, `Printer` and `Scanner` are the submans of their re-

spective subsystems, which should be shown in separate graphs.

A dashed arrow denotes a data flow. Notice-arrows often parallel a data flow, because data flows are usually implemented using notices. An example is the `pixelRow` notice sent from the `Scanner` subsystem to `ImageProc`.

Notice that the names of some commands and notices in Figure 4 are prefixed with numbers. These optional sequence numbers show the order of actions composing an operation. In this case, they show the sequence of calls to send a fax. A copy of this graph could be enhanced to show the sequence for receiving a fax. Such annotated graphs replace sequence diagrams in UML. They are easier to understand because you can see which actions are commands versus responses, in addition to their order.

It might surprise you that the IDAR graph in Figure 4 is the same design as the UML diagrams in Figures 1 and 2. Compare these diagrams. In the IDAR graph, you can easily see which objects control which others, thus revealing how this design operates.

Four Rules

The principles underlying IDAR graphs can be expressed in the form of four rules. They form the acronym IDAR, the namesake of these graphs. The rules are:

▸ **Identify.** Each public method in an object is identified as either a command or a notice. From its caller's viewpoint, a notice only imports or exports needed information. A command may do anything.

▸ **Down.** When graphing the calls to commands among objects, the arrows must point down.

Figure 1. UML class diagram for a fax machine.

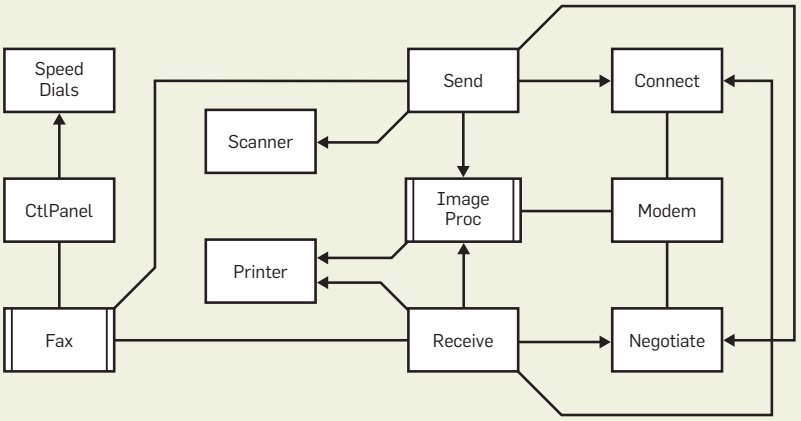
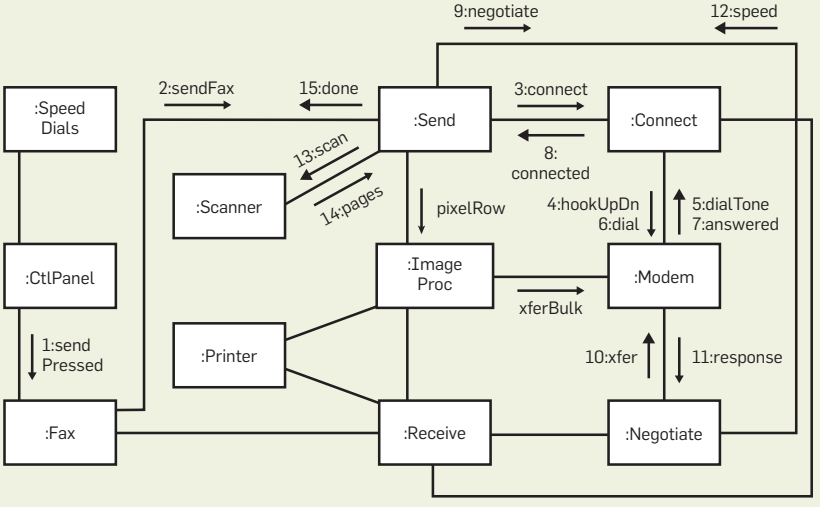


Figure 2. UML communication diagram for sending a fax.



▸ **Aid.** A command or notice may, unknown to its callers, aid its object by performing part or all of a previously commanded action.

▸ **Role.** A brief role is written for each object and method that summarizes the service it offers, avoiding any aspect of its implementation (including aid). Callers may rely on only what is stated in roles.

The Down rule ensures every design is a command hierarchy consisting of superiors and subordinates (bosses and workers). This rule produces a DAG (directed acyclic graph), so it's also known as the DAG rule. The Role rule requires that every method or object fulfill its role, doing no more and no less, precluding unexpected side effects. The Role and Down rules together force every design to be a role hierarchy. The Aid rule gives designers more flexibility by allowing public methods to help secretly with previously commanded duties, in addition to fulfilling their own roles. These rules don't apply to cross-cutting concerns.[2]

It's also helpful to think in terms of constraints on public behavior. Commands have one constraint: they must go down in the hierarchy (Down rule). Notices also have one constraint: they may only convey information (Identify rule).

Roles are important and warrant further discussion. A role is a purpose, responsibility, or duty. The Role rule requires that every object and method have a role that can be summarized in a few words, preferably containing only one verb. An example is: "Sends a fax." In an IDAR graph, the broadest role (greatest responsibility) is at the top, and the narrowest (most specialized) roles are at the bottom.

Inheritance creates a hierarchy, so why not use it? Unfortunately, inheritance creates a hierarchy of *categories*, which is less useful than a hierarchy of *roles*.

To see why, examine Figure 5, which shows a UML inheritance hierarchy for a CD player. `DiskMotor` and `LaserMotor` are subclasses of `Motor`, so they are in the motor category. You care little about their category, however, because you need to know which objects control these motors.

Likewise, `Laser`, `Motor`, and `Audio` are subclasses of `ElectronicDevice`, but that does not help because

you need to know which objects with broader roles command these devices. An inheritance hierarchy portrays categories, which are seldom helpful except in GUIs; it does not portray what you need to know—the hierarchy of roles.

Comparing UML to IDAR. An easy way to compare UML and IDAR is to follow an operation—for example, sending a fax. The sequence numbers on the commands and notices in Figure 4 indicate that after the user presses the `Send` button on the control panel, the `CtlPanel` object calls the `sendPressed` notice in `Fax`, which is clearly the main controller over the entire fax machine, and it commands `Send` to `sendFax`. Based on its high position in the hierarchy, you can see that `Send` handles the high-level aspects of sending faxes. It commands `Connect` to `connect` to the receiving machine, and `Connect` in turn commands `Modem` to take the phone off the hook via the `hookUpDn` method. After `Connect` gets the `dialTone` notice from `Modem`, it commands `Modem` to dial and waits for its answered notice. `Connect` then sends a connected notice back to `Send`. The figure also shows that `Send` commands `Scanner` to scan, and that data (the dashed arrows) will flow from `Scanner` into `ImageProc` and then into the `Modem` via the `pixelRow` and `xferBulk` notices. This graph reveals the structure of this software and how it works.

Figures 1, 2, and 6 are the UML class, communication, and sequence diagrams, respectively, for the same fax machine design. The communication diagram (Figure 2) has the same sequence numbers as the IDAR graph, making comparison easier.

Let's use the UML diagrams to show how a fax is sent. Which objects have primary roles? It's hard to tell. Which interactions among objects are the most important? It's hard to tell. Which objects are controllers versus workers? It's hard to tell. The best you can do is follow messages sequentially on the communication or sequence diagram, and even then it is difficult to determine which objects control which others, or which objects have broad versus narrow roles. UML fails to convey roles or their ranks, making designs hard to understand.

Benefits of IDAR Graphs

IDAR graphs provide several advantages over UML, two of which are predominant.

Ease of understanding. An IDAR graph is easier for developers to understand than the corresponding class, communication, and sequence diagrams in UML for the following reasons:

- The role hierarchy in IDAR is a generalized form of the AH (means-end abstraction hierarchy) employed in cognitive engineering,[7] which is known to impart understanding by means of the why-what-how triad. This triad consists of an object, its superiors, and its subordinates. It provides the following insights: the purposes of the object's superiors tell you *why* the object exists; the role of the object tells you *what* it does; and the purposes of its subordinates indicate *how* it works. UML lacks an AH, so it cannot tell you why an object exists or how it works.
- The hierarchy in an IDAR graph reveals which objects control which

Figure 3. Incomplete IDAR graph of a fax machine design.

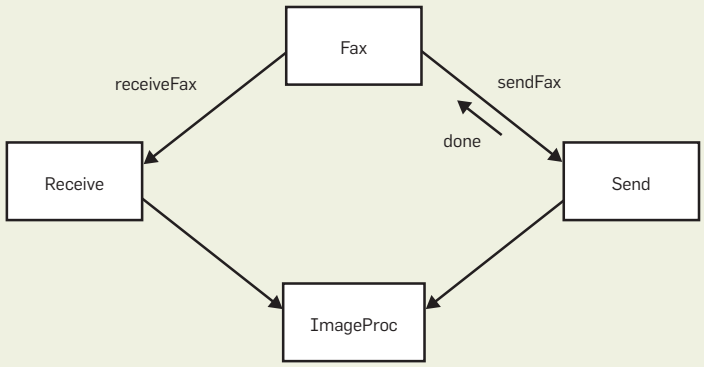
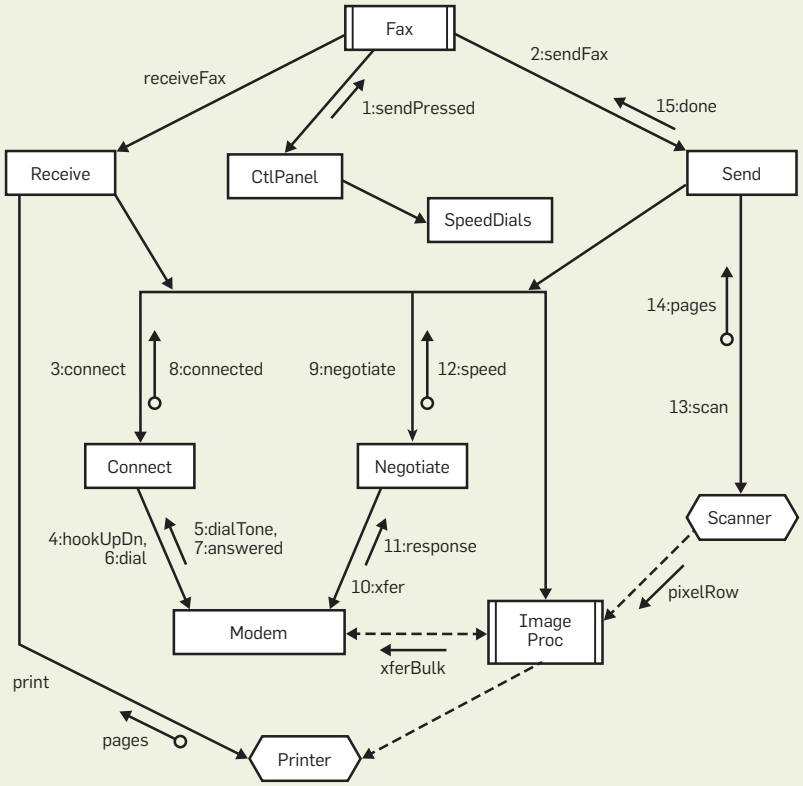


Figure 4. Complete IDAR graph of a fax machine.



others and, equivalently, which objects have broad versus narrow roles. In Figure 4, it is obvious that `Fax` is the top-level controller, and that `Send` and `Receive` are second-to-top-level controllers having rather broad roles. The corresponding UML diagrams conceal these helpful control relationships and role breadths.

▸ The subordinates of each superior form a closely related group, helping developers to associate functions with groups of objects. In Figure 4, it is clear that `Connect` and `Negotiate` are closely related workers under the same bosses, whereas UML conceals this tight affiliation. Unlike UML, IDAR reveals work groups.

▸ In an IDAR graph, command calls are more prominent than notice calls because they are more important. UML conceals degrees of importance.

▸ Throughout history, people have selected role hierarchies to represent organizations, indicating that they are most understandable.

▸ Research by experts in cognitive theory has shown that UML has severe problems with understandability ("cognitive effectiveness").[3] Specifically, UML has "alarmingly high levels of symbol redundancy and overload" and poor "visual discriminability." IDAR graphs were designed to avoid both of these problems.

▸ Other research has revealed that developers understand software design as an integrated interplay of its structure and behavior.[1] UML splits structure and behavior into two or more separate diagrams, reducing comprehension as developers are forced to flip back and forth between diagrams, attempting to integrate them mentally, which "unnecessarily strains developers' cognitive abilities." IDAR eliminates this wasteful mental effort by combining structure and behavior into one graph.

IDAR's resulting clarity should produce shorter learning curves and fewer misunderstandings and oversights, improving quality and shortening schedules.

Packages in UML may be nested, forming a hierarchy. This hierarchy, however, does not consist of roles, and the diagram inside each package is a network and not a hierarchy. Consequently, a hierarchy of packages doesn't improve understandability much. Subsystems in IDAR don't suffer from these disadvantages, and thus enjoy the full gain in understandability detailed here.

Note that organizational charts for corporations remain easy to understand regardless of their size. Because IDAR graphs are similar, they also should scale to any size and remain equally as easy to understand.

Resistance to messiness. A second important advantage of IDAR graphs over UML is they hinder the messiness (disorganization) that occurs when changes and enhancements are spliced into code with little regard for maintaining consistency of design. This claim is backed up by the following sensible constraints from the IDAR rules:

▸ The Identify rule prevents notices from initiating actions. In practice, it prevents a developer from creating spaghetti by scattering notice calls around, because notices are only allowed to convey needed information.

▸ The Down rule prevents a subordinate from commanding a superior.

▸ The Role rule prevents unexpected side effects, a common problem.

UML provides none of these defenses against messiness. For example, suppose you caused the `Modem` object in the fax machine to tell the `Receive` object to do something. This would add a line to the two UML diagrams (Figures 1 and 2) that is inconspicuous and acceptable. Doing so in the IDAR graph in Figure 4, however, would violate the Down rule because a subordinate would be commanding a superior. This is an example of the design decay that IDAR prevents.

Figure 5. Inheritance hierarchy of a CD player.

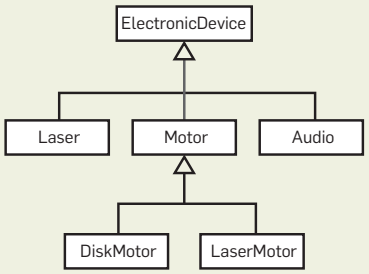
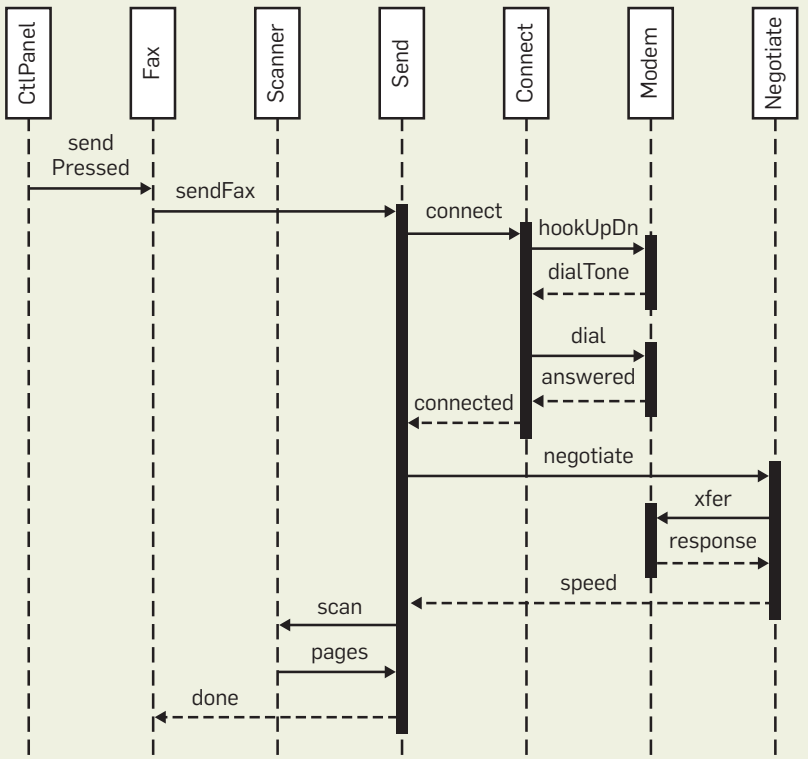


Figure 6. UML sequence diagram for sending a fax.



Limitations of IDAR Graphs

IDAR graphs do have the following limitations:

- **Object level.** IDAR is intended for object-level and subsystem-level design, so it's neither an ADL (architecture description language) nor a system modeling language. For modeling a system, OPM (Object Process Methodology)[1] is a strong contender.
- **Requires centralized control.** IDAR relies on control being organized as a command hierarchy, making it unsuitable for decentralized software with distributed control. The top levels of such software should be modeled in another way. At some level, however, the components of decentralized software are amenable to centralized control and can be designed using IDAR graphs like ordinary software.
- **Less expressive than UML.** UML can portray more views of designs than IDAR. For example, an IDAR graph is incapable of portraying transitions among states, deployment onto processors, or generalizations among classes. UML has diagrams for these and other aspects of design, and they should be employed when appropriate.

A Pilot Program

An important program was designed, coded, and deployed at Northrop Grumman using IDAR graphs. Responsible for calibration and testing of circuit boards and systems, the program is being used on the production line of an electronic product. We are forbidden from publishing this proprietary design, but we can say it has 23,000 lines of C++ code and is complex enough to have 38 classes, four subsystems, and 10 threads to handle various realtime matters. This medium-size program is not a toy.

Several people wrote and modified this program over several years, so it had become somewhat messy and was not even object oriented. The program consisted solely of tests, and I was charged with adding much nontest functionality to it, more than doubling its size. Thus, more than half of the code represents new design.

The existing code was refactored, creating objects conforming to the IDAR rules. I then designed and added the new capabilities in stages. During this process, unexpected requirements were added to the project, stress-testing the IDAR approach. IDAR graphs accomplished the following:

- Maintained clarity throughout design and implementation. Interactions among objects were so clear that any potential problems of misunderstandings among objects were avoided;
- Easily accommodated several changes and additions to the requirements. The hierarchy's clarity made it obvious where changes required by new features should be made;
- Enforced good organization;
- Did not impose excessive constraints on the design. The four rules provided enough flexibility that the design did not need to be contorted in order to satisfy them; and,
- Made design easier because the rules provided guidance. The top and bottom objects are easy to define, and defining objects between those anchor objects is not difficult. This ease of design was a surprise because imposing four rules would be expected to make designing more difficult, not easier.

Based on its results, those of us familiar with this effort believe the chief benefits of IDAR graphs over UML are their great clarity and enforcement of good organization. This pilot program was a strong success, and managers were pleased enough that they arranged for IDAR to be taught to the other software developers.

In addition to this pilot program, many trial designs have been created using IDAR graphs, and four life-size applications are described in *The IDAR Method of Software Design*.[5]

Conclusion

A hierarchy of roles appears to be essential for clearly portraying the design of any centralized organization, whether it consists of people or objects. The inability of today's object-oriented programming technology to represent this crucial kind of hierarchy is surprising, and perhaps its absence has been accepted based on the incorrect belief that an inheritance hierarchy is a suitable substitute.

An IDAR graph is clearer than UML for two main reasons: It reveals the hierarchy of roles and the breadths of those roles; and the triads (why-what-how) offer deeper insights into the nature of objects. UML cannot provide these. Given that IDAR graphs are clearer than UML, and that the four rules underlying them resist messiness, developers should produce fewer bugs when designing and implementing software using IDAR graphs. The result will be improved quality and timeliness.

This article contains enough information to enable readers to create designs using IDAR graphs. For more information, you can download the slides from a presentation at the IEEE Software Technology Conference in 2014.[4] Also, refer to *The IDAR Method of Software Design*,[5] which not only details this method (and related topics), but also includes the four life-size applications mentioned in this article.

Acknowledgments. I am indebted to Jim Ray for his consistent support of the IDAR method. Thanks to Jim Wilk and Dorothy Kennedy for their unswerving support. Sammy Messian managed the pilot program that used IDAR and values it for the fine results it produced. All four are managers at Northrop Grumman.

Related articles on queue.acm.org

UML Fever: Diagnosis and Recovery
Alex E. Bell
http://queue.acm.org/detail.cfm?id=1053347

Coding for the Code
Friedrich Steimann and Thomas Kühne
http://queue.acm.org/detail.cfm?id=1113336

Software Development with Code Maps
Robert DeLine, Gina Venolia, and Kael Rowan
http://queue.acm.org/detail.cfm?id=1831329

References

1. Dori, D. Why significant UML change is unlikely. *Commun. ACM 45*, 11 (2002), 82–85.
2. Kiczales, G., et al. Aspect-oriented programming. *Proceedings of the 11th European Conference on Object-Oriented Programming*: (1997), 220–242.
3. Moody, D., van Hillegersberg, J. Evaluating the visual syntax of UML: An analysis of the cognitive effectiveness of the UML family of diagrams. *Software Language Engineering.* Springer-Verlag, Berlin, Heidelberg, 2009, 16–34.
4. Overton, M. Command graphs: a significant improvement to OOP/OOD. *IEEE Software Technology Conference*, 2014; http://conferences.computer.org/stc/2014/index.html or http://www.ieee-stc.org.
5. Overton, M. 2014. *The IDAR Method of Software Design.* CreateSpace, Seattle, WA, 2014.
6. Rumbaugh, J., Jacobson, I., Booch, G. *Unified Modeling Language Reference Manual*, 2nd ed. Addison-Wesley, Boston, MA, 2004.
7. Vicente, K. *Cognitive Work Analysis.* Lawrence Erlbaum Associates, Mahway, NJ, 1999, 174–176.

Mark Overton is a software engineer at Northrop Grumman working on various parts of software-defined radios. He previously worked at HP, contributing to both the architecture and implementation of all-in-one printers, particularly their scanners.

DOI:10.1145/3052942

Article development led by acmqueue
queue.acm.org

Expert-curated guides to the best of CS research.

Research for Practice: Tracing and Debugging Distributed Systems; Programming by Examples

THIS INSTALLMENT OF Research for Practice covers two exciting topics in distributed systems and programming methodology. First, **Peter Alvaro** takes us on a tour of recent techniques for debugging some of the largest and most complex systems in the world: modern distributed systems and service-oriented architectures. The techniques Alvaro surveys can shed light and order amid the chaos of distributed call graphs. Second, **Sumit Gulwani** illustrates how to program without explicitly writing programs, instead synthesizing programs from examples! The techniques Gulwani presents allow systems to "learn" a program representation from illustrative examples, allowing nonprogrammer users to create increasingly nontrivial functions such as spreadsheet macros. Both of these selections are well in line with RfP's goal of accessible, practical research; in fact, both contributors have successfully transferred their own research in each area to production, at Netflix and as part of Microsoft Excel. Readers may also find a use case.

As always, our goal in this column is to allow our readers to become experts in the latest topics in computer science research in a weekend afternoon's worth of reading. To facilitate this process, we have provided open access to the ACM Digital Library for the relevant citations from these selections so you can enjoy these research results in full. Please enjoy!

—Peter Bailis

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

OK, But Why? Tracing and Debugging Distributed Systems

By Peter Alvaro

Large-scale distributed systems can be a nightmare to debug. Unlikely events (for example, a server crashing or a process taking too long to respond to a request) are commonplace at the massive scale at which many Internet enterprises operate. State-of-the-art monitoring systems can help measure the frequency of these anomalies but do little to identify their root causes. Pervasive logging may record events of interest at appropriate granularity, but correlating events across the logs of large numbers of machines is prohibitively difficult.

Distributed tracing systems overcome many of these limitations, mak-

ing it easier to derive high-level explanations of end-to-end interactions spanning many nodes in distributed computations. But there is no free lunch. Broadly speaking, large-scale tracing systems impose on adopters both an instrumentation burden (the effort that goes into tweaking existing code to add instrumentation points or to propagate metadata, or both) and an overhead burden (the runtime cost of trace capture and propagation). The collection of papers chosen here illustrates some strategies for ameliorating these burdens, as well as some creative applications for high-level explanations.

Tracing with Context Propagation

Sigelman, B.H., Barroso, L.S., Burrows, M., Stephenson, P., Plakal, M., Beaver, D., Jaspan, S., Shanbhag, C.
Dapper, a large-scale distributed systems tracing infrastructure, 2010; http://research.google.com/pubs/pub36356.html

Dapper represents some of the "early" industrial work on context-based tracing. It minimizes the instrumentation burden by relying on Google's relatively homogeneous infrastructure, in which all code relies on a common RPC (remote procedure call) library, threading library, and so on. It minimizes the overhead burden by selecting only a small sample of requests at ingress and propagating trace metadata alongside requests in order to ensure that if a request is sampled, all of the interactions that contributed to its response are sampled as well.

Dapper's data model (a tree of nested *spans* capturing causal and temporal relationships among services participating in a *call graph*) and basic architecture have become the de facto standard for trace collection in industry. Zipkin (created at Twitter) was the first open-source "clone" of Dapper; Zipkin and its derivatives (including the recently announced Amazon Web Services X-Ray) are in widespread use today.

Mace, J., Roelke, R., Fonseca, R.
Pivot Tracing: Dynamic causal monitoring for distributed systems. In *Proceedings of the 25th Symposium on Operating Systems Principles* (2015), 378–393; http://cs.brown.edu/~rfonseca/pubs/mace15pivot.pdf

Dapper was by no means the first system design to advocate in-line context propagation. The idea goes back at least as far as Xtrace, which was pioneered by Rodrigo Fonseca at UC Berkeley. Fonseca (now at Brown University) is still doing impressive work in this space. Pivot Tracing presents the database take on low-overhead dynamic tracing, modeling events as tuples, identifying code locations that represent sources of data, and turning dynamic instrumentation into a query planning and optimization problem. Pivot Tracing reuses Dapper/Xtrace-style context propagation to allow efficient correlation of events according to causality. Query the streams!

Trace Inference

Chow, M., Meisner, D., Flinn, J., Peek, D., Wenisch, T.F.
The mystery machine: End-to-end performance analysis of large-scale Internet services. In *Proceedings of the 11th Usenix Conference on Operating Systems Design and Implementation* (2014), 217-231; https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-chow.pdf

What about enterprises that can't (or just don't want to) overcome the instrumentation and overhead burdens of tracing? Could they reconstruct causal relationships after the fact, from unstructured system logs? The mystery machine describes a system that begins by liberally formulating hypotheses about how events across a distributed system could be correlated (for example, Is one a cause of the other? Are they mutually exclusive? Do they participate in a pipelined computation?) and then mines logs for evidence that contradicts existing hypotheses (for example, a log in which two events A and B are concurrent immediately refutes a hypothesis that A and B are mutually exclusive). Over time, the set of hypotheses converges into models of system interactions that can be used to answer many of the same questions.

New Frontiers

Alvaro, P., Andrus, K., Sanden, C., Rosenthal, C., Basiri, A., Hochstein, L.
Automating failure-testing research at Internet scale. In *Proceedings of the 7th ACM Symposium on Cloud Computing* (2016), 17–28; https://people.ucsc.edu/~palvaro/socc16.pdf

The *raison d'etre* of the systems just described is understanding the causes of end-to-end latency as perceived by users. Armed with detailed "explanations" of how a large-scale distributed system produces its outcomes, we can do so much more. My research group at UC Santa Cruz has been exploring the use of explanations of "good" or expected system outcomes to drive fault-injection infrastructures in order to root out bugs in ostensibly fault-tolerant code. The basic idea is that if we can explain how a distributed system functions in the failure-free case, and how it provides redundancy to overcome faults, we can better understand its weaknesses.

This approach, called lineage-driven fault injection (LDFI), originally relied on idealized, fine-grained data provenance to explain distributed executions (see our previous paper, "Lineage-driven Fault Injection," by Peter Alvaro, Joshua Rosen, and Joseph M. Hellerstein, presented at SIGMOD 2015). This more recent paper describes how the LDFI approach was adapted to "snap in" to the microservice architecture at Netflix and to build rich models of system redundancy from Zipkin-style call-graph traces.

Conclusion

Despite the fact that distributed systems are a mature research area in academia and are ubiquitous in industry, the art of debugging distributed systems is still in its infancy. It is clear that conventional debuggers—and along with them, conventional best practices for deriving ex-

A key challenge in programming by example is to develop an efficient search algorithm that can discover a program that is consistent with the examples.

planations of computations—must be replaced, but it is too soon to say which approaches will come to dominate. Industry has led in the design and particularly in the popularization of large-scale tracing systems in reaction to a practical need: understanding the causes of user-perceived latency for online services. As these systems become common infrastructure, we will find that this use case is only the tip of the iceberg. The ability to ask and answer rich "why" questions about distributed executions will continue to engender new research that improves the consistency, predictability, and fault tolerance of massive-scale systems.

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

Programming By Examples

By Sumit Gulwani

Programming by examples (PBE) is the task of synthesizing or searching for a program from an underlying program space that satisfies a given set of input-output examples.

A key challenge in PBE is to develop an efficient search algorithm that can discover a program that is consistent with the examples. Various *search techniques* have been developed, including deductive methods, use of constraint (SAT/SMT) solvers, smart heuristics for enumerative search, and stochastic search. Another key challenge in PBE is to deal with the ambiguity in intent specification, since there are many programs that satisfy the given examples but not the user's intent. *Ranking techniques* are used to predict an intended program from within the set of programs consistent with the examples. *Interaction techniques* are used in a refinement loop to converge to an intended program.

PBE has varied applications. It allows end users, 99% of whom are non-programmers, to create small scripts for automating repetitive tasks from examples. It facilitates software development activities, including program refactoring, superoptimization, and test-driven development. The following sample of recently published work addresses applications from different domains while employing different kinds of search and disambiguation algorithms.

Data Manipulation Using Back-Propagation

Gulwani, S.
Automating string processing in spreadsheets using input-output examples. In *Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, (2011), 317–330; https://www.microsoft.com/en-us/research/publication/automating-string-processing-spreadsheets-using-input-output-examples/

The first paper describes a technology for automating string transformations such as converting "FirstName LastName" to "LastName FirstName." This technology was released as the Flash Fill feature in Microsoft Excel (https://youtu.be/w-k9WjRJvIY). The paper motivates the design of an expressive DSL (domain-specific language) that is also restricted enough to allow for efficient search. The inspiration came from studying spreadsheet help forums, wherein end users solicited help for string transformations, while describing their intent using examples. The paper describes a domain-specific search algorithm that achieves real-time efficiency, breaking from the previous community tradition of reducing the search problem to querying an off-the-shelf general-purpose constraint solver. The latter, while allowing quicker prototyping, lacks the effectiveness of a custom solution.

The paper also gives first-class treatment to dealing with ambiguity, instead of requiring a larger number of examples, thus improving usability and trust. The search algorithm returns a huge set of programs (represented succinctly) that satisfy the examples, and a ranking function that prefers small programs with few constants is used to guess an intended program.

The success of Flash Fill inspired a wave of interest in both academia and industry for developing PBE technologies for other domains, including number/date transformations, tabular data extraction from log files/Web pages/JSON documents, and reformatting tables. With data scien-

tists spending 80% of their time transforming and cleaning data to prepare it for analytics, PBE is set to revolutionize this space by enabling easier and faster data manipulation.

Polozov, O., Gulwani, S.
FlashMeta: A framework for inductive program synthesis. In *Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*, 107–126; https://www.microsoft.com/en-us/research/publication/flashmeta-framework-inductive-program-synthesis/

Developing and maintaining an industrial-quality PBE technology is an intellectual and engineering challenge, requiring one to two person-years. The second paper observes that many PBE algorithms are natural fallouts of a generic meta-algorithm and the logical properties of operators in the underlying DSL. The meta-algorithm is based on back-propagation of example-based constraints over the underlying DSL, reducing the search problem over program expressions to simpler problems over program subexpressions. The meta-algorithm can be implemented once and for all. The operator properties relate to its inverse semantics and can be reused across multiple DSLs.

This allows for construction of a synthesizer generator that takes a DSL and semantic properties of operators in the DSL and generates a domain-specific synthesizer. With such a generator, PBE technologies become modular and maintainable, facilitating their integration in industrial products. This framework has been used to develop many PBE tools that are deployed in several industrial products, including Microsoft Operations Management Suite, PowerShell 3.0, and the Cortana digital assistant.

Hempel, B., Chugh, R.
Drawings Using Prodirect Manipulation Semi-automated SVG programming via direct manipulation. In *Proceedings of the 29th Annual ACM Symposium on User Interface Software and Technology*, (2016), 379–390; https://arxiv.org/abs/1608.02829

PBE can bring together the complementary strengths of direct GUI (graphic user interface) manipulation (mouse- and menu-based) and programmatic manipulation of digital artifacts such as spreadsheets, images, and animations. While direct manipulation enables easy manipulation of a concrete object, programmatic manipulation allows for much more freedom and reusability (but requires skill). This paper bridges this gap by proposing an elegant combined approach, called prodirect manipulation, that enables creation and modification of programs using GUI-based manipulation of example objects for the domain of SVG (scalable vector graphics).

The user draws shapes, relates their attributes, and groups and edits them using the GUI, and the drawing is kept synchronized with an underlying program. The various GUI-based actions translate to constraints over the example drawing. Constraint solvers are used to generate candidate modifications to the underlying program so that the resultant program execution generates a drawing satisfying those constraints. Smart heuristics are used to select an intended modification from among the many solutions. A skilled user can edit the resulting program during any step to refine the automatically generated modification or to implement some new functionality.

Superoptimization Using Enumerative Search

Phothilimthana, P.M., Thakur, A., Bodík, R., Dhurjati, D.
Scaling up superoptimization. In *Proceedings of the 21st International Conference on Architectural Support for Programming Languages and Operating Systems*, (2016), 297–310; https://people.eecs.berkeley.edu/~mangpo/www/papers/lens-asplos16.pdf

PBE can be used to solve the general problem of program synthesis from an arbitrary specification, given an oracle that can produce counterexamples where the synthesized artifact does not match the intended behavior. This paper uses this reduction, also referred to as CEGIS (counterexample-guided inductive synthesis), to advance the state of the art in superoptimization, which is the problem of finding an optimal sequence of instructions for a given code fragment.

The heart of the paper is a novel PBE algorithm based on enumerative search that considers programs in the underlying state space in order of increasing size. The algorithm leverages an elegant memorization strategy, wherein it computes the set of programs of bounded size that satisfy a given collection of examples and incrementally refines this set with more examples in the next iteration. The programs are represented succinctly using their behavior on the example states. The algorithm also leverages a powerful meet-in-the-middle pruning technique based on bidirectional search, where the candidate programs are enumerated forward from input states, as well as backward from output states.

The paper further studies the strengths and weaknesses of different search techniques, including enumerative, stochastic, and solver-based, and shows that a cooperative search that combines these is the best.

The Future

PBE can be regarded as a form of machine learning, where the problem is to learn from very few examples and over a rich space of programmatic functions. While past developments in PBE have leveraged logical methods, can recent advances in deep learning push the frontier forward? Another exciting direction to watch out for is development of natural-language-based programming interfaces. Multimodal programming environments that would combine example- and natural-language-based intent specification shall unfold a new era of programming by the masses.

Acknowledgments. Thanks to Ravi Chugh, Phitchaya Mangpo Phothilimthana, and Alex Polozov for providing useful feedback on this article.

Sumit Gulwani leads a research and engineering team at Microsoft that develops program synthesis technologies for data wrangling and incorporates them into real products. His programming-by-example work led to the Flash Fill feature in Microsoft Excel used by hundreds of millions of people.

DOI:10.1145/3098342

Explore the limits of using the computer to imagine yourself as whomever or whatever you want to be.

BY D. FOX HARRELL AND CHONG-U LIM

Reimagining the Avatar Dream: Modeling Social Identity in Digital Media

COMPUTER SCIENCE HAS long been intertwined with society's technological dreams. The dream of automated homes relates to ubiquitous computing, just as the dream of sentient machines relates to artificial intelligence (AI). Another of society's dreams could be called the "Avatar Dream," a culturally shared vision of a future in which, through the computer, people can become whomever or whatever we want to be.

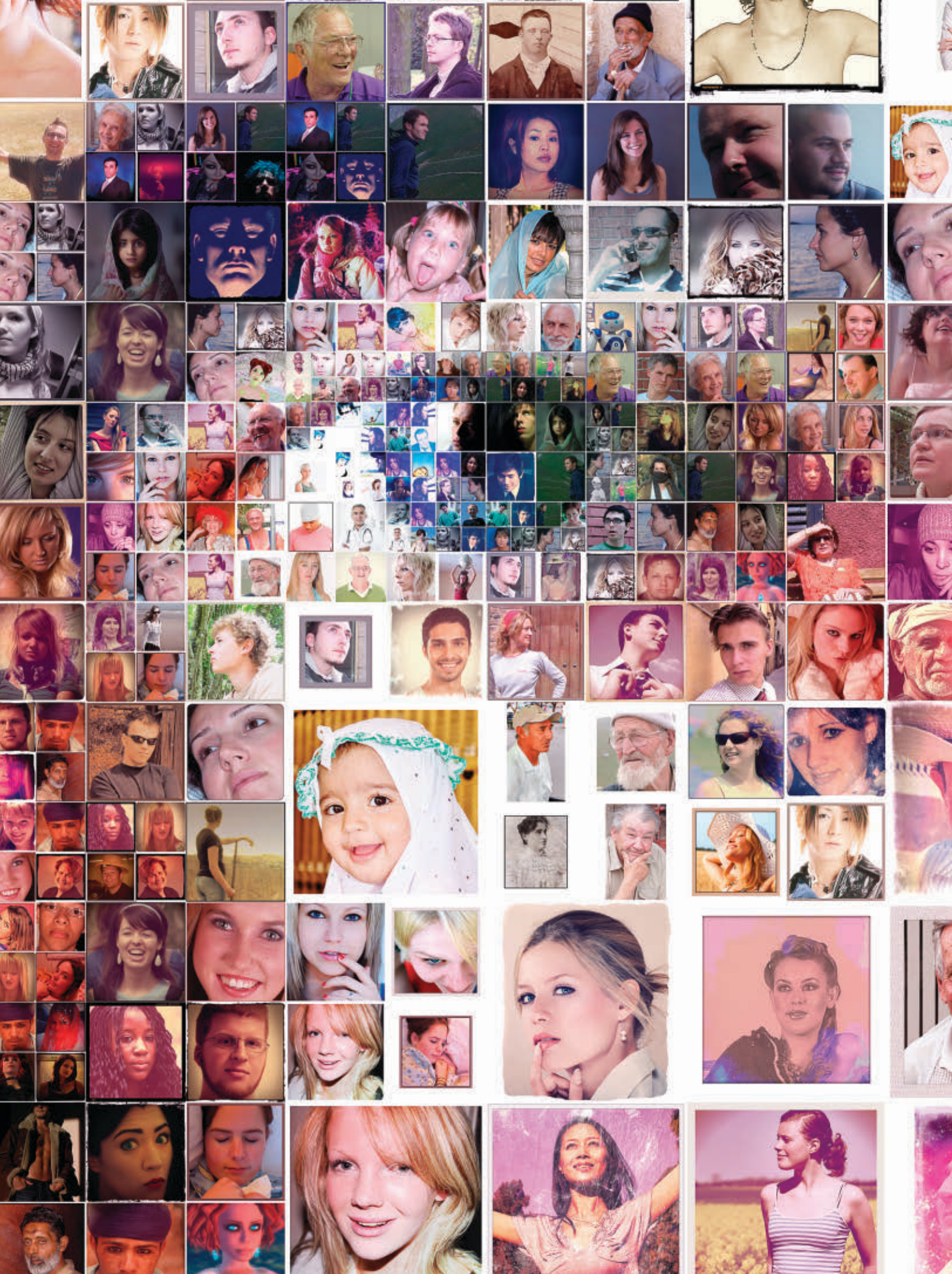
Our focus is not, however, on simply developing the technologies that can support the Avatar Dream. We instead argue for the need to reimagine the Avatar Dream where the potential social and cultural impacts of virtual identities are considered intrinsic to the engineering practices of inventing them. This article is an overview of our endeavors toward this end and several key results and findings demonstrating our reimagining of the Avatar Dream.

We first define the Avatar Dream, describe its current state, and motivate our work by considering problems with current virtual identity systems. We then provide a theoretical framework for characterizing the relationships between virtual and real-world (physical) identities necessary for precise articulation of the sociocultural phenomena we study. The remainder of the article focuses on two key endeavors. The first is our computational approach to analyzing sociocultural identity phenomena in virtual identity systems; these techniques support engineers developing systems that avoid or combat negative phenomena (such as discrimination and prejudice). For example, we use AI to reveal how sexist and racist biases are embedded in a bestselling computer game, demonstrating an approach applicable to other systems. The second is our approach to simulating sociocultural identity phenomena; it includes developing technologies (such as an authoring platform called *Chimeria* and interactive narratives made using it that convey how individuals navigate so-

» key insights

- **Using virtual identity technologies to just look like someone different from yourself is not enough to understand the experiences of someone different from yourself.**
- **Researchers and developers should practice methods for engineering virtual identity technologies such that modeling their social and cultural implications is intrinsic to the practice of inventing them.**
- **Analyzing social phenomena involving virtual identity systems, and designing new virtual identity authoring tools and applications to simulate social phenomena, enables researchers and developers to better support users' needs for more powerfully nuanced forms of what we call "technologies of self-imagination."**

ILLUSTRATION BY CHARIS TSEVIS



cial categories). These technologies support the aims of creating richer experiences for users, helping educate diverse learners, and conducting social-science research studies. We conclude by reflecting on this reimagined Avatar Dream.

Defining the Avatar Dream. The Avatar Dream has two elements. One is technical, enabling users to control a virtual surrogate for themselves in a virtual world. These computational surrogate selves are often computer-generated images (CGI) but can range from text descriptions in games or social media to virtual representations that engage all the senses in futuristic virtual reality environments. The second is experiential, enabling users of these virtual surrogate selves to have experiences beyond those they encounter in the physical world, ranging from having new abilities to better understanding the experiences of others (such as of another gender or even another type of creature).

The current expression of the Avatar Dream in many contemporary societies includes using virtual identities to communicate, share data, and interact in computer-based (virtual) environments. We can thus view a manifestation of this dream as the avatar,[a] or user-controlled representations of self in virtual environments. Neal Stephenson's 1993 novel *Snow Crash* provides a science-fictional vision of avatars as technologies to reimagine one's self. Stephenson wrote, "Your avatar can look any way you want it to, up to the limitations of your equipment. If you're ugly, you can make your avatar beautiful. If you've just gotten out of bed, your avatar can still be wearing beautiful clothes and professionally applied makeup. You can look like a gorilla or a dragon or ..."[36] Another manifestation of the dream is the social-media profile. Even in the heady days of the 1990s, it was understood that these profiles were distinct from our physical selves. In one of the most venerable social media sites—The Well[b]—where users' real names were available, self imagination was a central part of the appeal. The Well purportedly offered the freedom of projecting whatever personality you wished, along with the intriguing possibility of highlighting subtle variations of your character."[14] Virtual identities in digital media, including virtual worlds, videogames, and social media, all hearken back to the Avatar Dream.

This article introduces the term "box effects" to refer to the experiences of people that emerge from the failure of classification systems.

Problems with current virtual identity systems. The need for socio-culturally informed virtual identity research is urgent; nearly everyone today has social media accounts for connecting with friends, e-commerce accounts for shopping, and videogame characters for playing. One problem in designing virtual identity systems is the need for intuitive, appropriate, and robust tools for avatar creation and customization. Just being able to edit avatar appearance is not enough to support peoples' needs for self-expression when using virtual identities. It is important that avatars embody enough sociocultural nuance to express facial expressions, body language, gait, discourse style, and personality. Users should not need to program all these forms of self-expression from scratch and should, instead, be able to express themselves with their avatars through simple interfaces. Researchers Joseph Bates, Michael Mateas, Brenda Laurel, Ken Perlin, and others[1,24,31,32] have developed such tools and virtual worlds for their deployment. Furthermore, some user groups are underrepresented and/or unfairly stigmatized in virtual environments. Social-network usage percentages are often higher for underrepresented U.S. racial groups and women than for white individuals and men; 90% of African-American females and 40% of white females are depicted as victims of violence in games.[7,8] Such phenomena are not only embedded in systems but also enacted by users. Julian Dibbell's classic 1996 article "A Rape in Cyberspace,"[6] describing how a user's female avatar in a text-based environment was taken over and subjected to violent acts, presaged such negative repercussions of

a Avatars, agents, and player characters each have slightly different definitions, although here the term "avatar" is used as the most general one; for us, the term "virtual identity" is even more general, including social-media profiles, e-commerce accounts, and other computational representations of users.

b "The Well" is short for "Whole Earth 'Lectronic Link."

the Avatar Dream. Such observations undergird an important motivation for our work: Virtual identities can and should better serve the needs of diverse users.

Defining Physical, Virtual, and Blended Identities

Before describing our computational approaches, we clarify the terminology we use, in light of the ambiguities in terms like "real" and "virtual" identity.

Physical identities. For identities in the physical world, our focus includes but also goes beyond the notions often associated with identity like gender, race, and age. We are instead concerned with users' identity experiences, which are informed by history, culture, and values in the physical world. While it is impossible to give a comprehensive definition of everything that affects people's identities in the real world, we offer a simplified overview useful for the discussion in this article. Identity experiences are informed by history, culture, and values that exist in the physical world and manifest in the ways people behave. Identity experiences include cognitively grounded,[c] material (such as resources), and social (such as power relationships) aspects.

Virtual identities. Our definition of virtual identities focuses on their components as technical systems. A virtual identity in this model is characterized by its data structures and algorithms that are deployed to provide both representation and control to the user. In videogames, a common virtual identity type is the player character, or avatar, players take control of. Computers have long been a medium for humans to create such "second" selves[37] or even many selves. Such use of computers as digital media for self-representation has become even more pervasive with the proliferation of increasingly immersive virtual environments and game worlds that enable interactions among multiple players at the same time. Each of these other selves can be viewed as an "externalization of self"[37] beyond our physically embodied selves. Hence, despite the fact that avatars are sometimes narrowly regarded as mere technically constructed visual artifacts, a more expansive view holds that virtual identities serve as important ways through which people represent or express themselves.

Blended identities. We have highlighted aspects of the physical and virtual world identities we are seeking to better understand. However, rather than considering each of them individually, this research is based on their interrelationships. We particularly seek to consider how values are socially and culturally constructed, enacted, and manipulated via blends between physical and virtual identities.

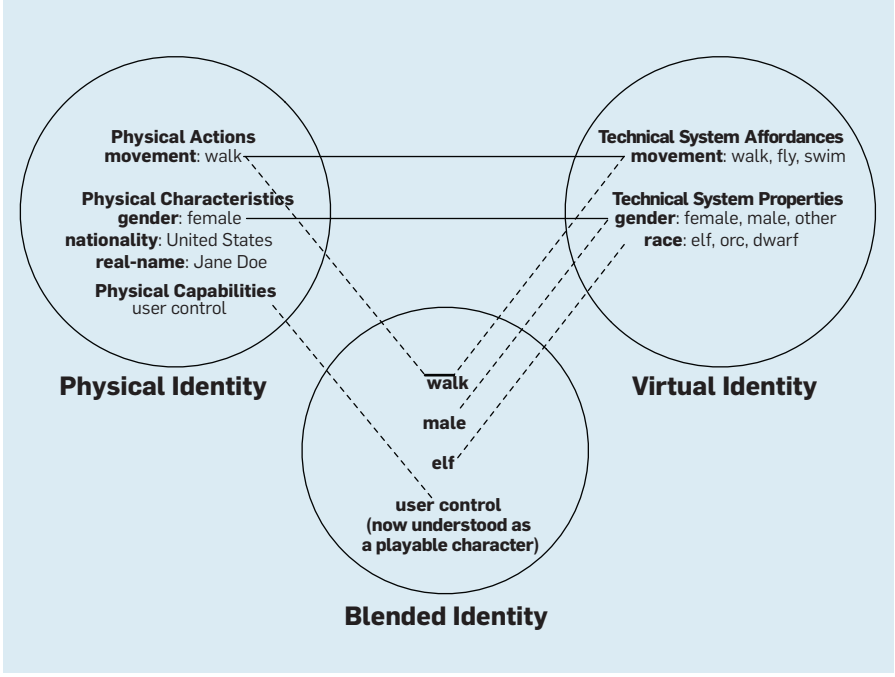
James Gee's notion of the "projective identity"[10] is a useful starting point. His use of the term refers to the reflection of players' values in how they make sense of their avatars. However, while it provides a high-level descriptive characterization of identity, it is insufficient for our needs. It fails to capture important structural phenomena like mappings between a user's actions and a virtual identity being controlled. We thus enrich Gee's model with an approach from cognitive science called "conceptual blending theory"[9] in which blending is a proposed cognitive mechanism by which humans integrate concepts.[d] We thus use Harrell's notion of a "blended identity"[17] in which aspects of a player's physical identity (such as preferences, control, appearance, and understanding social categories) are selectively projected[9] with aspects of the virtual identity onto a blended identity, integrating and elaborating aspects of each (see Figure 1). Blended identities can be studied in light of the interrelationship between both worlds. We later give examples of our approaches for analyzing blended identities computationally to reveal how physical-world values can be both embedded in virtual identity systems and enacted by virtual identity users.

It is also important to note that a single blended-identity user is not restricted to a single virtual world platform but often has multiple different virtual identities and behaves differently depending on the platform, what we term "cross-platform identities."

c Embodiment recognizes that "cognition depends on the kind of experience that comes from having a body."[38] "Situatedness" highlights the role of the context or social situation for cognitive processing.[25] Distributed cognitive processes are projected onto aspects of environments (such as artifacts[22]), members of social groups, and time (such as animated CGI[4]).

d Conceptual blending theory has been criticized for using ad hoc, overly broad explanations.[11] The theory has also been defended as being supported by a convergence of data from psychology, AI, sociology, literature, and philosophy and focus on exemplary phenomena resulting in a theory that comprehensively covers even challenging cases.[5]

Figure 1. A blended-identity diagram; cross-space mappings reveal aligned characteristics of the physical and virtual identities (called "input spaces" in conceptual blending theory).



For each platform, there is a projection of a certain set of identity features from the user.

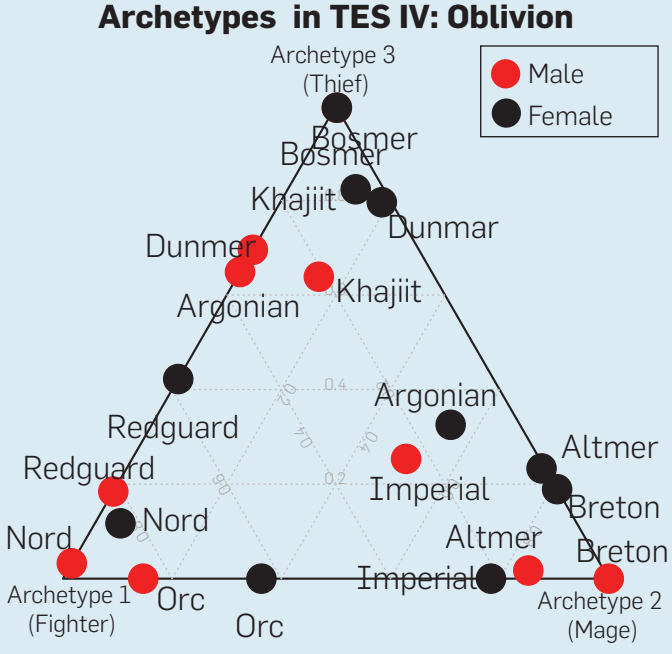
Box effects. This article introduces the term "box effects" to refer to the experiences of people that emerge from the failure of classification systems. Box effects include, but are not limited to, such related phenomena as stereotypes, social biases, stigmas, discrimination, prejudice, racism, and sexism. In the phrase "classification system," as used here, "system" does not refer to a technical computer system but rather to the notion of classification put forward by Geoff Bowker and Susan Leigh Star, who said, "A classification is a spatial, temporal or spatiotemporal segmentation of the world," and a classification system "is a set of boxes (metaphorical or literal) into which things can be put in order to then do some kind of work—bureaucratic or knowledge production."[3] Our term "box effects" is useful because there is no common term for social phenomena with roots in classification systems. It is also useful to have an overarching term like box effects because specific terms (such as stereotyping and prejudice) have multiple definitions in different academic disciplines, as well as general, popular uses.

Better understanding existing systems and designing new systems that take box effects into account cannot be accomplished within a narrowly technical vision of the Avatar Dream. Our reimagined Avatar Dream recommends addressing box effects in several ways. First, developers must understand the ways box effects from the physical world persist in virtual environments in terms of system structure. Second, we must look at how box effects emerge from users' behaviors in relation to categories from the physical world; for example, users might create stereotypical female characters in a game based on their own preconceptions about categories of physical-world gender. Finally, beyond needing techniques to understand and identify box effects in virtual environments, developers need tools to support addressing them (such as through more nuanced models of identity in games and interactive narratives) like those we provide with the *Chimeria* platform.

Figure 2. Initial racial attribute values in *The Elder Scrolls IV: Oblivion*; interesting discrepancies are highlighted between races (blue) and genders (red).

Attribute	Altmer		Argonian		Bosmer		Breton		Dunmer		Imperial		Khajiit		Nord		Orc		Redguard	
	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F	M	F
Strength	30	30	40	40	30	30	40	30	40	40	40	40	40	30	50	50	45	45	50	40
Intelligence	50	50	40	50	40	40	50	50	40	40	40	40	40	40	30	30	30	40	30	30
Willpower	40	40	30	40	30	30	50	50	30	30	30	40	30	30	30	40	50	45	30	30
Agility	40	40	50	40	50	50	30	30	40	40	30	30	50	50	40	40	35	35	40	40
Speed	30	40	50	40	50	50	30	40	50	50	40	30	40	40	40	40	30	30	40	40
Endurance	40	30	30	30	40	30	30	30	40	30	40	40	30	40	50	40	50	50	50	50
Personality	40	40	30	30	30	40	40	40	30	40	50	50	40	40	30	30	30	25	30	40
Luck	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50

Figure 3. An archetypal analysis ternary-plot of statistical attribute allocations in *Oblivion*; note, at Archetype 3, the Male Bosmer (red) is behind the Female Bosmer marker (black).



Analyzing Virtual Identity Phenomena

Since 2010, Harrell has led a research initiative to better understand, design, and develop virtual identities called the "Advanced Identity Representation (AIR) Project." We use the term "advanced" with humility. Our computational systems cannot completely express the nuances of physical-world identities. Yet they provide advances over current systems in their emphasis on physical-world identity categories social scientists have identified as important for modeling user experiences (such as of gender, race, and ethnicity) in addition to personality, values, and preferences. In this way, we can achieve advances in modeling phenomena (such as box effects), goals that increase the expressive range and utility of virtual identities. Our resultant systems are often necessarily reductive (from vast real-life experience to more limited data structures and algorithms) in order to be implementable. Yet this reduction is done knowingly with the benefit of expanding the expressive capacity of computational systems to address social-identity phenomena. As mentioned earlier, this work includes two types of computational modeling: analyzing sociocultural phenomena involving virtual identities using AI/machine learning techniques; and simulating sociocultural identity phenomena. We next describe our work in socio-

cultural phenomena involving virtual identities, including values built into systems (embedded) and patterns discovered (emergent) from users.

Limitations of current approaches. There are many survey-based studies of how behavioral patterns by users in virtual environments replicate identity experiences based in the physical world, but they have notable limitations. While useful for assessing subjective notions of identities expressed by users (such as preferences), self-reported survey data is often difficult to evaluate and subject to survey bias.[2] Also, while useful for understanding certain user characteristics (such as articulated reasons for choosing among options), some aspects of users' experiences (such as tacit knowledge) cannot be articulated, are intrusive, or are mentally or physically strenuous for participants or interviewers, and are often better suited to automated data collection and analysis.

New approaches to analyzing blended identities. Here, we present our approach to using AI for computationally modeling categorization, focusing on the novel use of these algorithmic techniques to address aspects of social identity often deemed challenging to quantify due to the subjectivity of their manifestations.

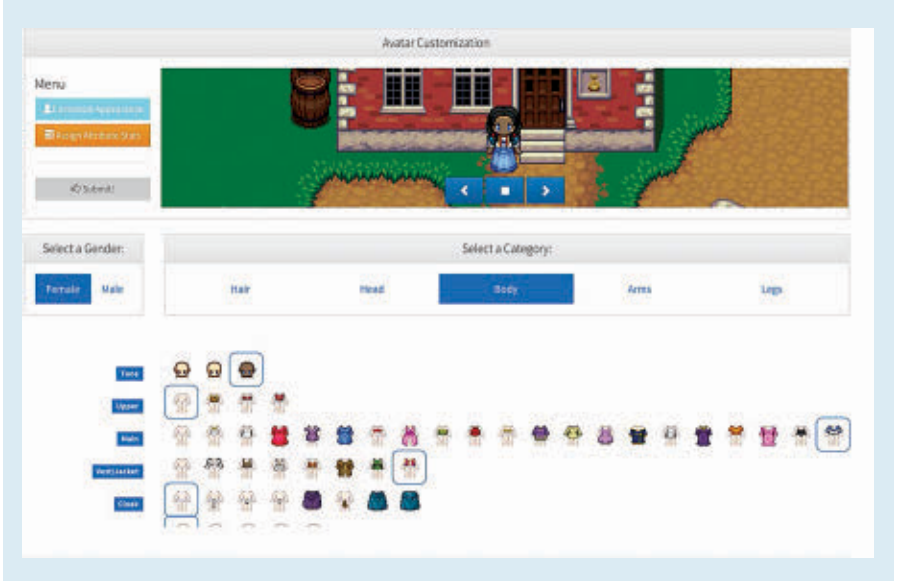
Clustering for computational categorization. In the field of AI, cluster analysis, or "clustering," is the algorithmic process of grouping observations into categories (clusters) based on measurements of similarity between individual observations. An observation refers to a data point within the set of observations (dataset) and is represented through measurements of one or more of its properties, or "features." Observations often correspond to players, each characterized by features that describe aspects of their physical identities (such as biological sex) or virtual identities (such as avatar appearance and behaviors). Cluster results are based on the definition of a cluster, or which features determine membership, and the similarity measure of observations, or how features are used to measure the similarity or differences between observations. Clustering is appropriate for our aims, as it enables quantitative analysis and can reveal new or unknown categories.

There are many different approaches to clustering users according to their behavior in systems using virtual identities (such as videogames).[27] Our own experience includes investigating techniques like the k-means algorithm, principal component analysis, non-negative matrix factorization, and archetypal analysis for social analysis using virtual identities.[28] Our focus here is on archetypal analysis, identifying a set of key observations in a dataset called "archetypes," or certain external points in the dataset. Other observations can be represented as mixtures of these archetypes. This approach provides insight into definable characteristics of highly distinctive virtual representations or behaviors. It is also useful for revealing patterns of users' behavior that either conform to or subvert conventions.[28] In addition, as it provides a visible way to identify marginalized individuals, defined as observations notably distant from all archetypes,[e] we find archetypal analysis to model more effectively than other clustering techniques.[27]

Analyzing systems: Revealing values embedded in technologies. We created a system called *AIRvatar* to perform fine-grain telemetric data collection of users' virtual identify creation and customization behaviors. Named for an initiative called the Advanced Identity Representation (AIR) Project, *AIRvatar* also implements the clustering approaches described earlier and has been deployed in videogame and social media data systems.[28,29] Here, we provide an example of how we reveal race and gender stereotypes through archetypal analysis. We analyzed the critically acclaimed and commercially successful videogame *The Elder Scrolls IV: Oblivion* that features an exemplary diverse roster of player character types; we cite it here as one example of a general phenomenon, not to single it out as especially inequitable to diverse users in contrast to other games. The upshot is we found and validated several forms of race and gender inequity. Players may choose to play as one of 10 different races available. Though fictional, some of the races are based on physical-world national, racial, and ethnic groups through their textual descriptions and visual appearances; for example, Redguards represent people broadly of African descent (a single country or subgroup is not suggested); Nords represent Norwegians; Bretons represent French people; and so on. Player characters possess eight attributes representing abilities (such as strength and intelligence). Based on the player's choice of race and gender, these attributes are initialized with a set of default values. In previous work,

e Drawing on definitions from sociological and cognitive theories of classification and categorization.[3]

Figure 4. A screenshot of the interface of *Heroes of Elibca*, a custom avatar-creation system implemented in AIRvatar.



Harrell observed several forms of racial and gender inequity in the game (see Figure 2). For example, Bretons are 20 points more intelligent than Redguards and Nords of either gender; females Orcs and Argonians are 10 points more intelligent than males of the same race.

Examples of system-embedded biases (stereotypes of race and gender). While Harrell previously highlighted inequity and biases in *Oblivion*,[17] such insight is typically anecdotal and requires manual assessment. In order to quantitatively model these effects, we performed archetypal analysis on *Oblivion*'s distribution of statistical attributes for characters within the game based on gender and race, as in Figure 2. All races' relationships to these archetypes can be presented using a ternary plot of the results (see Figure 3). This analysis automatically revealed the typical "archetypal" roles developers intend for players to conform to based on how the statistical attributes are distributed within the game.[28] More interesting, however, the analysis also revealed several box effects within the game's design. Observing the three archetypes shows that male characters have better stats for playing in any of the most common roles than females. It also validates the observation that characters of African descent are optimized for strength- rather than intelligence-based roles. This quantitatively and visually represents biases that are often not as obvious, as in the following results:

Traditional game roles. Using archetypal analysis, we observed that the statistical distribution of numerical attributes corresponds with traditional role-playing game roles we call "physical-fighter," "intelligence-mage," and "stealth-thief";

Key individuals. Our system calculated particular races to be key individuals (archetypes). For example, the Viking-like Nords and ostensibly African Redguards are stereotypically close to the physical-fighter archetype with no characteristics of the intelligence-mage archetype, though the Redguards exhibit some stealth-thief characteristics; and

Physical-world stereotypes. We observed a bias toward the male gender based on these archetypal races since male characters are generally closer to the archetypes. The game is thus inequitable toward certain races and female characters in ways that replicate physical-world stereotypes.

Revealing such box effects computationally enables us to quantitatively assess virtual identity systems, providing actionable insights into how designers' decisions affect users and assistance in developing systems that enable users to take on virtual roles while avoiding undesirable biases. Our results do not suggest all characters should have equal attributes. Rather, they can inform creative designs that are just as effective and even more tied in to game narratives; for example, initial attributes could be based on characters' backstories, rather than on essential characteristics of races or genders, by having

Figure 5. Plot of how female players allocated statistical attributes of conventional role-playing games based on the gender of their avatar (female/male); note, the error bar for male avatars' intelligence is zero because all were assigned a value of 4 on a 7-point scale.

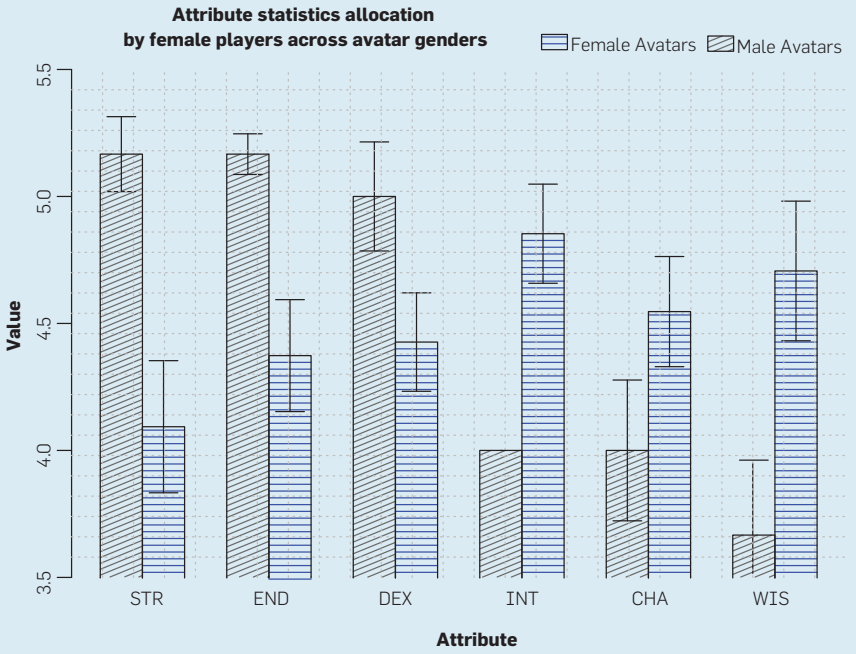
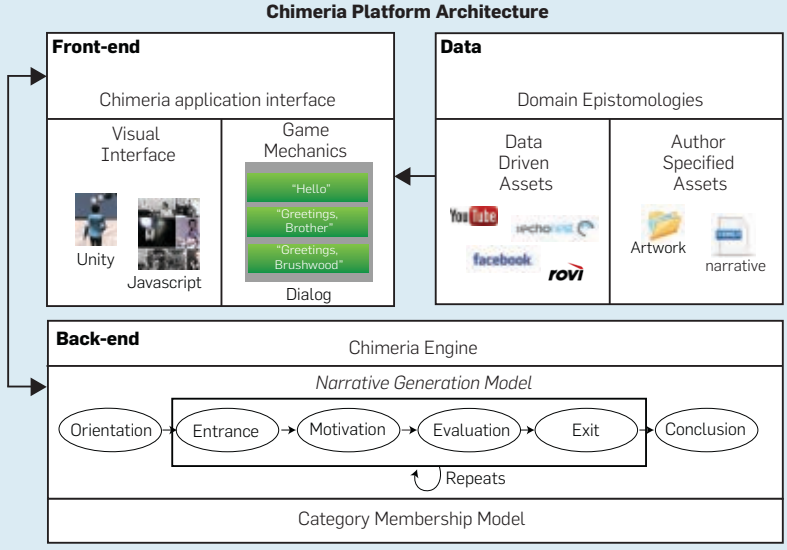


Figure 6. The *Chimeria* platform architecture.



players choose characters' prior life events from categories corresponding to the archetypes (such as studious, or mage, physically strenuous, or fighter, or street-smart, or thief, upbringings).

Analyzing users: Revealing user-enacted values. We have revealed user values and preferences in the following ways:

Modeling identity expression from player data. Using our *AIRvatar* system, we created a custom-developed avatar customization system called *Heroes of Elibca* (see Figure 4)[f] to give us full control over aspects of data collection for our experiment design. Players were presented with an introductory sequence to provide them with a familiar, computer-role-playing-game setting. Additionally, this helped us to contextualize the study as a scenario in which created avatars would be used as part of a videogame. Out of Harrell's taxonomy of technical components of computational identity systems[16]—static media assets, flat text profiles, modular graphical models, statistical/numerical representations, formal annotation, and procedural/behavioral rules—the results in the following subsections focus on statistical/numerical representations.[g] Players customized their avatars by modifying the values of six statistical attributes—strength, endurance, dexterity, intelligence, charisma, and wisdom—on a seven-point scale; see the online appendix for descriptions based on commonly used conventions in role-playing videogames. Each attribute had default values of 4; there were 27 allocatable points for each avatar.

Examples of user-enacted biases (stereotypes from gender-category expectations). While previous research has shown that players exhibit gender bias toward avatars controlled by others in a virtual environment,[39] here we present findings showing how biases can be modeled when constructing one's own avatar. We conducted a user study with 185 participants who were asked to customize their avatars using the avatar creator *Heroes of Elibca*. The demographic breakdown included 104 participants (or 56%) self-identified as male and 81 (or 44%) self-identified as female. We studied how they assigned statistical attributes to their avatars based on gender.

The underlying engine allows for the movement of individuals within, between, and across social categories.

The results reveal the phenomenon of gender stereotyping in how some of these avatars were customized. In Figure 5, observe that female players gave male avatars significantly higher values for physical-related traits (such as "strength," "endurance," and "dexterity") while giving them significantly lower values for intellect-related traits (such as "intelligence," "wisdom," and "charisma"). Female players here appear to be projecting aspects of an identity experience from the physical world (such as stereotypes of gender) onto the avatars. This demonstrates a kind of box effect, as it reflects a player's cognitive formation of categories of gender roles, along with associated assumptions and expectations. Interestingly, we did not observe these effects in avatars created by male users, as in the online appendix. We observed this asymmetry in earlier results of a smaller sample size[29] and also by other researchers.[21] Such results are not meant to portray female players negatively; factors like the genre of the game may reward traditionally "male" behaviors like physical aggressiveness.[35] These user-enacted social-identity phenomena reflect the situated nature of cognitively forming categories.

Simulating Social Identify Phenomena

The last section focused on techniques for identifying and analyzing box effects, but analysis alone is not enough for reimagining the Avatar Dream. Developers need tools that are better able to model sociocultural-identity categories and the experiences that people have based on them. Here, we provide an overview of our platform developers can use to design and implement virtual identity systems that help users better understand box effects and/or en-

f Art assets and resources from the publicly available Mack Looseleaf Avatar Creator (http://www.geocities.jp/kurororo4/looseleaf) and Liberated Pixel Cup (http://lpc.opengameart.org).

g We do not cover our experiments analyzing other components; for example, in analyzing text profiles and tags, we find avatars clustered into three categories of narrative theme and genre: personal stories (such as gender, age, and family relationships); functional stories (such as occupation, geographical location, and work); and fantastical themes (such as magic and power).[26] Image analysis reveals patterns of players conforming to gender stereotypes when creating avatars for themselves.[30]

able more nuanced identity-category models that avoid them. Modeling box effects is necessary for the first part of the dream—being whomever you want using a virtual identity—because being someone is not just a matter of graphical appearance, but of modeling systematic experiences. The second part of the Avatar Dream—understanding the experience of others—requires modeling social experiences more robustly to avoid box effects. While one may not be able to directly experience what it means to live a physical-world life as a member of another social category using virtual identity, it is possible to use virtual identities to convey some of the patterns of experience people in other categories face and that exist structurally in societies. Enabling users to be a virtual female superhero or even just a more suave and dandy self requires techniques to help them imagine the subjective experience of those types of identity. Our platform demonstrates representational benefits of a gradient model of social identity; our examples demonstrate applications that aim to engender critical awareness about the nuances of social identity.

Computational models of social identity are found in a wide range of digital-media works. In computer role-playing games, racial categorization is typically used to style the visual appearance of a player's avatar or trigger several different canned reactions when conversing with a non-player character. In social media, users typically join groups based on shared taste or categorize each other as "colleagues" or "family members" using privacy settings. In such systems, category membership is determined in a top-down fashion; members often slot into single, homogeneous groups with no hybrid identities, identities at the margins of groups, or identities that change over time. They neither provide developers and users elegant ways to avoid box effects nor simulate them to create via more expressive virtual worlds.

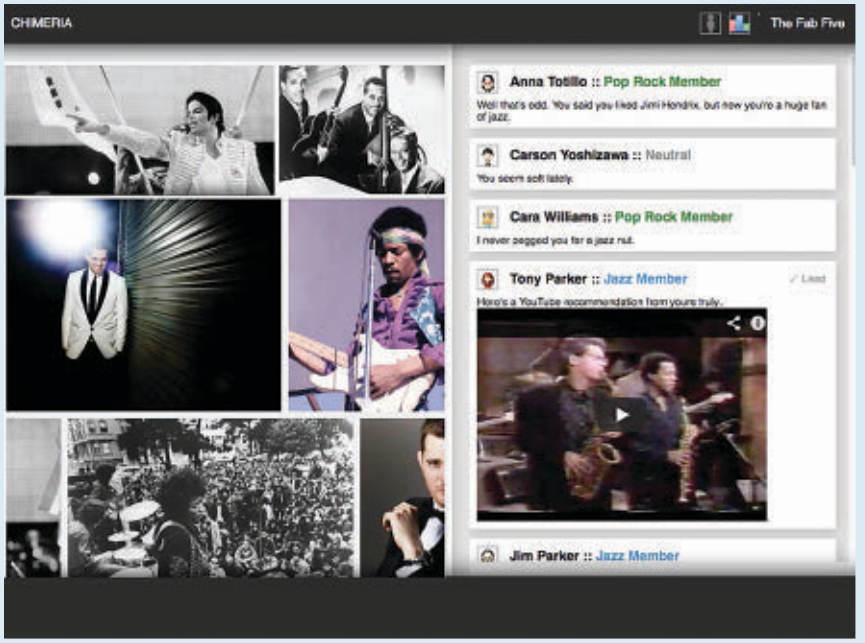
Simulating and avoiding box effects. Our "*Chimeria* platform" (hereafter *Chimeria*) is a system that supports simulating of physical world identity phenomena in virtual identity systems ranging from social-media accounts to videogames. Such simulation augments virtual identity models with gradience and dynamics, increasing their sociocultural nuance. Such additional nuance supports demonstrating how box effects are detrimental. And demonstrations are performed by creating expressive systems (such as videogames) that reveal how forms of discrimination function or avoiding box effects in utilitarian systems (such as social-media platforms). It does so in two primary ways: modeling the underlying structure of many social categorization phenomena with a computational engine; and enabling users to build their own creative applications about social categorization using the engine as a backbone. The underlying engine allows for the movement of individuals within, between, and across social categories.

It also allows for category members to have varying degrees of centrality to each group, assimilate or naturalize in relation to a hegemonic group, and be members of multiple groups. These aspects of the system are grounded in theories from sociolinguistics,[33] cognitive science,[23,h] and the sociology of classification.[3] The system is thus capable of modeling complex social behaviors (such as "impression management," addressed later). We next describe the architecture of *Chimeria* and two applications built with it.

***Chimeria* authoring platform.** *Chimeria* supports simulating experiences based on social-group membership using a data-driven approach and consists of three main components (see Figure 6). Simulations may take different forms (such as a 2D visual novel game, a fictitious social network chat narrative, or 3D virtual environment[i]).

Chimeria engine. This is our implementation of a mathematical model of users' degrees of membership across multiple categories. The *Chimeria* engine is designed to calculate, modify, and simulate changes to these memberships, acting as the system's logical processing component. It models users' category memberships as gradient values relative to the more central members,[3,17,23] enabling more representational nuance than binary status of member/nonmember commonly used in applications; for example, on the social network Facebook,

Figure 7. Screenshot from *Chimeria:MusicNet*.



h Lakoff cited a convergence of work in multiple fields, suggesting a need for more nuanced categorization models. For example, from computer science, he cited Zadeh's fuzzy logic,[40] which is useful for formalizing our models of gradient membership but unnecessary for the current implementation.

i We also made a demo integrating *Chimeria* with a 3D game interface, using the Unity game engine; see https://unity3d.com/

being someone's friend can be viewed as a basic Boolean flag; in the physical world, however, there are varying types and levels of friendship people have with others.[j] *Chimeria* is intended to enable both a greater range of expression of such nuances and representations that better serve users.

Chimeria application interface. This is a visual interface for user interaction and for experiencing the narratives of category membership changes, or game or story interfaces. The separation between the back-end (*Chimeria* engine) and the front-end (*Chimeria* application interface) provides the flexibility to go through the same narrative trajectory in relation to membership shifts with varying visuals. *Chimeria* narratives are authored by developers using an XML file format with a narrative structure, as described in Harrell et al.[19]

Chimeria domain epistemologies. An "epistemology" is an ontology[k] that describes cultural knowledge and beliefs.[18] In *Chimeria*, they are the knowledge representations of the categories being modeled. Assets used to present these categories can be author-contributed (such as graphics and text) or data-driven (such as retrieved YouTube videos).

***Chimeria* applications.** To better illustrate the capabilities of the components within *Chimeria*, we describe two very different simulations of social experience created using *Chimeria*. These are *Chimeria:MusicNet*, a social-networking simulation application that models social categories in the domain of musical preferences[15] and *Chimeria:Gatekeeper*, a computer role-playing-game scenario that models a conversational narrative between the player and a non-player character.

Chimeria:MusicNet. This application, the name of which is an abbreviation of *Chimeria:Musical-Identity-Social-Network*, uses the *Chimeria* engine to model social experiences based on categories of music preference. Psychologists David Hargreaves, Dorothy Miell, and Raymond MacDonald note that the music people listen to becomes a venue for the expression and formulation of their sense of self-identity and identity portrayed toward others, or "a musical identity."[15] The system models category membership using musical preferences that are automatically constructed from a user's set of music "likes," or binary indications of positive valuation, on a social-network profile. These "likes" are musical artists from which the *Chimeria* engine extrapolates (using commercially available musical-classification data) moods (such as cheerful and gloomy), themes (such as adventure and rebellion), and genres (such as film score). This extrapolation leads to a set of musical-identity categories, that is, musical-affinity groups that provide the context for non-binary group membership and passing, or the "ability of a person to be regarded as a member of social groups other than his or her own ... generally with the purpose of gaining social acceptance."[34] Each user's set of moods, themes, and genres then affect the generated narrative in fundamental ways. The focus for *Chimeria:MusicNet* is not on categorizing music but on the modeling of musical preferences using a knowledgebase aggregated from external data. These models are used to dynamically construct a narrative conveyed through a social network interface, or "conversational narratives," structured by a model of conversation from sociolinguistics.[33]

Figure 7 is a screenshot of *Chimeria:MusicNet*. A dynamic collage of photos, or photowall, is procedurally generated to represent the user's musical-taste preferences; a feed of recent updates, posts, and invitations appears in an adjacent vertical timeline, as in Figure 7. Using musical preferences from the user's Facebook music likes or by manual entry, a hybrid real/fictitious conversational narrative experience progresses over time in a manner described as follows. Dynamically generated posts by the user's non-player character friends comment on the user's membership in multiple musical-affinity groups, as in "You're a raucous rock fan now?" or "Want to hear some airy jazz music?" The user may

j For example, using features related to tie strength,[12] the *Chimeria* engine could be used to dynamically compute friend centrality.

k Unlike its use in philosophy regarding the nature of existence, we use the term "ontology" here in its computer-science sense, referring to the AI notion of computational-knowledge representations. Ontologies provide ways to specify conceptualizations of aspects of some domain (such as using objects, types of objects, attributes, relations, and events) and can be represented informally, semi-formally (such as markup languages), or very formally (such as first-order logic).

Figure 8. Screenshots from *Chimeria:Gatekeeper*.

"like," "dislike," or simply ignore these posts, resulting in group-membership changes. Some friends question newly discovered interests while others pass judgment on prior affiliations. The resulting narrative may describe passing or assimilating as a member of a new group of music listeners, reinforcing a prior group affiliation, or even being marginalized in every group.

Chimeria:Gatekeeper. This application models a common role-playing-game scenario—a player trying to gain access to the inside of a castle. The scenario illustrates a phenomenon noted by Harrell,[16] who wrote, "There exists a perceived appropriateness of particular ways to present one's self in different situations, as well as social avenues that may be closed off or accessed only with more difficulty due to externally defined social prejudices and biases. This perceived negative difference between diverse individuals and socially defined, desirable and privileged norms is called stigma." The *Chimeria:Gatekeeper* scenario is based on sociologist Erving Goffman's work on stigma.[13] The unseen player character is initialized in a "discredited" (stigmatized) category, and the non-player character is initialized in an "accepted" category. The discredited category is prototypically defined as the Sylvanns race—tall, well-spoken, and wearers of fine clothing. The accepted category is prototypically defined as the Brushwoods race—short, plain-spoken, and wearers of rough-spun clothing. To gain access into the castle, the player must exhibit behaviors that convince the guard that she or he should be admitted; most players try to demonstrate that the player character fits in the accepted category, a social-identity phenomena known as "passing."[13] Figure 8 depicts choosing a dialog option to fit into the accepted category. Actions (such as slouching to adopt the posture of a prototypical Brushwood or displaying fine Sylvann clothing) shift the non-player character's model of the player character's category memberships, rendering the outcome closer to gaining access or being rejected. *Chimeria* handles alternatives to the common strategy of intentionally passing, simulating experiences of a variety of box effects based on Goffman's notion of impression management.[13] Other simulated experiences include voluntary disclosure of stigma and slipping, or trying to pass as an accepted member but failing. They capture trade-offs between gaining utilitarian access versus loss of self-identity.

User testing has revealed *Chimeria* overcomes limitations common in virtual identity systems while enabling critical examination of how identities are negotiated in the physical world.[20,l] While this fantasy scenario may seem far removed from physical-world experiences of stigma like sexism and racism on the job, such tensions exist and are common; for example, in the U.S, speakers of southern dialects of English have described needing to change their speech patterns to suitably impress an employer, and female entrepreneurs and politicians have described pressure to de-emphasize stereotypically feminine characteristics to be taken more seriously by those in positions of power. Such people have described having to get past "gatekeepers" as an apt metaphor for their experience.

Future Work

The outcomes of the work we have described here have led to several new projects at the intersection of computing, sociocultural identity, and imaginative cognition. The projects further model dynamic relationships between virtual identities and sociocultural identity phenomena in the physical world. These projects aim to, respectively, use avatars to support public high school students from groups currently underrepresented in STEM fields in seeing themselves as powerful learners and doers of computer science and to excite them about the field; better understand global culturally specific everyday uses of virtual identities in social media and videogames; and create a virtual reality system that helps engender empathy in the midst of global conflict (a collaborative project directed by war photojournalist Karim Ben Khelifa).[m]

Technologies we use to imagine ourselves can be powerful media for social empowerment through critical thought and social awareness. For us, this is a more urgent dream. Like dreams of ubiquitous computing and AI, the most important aspect of the Avatar Dream is not whether it is achievable but that it pushes us to consider the limits and ethics of virtual identity technology development and propels us toward innovations that benefit society.

Conclusion

This article is a result of more than seven years of research toward our reimagined Avatar Dream wherein addressing social and cultural concerns is intrinsic to its realization. The Avatar Dream is not a panacea for social-identity problems; virtual identities are mere technical components of broader phenomena of human identities and the many concepts, artifacts, and interactions that produce them. We must move beyond questions of whether the Avatar Dream is achievable and also consider whether it would be good if achieved.

Still, we have thoughts regarding whether the Avatar Dream is indeed achievable. Answering first necessitates clarifying what it means to become someone or something else using a computer. Humans have great power of self-imagination. Yet the physical world we live in is rife with individual, social, and cultural histories that affect people's capacities to determine their own identities. Such histories constrain our ability to directly understand the experience of others. As human-created artifacts, virtual identities reflect historical, social, and cultural constraints from the physical world. Achieving the Avatar Dream requires a better understanding of the relationships between the constraints imposed by our social-identity experience in the physical world and our potential for self-imagination in virtual worlds. Ignoring these constraints on our social identities results in both system-embedded and user-enacted box effects, rendering the Avatar Dream unachievable. While the existence of negative box effects has been forcefully argued in anecdotal terms, we have demonstrated a method for em-

l For example, a majority of players perceived that identity was central to the interaction in the game scenario and that it affected conversation more than in other games in the genre.

m See https://www.nytimes.com/2016/10/30/arts/design/meeting-the-enemy-face-to-face-through-virtual-reality.html

pirically demonstrating their existence through computational modeling. If virtual identities are used to reinforce cognitive or structural constraints to the detriment of individuals, achieving the Avatar Dream would be harmful, even if possible.

A child growing up in poverty imagining herself as a future successful engineer—despite having never lived as one—is a powerful act of self-imagination. If she is discriminated against because she is deemed poor (or any other identity-related reason) and denied access to the resources to become an engineer, then structural constraints have limited her ability to take on a social identity she aspires to. If she believes that becoming an engineer is not achievable because of her socioeconomic status, then cognitive constraints based on her experience of social identity have limited her capacity to self-imagine. Our work using AI to analyze blended identities aims to reveal both structural constraints embedded in systems and cognitive constraints emerging from users. We seek to support individuals' capacities to self-imagine in empowering ways while negotiating oppressive social constraints they face. At times, this may entail supporting users to imagine themselves as whomever they want to be; at other times, it entails supporting users in realizing and negotiating constraints rooted in the physical world. This is our reimagined Avatar Dream—a socially and culturally informed vision that *would* be good if achieved.

Acknowledgments

This material is based on work supported by the National Science Foundation Grant #1064495 and extended under NSF Grant #1542970 and a QCRI-CSAIL Collaboration. We thank the anonymous reviewers, as well as Dominic Kao and Pablo Ortiz, for their helpful feedback.

References

1. Bates, J. Virtual reality, art, and entertainment. *Presence: Teleoperators and Virtual Environments 1*, 1 (Jan. 1992), 133–138.
2. Bauckhage, C., Kersting, K., Sifa, R., Thurau, C., Drachen, A., and Canossa, A. How players lose interest in playing a game: An empirical study based on distributions of total playing times. In *Proceedings of the IEEE Conference on Computational Intelligence and Games* (Granada, Spain, Sept. 11–14). IEEE, 2012, 139–146.
3. Bowker, G.C. and Star, S.L. *Sorting Things Out: Classification and its Consequences*. MIT Press, Cambridge, MA, 1999.
4. Chow, K.K.N. and Harrell, D.F. *Elastic Anchors for Imaginative Conceptual Blends: A Framework for Analyzing Animated Computer Interfaces*. CSLI Publications, Stanford, CA, 2013.
5. Coulson, S. and Oakley, T. Blending basics. *Cognitive Linguistics 11*, 3-4 (2006), 175–196.
6. Dibbell, J. A rape in cyberspace; or how an evil clown, a Haitian trickster spirit, two wizards, and a cast of dozens turned a database into a society. Chapter in *High Noon on the Electronic Frontier: Conceptual Issues in Cyberspace*, P. Ludlow, Ed. MIT Press, Cambridge, MA, 1996, 375.
7. Duggan, M. and Smith, A. *Social Media Update 2013*. Pew Internet and American Life Project, Washington, D.C., 2013; http://www.pewinternet.org/2013/12/30/social-media-update-2013/
8. Everett, A. and Watkins, S.C. The power of play portrayal and performance of race in video games. Chapter in *The Ecology of Games: Connecting Youth, Games, and Learning*, K. Salen Tekinbaş, Ed. The John D. and Catherine T. MacArthur Foundation Series on Digital Media and Learning. MIT Press, Cambridge, MA, 2008, 141–166.
9. Fauconnier, G. and Turner, M. *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities*. Basic Books, New York, 2008.
10. Gee, J.P. *What Video Games Have to Teach Us about Learning and Literacy*. Palgrave Macmillan, New York, 2007.
11. Gibbs, R.W. Jr. Making good psychology out of blending theory. *Cognitive Linguistics 11*, 3–4 (2001), 347–358.
12. Gilbert, E. and Karahalios, K. Predicting tie strength with social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, Apr. 4–9). ACM Press, New York, 2009, 211–220.
13. Goffman, E. *Stigma: Notes on the Management of Spoiled Identity*. Touchstone, New York, 1963.
14. Hafner, K. The epic saga of The Well. *Wired* (May 1, 1997); https://www.wired.com/1997/05/well/
15. Hargreaves, D.J., Miell, D., and MacDonald, R.A., Eds. What are musical identities, and why are they important? Chapter in *Musical Identities*. Oxford University Press, Oxford, U.K., 2002, 1–20.
16. Harrell, D.F. Computational and cognitive infrastructures of stigma: Empowering identity in social computing and gaming. In *Proceedings of the Seventh ACM Conference on Cognition and Creativity* (Berkeley, CA, Oct. 27–30). ACM Press, New York, 2009, 49–58.
17. Harrell, D.F. Toward a theory of critical computing. *CTheory* (May 13, 2010); http://ctheory.net/articles.aspx?id=641
18. Harrell, D.F. *Phantasmal Media: An Approach to Imagination, Computation, and Expression*. MIT Press, Cambridge, MA, 2013.
19. Harrell, D.F., Kao, D., and Lim, C.-U. Computationally modeling narratives of social group membership with the Chimeria system. In *Proceedings of the 2013 Workshop on Computational Models of Narrative, Volume 32 of OpenAccess Series in Informatics*, M.A. Finlayson, B. Fisseni, B. Löwe, and J.C. Meister, Eds. (Hamburg, Germany, July 31–Aug. 3). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2013, 123–128.
20. Harrell, D.F., Kao, D., Lim, C.-U., Lipshin, J., Sutherland, A., Makivic, J., and Olson, D. Authoring conversational narratives in games with the *Chimeria* platform. In *Proceedings of the Ninth Annual Conference on Foundations of Digital Games* (Ft. Lauderdale, FL, Apr. 3–7). Society for the Advancement of the Science of Digital Games, Copenhagen, Denmark, 2014.
21. Huh, S. and Williams, D. Dude looks like a lady: Gender swapping in an online game. Chapter in Online Worlds: *Convergence of the Real and the Virtual*. Springer, London, U.K., 2010, 161–174.
22. Hutchins, E. Material anchors for conceptual blends. *Journal of Pragmatics 37*, 10 (Oct. 2005), 1555–1577.
23. Lakoff, G. *Women, Fire, and Dangerous Things*. University of Chicago Press, Chicago, IL, 1987.
24. Laurel, B. and Strickland, R. Placeholder: Landscape and narrative in virtual environments. In *Proceedings of the Second ACM International Conference on Multimedia* (San Francisco, CA, Oct. 15–20). ACM Press, New York, 1994, 121–127.
25. Lave, J. and Wenger, E. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press, Cambridge, U.K., 1991.
26. Lim, C.-U. and Harrell, D.F. Developing computational models of players' identities and values from videogame avatars. In *Proceedings of the 10th International Conference on the Foundations of Digital Games* (Pacific Grove, CA, June 22–25). Society for the Advancement of the Science of Digital Games, Copenhagen, Denmark, 2015.
27. Lim, C.-U. and Harrell, D.F. The marginal: A game for modeling players' perceptions of gradient membership in avatar categories. In *Proceedings of the Second AIIDE Workshop on Experimental AI and Games* (Santa Cruz, CA, Nov. 14–18). AAAI Press, Palo Alto, CA, 2015, 11, 49–55.
28. Lim, C.-U. and Harrell, D.F. Revealing social identity phenomena in videogames with archetypal analysis. In *Proceedings of the Sixth International AISB Symposium on AI and Games* (Kent, U.K., Apr. 20–22). Society for the Study of Artificial Intelligence and Simulation of Behaviour, London, U.K., 2015.
29. Lim, C.-U. and Harrell, D.F. Toward telemetry-driven analytics for understanding players and their avatars in videogames. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (Seoul, Republic of Korea, Apr. 18–23). ACM Press, New York, 2015, 1175–1180.
30. Lim, C.-U., Liapis, A., and Harrell, D.F. Discovering social and aesthetic categories of avatars: An artificial intelligence approach using image clustering. In *Proceedings of the First Joint International Conference of DiGRA and FDG* (Dundee, U.K., Aug. 1–6). Digital Games Research Association, Tampere, Finland, 2016; http://www.digra.org/wp-content/uploads/digital-library/paper_1971.pdf
31. Mateas, M. *Interactive Drama, Art and Artificial Intelligence*. Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, 2002; http://www.cs.cmu.edu/~dod/papers/CMU-CS-02-206.pdf
32. Perlin, K. and Goldberg, A. Improv: A system for scripting interactive actors in virtual worlds. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*. (New Orleans, LA, Aug. 4–9). ACM Press, New York, 1996, 205–216.
33. Polanyi, L. *Telling the American Story: A Structural and Cultural Analysis of Conversational Storytelling*. MIT Press, Cambridge, MA, 1989.
34. Renfrow, D.G. A cartography of passing in everyday life. *Symbolic Interaction 27*, 4 (Nov. 2004), 485–506.
35. Schrier, K. Avatar gender and ethical choices in *Fable III*. *Bulletin of Science, Technology & Society 32*, 5 (Oct. 2012), 375–383.
36. Stephenson, N. *Snow Crash*. Spectra, New York, 2003.
37. Turkle, S. *The Second Self: The Human Spirit in a Computer Culture*. Simon & Schuster, New York, 1984.
38. Varela, F.J., Thompson, E., and Rosch, E. *The Embodied Mind. Cognitive Science and Human Experience*. MIT Press, Cambridge, MA, 1991.
39. Yee, N., Ducheneaut, N., Nelson, L., and Likarish, P. Introverted elves and conscientious gnomes: The expression of personality in *World of Warcraft*. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada, May 7–12). ACM Press, New York, 2011, 753–762.
40. Zadeh, L.A. Fuzzy sets. *Information and Control 8*, 3 (June 1965), 338–353.

D. Fox Harrell (fox@csail.mit.edu) is Professor of Digital Media in both the Comparative Media Studies Program and the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology, Cambridge MA, and the founder and director of the Imagination, Computation, and Expression Laboratory.

Chong-U Lim (culim@csail.mit.edu) recently completed his Ph.D. in electrical engineering and computer science from the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology, Cambridge MA, where he was a member of the Imagination, Computation, and Expression Laboratory.

Watch the authors discuss their work in this exclusive *Communications* video. https://cacm.acm.org/videos/reimagining-the-avatar-dream

DOI:10.1145/3019940

Information and communication technology patents are more influential on subsequent inventions than are other types of patents.

BY PANTELIS KOUTROUMPIS, AIJA LEIPONEN, AND LLEWELLYN D W THOMAS

How Important Is IT?

THROUGHOUT THE 20TH century, particularly after 1970, the technological revolution in semiconductors, digital communications, and consumer electronics resulted in dramatic changes to our everyday lives. We work, travel, consume, and communicate differently thanks to technological innovation. As a consequence, the role of information and communications technologies (ICTs) in the global economy is often the center of popular and academic attention—to the point that the word "technology" in common parlance has come to mean "information technology." But just how influential have ICTs become in affecting subsequent inventions compared to other technologies? This article examines the special role of ICTs in influencing technological development over the 20th century. The analysis is intended to highlight the role of ICTs in societal accumulation of technology, not only in terms of production and consumption.

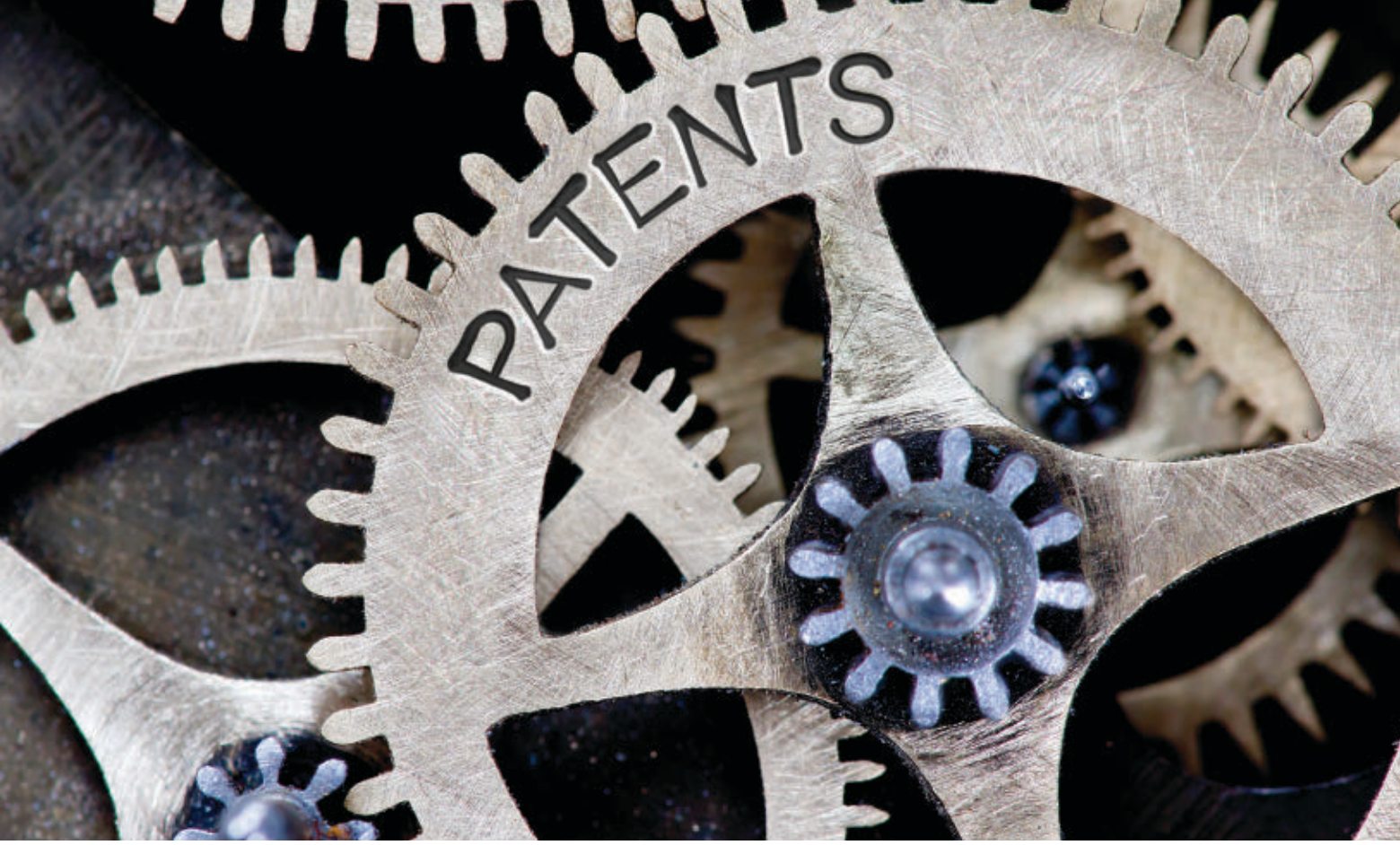
ICTs can be utilized as components and enablers in many different production activities. For this reason, they are thus seen as drivers of economic growth. At the national-economy level, ICTs have a strong influence on economic growth,[8,14] and although there was some early debate as to whether ICT improves productivity,[a] it is now accepted among economists that at the firm level, ICTs significantly contribute to industrial productivity.[4,7] Academic economists generally agree that such productivity benefits first appear in the ICT-producing industries and gradually spread to ICT-using industrial sectors.[19]

Economic history suggests that innovation over long economic and technological cycles is brought about by general-purpose technologies,[6] or those that can be adopted in diverse economic sectors and lead to efficiency and improvements. For instance, electrical engines can power cars, factory machines, or home appliances. Electricity is thus a general-purpose technology that has been adopted by the car manufacturing industry, along with other machine industries, leading to improvements and efficiencies to the car as a product. However, the economywide contribution of such general-purpose technologies on subsequent invention has not been thoroughly examined thus far, with studies focusing instead on specific technologies (such as broadband).[2,3] There are also several studies that link adoption of ICTs with subsequent invention activities,[1,10] while others have considered the

a "You can see the computer age everywhere but in the productivity statistics."[17]

» key insights

- **ICT patents are consistently more central to invention, and much more central since 1970, than patents from other industrial sectors.**
- **ICT patents receive up to 0.406 more citations and a considerably higher PageRank than non-ICT patents.**
- **PageRank offers a quality-adjusted indicator that helps measure the true influence of inventions.**



processes of "co-invention" associated with information technologies.[5] Unlike these earlier studies, in this article we empirically assess whether ICTs have had a greater influence on cumulative technological change compared to other technological fields of the 20th century.

We undertake our investigation using patent data, in particular the citations between individual patents. Patent data, with its global reach and wide technological spectrum, is an ideal testbed for describing how (protected) knowledge from ICT sectors has influenced other non-ICT sectors. Patent citations are references to "prior art" that the focal invention builds on or relates to and thus limit the originality of the focal invention.[11] They are widely used as a proxy for knowledge spillovers indicating the transfer of innovative applications and ideas within and across fields of technology.[18] Consequently, citation patterns have been used extensively to inform industry and technology policy making; see, for instance, Dechezlepêtre et al.[9] and Jaffe and Trajtenberg.[12] Citations to a patent have also been used as a measure of its importance in industry comparisons, intellectual property management, and valuation of firms.[11]

We empirically compare knowledge spillover from ICT and non-ICT inventions at the patent-application level. Utilizing the PatStat database, which consists of the entirety of patent records for more than 160 patent offices over the past 100 years, we use two methods. We first analyze the number of prior art (forward) citations per patent, controlling for the type of the underlying invention (ICT or non-ICT) to assess the differences between the two groups. Second, we compute the importance of each patent based on the Google PageRank algorithm and substitute the number of prior-art citations with the PageRank metric; based on this model we rerun our analysis to assess the difference between ICT and non-ICT patents. For both approaches we apply a wide range of patent-specific, fixed, and time-varying controls.

We confirm that ICT patents are more influential than other types of patents, observing a significant difference in the citations of ICT and non-ICT technologies. ICT patents receive up to 0.406 more citations and a considerably higher PageRank than non-ICT patents. These findings quantify the influence of ICT inventions on other technological inventions. Moreover, the PageRank method provides a quality-adjusted indicator that helps measure the true influence of inventions. We suggest the exceptional influence of ICTs is due to their openness and flexibility enabling complementary invention and the fundamental roles of information and communication in the very process of invention.

Method

Our data source is PatStat, a comprehensive dataset available from the European Patent Office, with data from more than 160 publication authorities, 90 million awarded patents, and 160 million citations for the period 1900 to 2014. This information is also linked to detailed data about patent-level publication claims, patent families, technology fields, and classification data. Table 1 details the PatStat organization of technology sectors and fields. We use the patent universe (all listed patents from the PatStat dataset) to identify the complete network of prior-art citations. Table 1 also details our identification of ICT patents.[b] As ICT is not

b Patent-classification schemes are occasionally modified, and a patent can change its specific classification. However, past reclassifications do not influence our analysis, as most reclassifications happen at quite granular levels. Our analysis is at the rather coarse sector level; that is, it is unlikely for a patent to be reclassified between technology sectors.

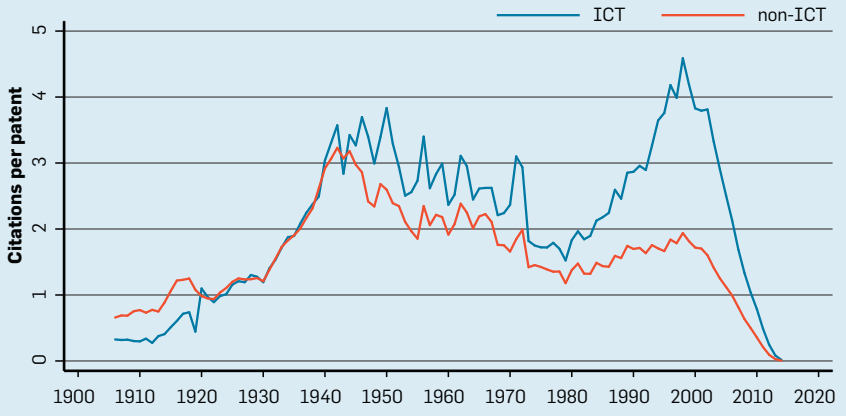
IMAGE BY ETTAMMOS

Table 1. Technology sectors, technology fields, and ICT relevance.

Technology Sector	Technology Field	ICT Status
Chemistry	Basic materials chemistry	non-ICT
Chemistry	Biotechnology	non-ICT
Chemistry	Chemical engineering	non-ICT
Chemistry	Environmental technology	non-ICT
Chemistry	Food chemistry	non-ICT
Chemistry	Macromolecular chemistry, polymers	non-ICT
Chemistry	Materials, metallurgy	non-ICT
Chemistry	Microstructural and nanotechnology	non-ICT
Chemistry	Organic fine chemistry	non-ICT
Chemistry	Pharmaceuticals	non-ICT
Chemistry	Surface technology, coating	non-ICT
Electrical Engineering	Audio-visual technology	ICT
Electrical Engineering	Basic communication processes	ICT
Electrical Engineering	Computer technology	ICT
Electrical Engineering	Digital communication	ICT
Electrical Engineering	Electrical machinery, apparatus, energy	ICT
Electrical Engineering	IT methods for management	ICT
Electrical Engineering	Semiconductors	ICT
Electrical Engineering	Telecommunications	ICT
Instruments	Analysis of biological materials	non-ICT
Instruments	Control	ICT
Instruments	Measurement	non-ICT
Instruments	Medical technology	non-ICT
Instruments	Optics	non-ICT
Mechanical Engineering	Engines, pumps, turbines	non-ICT
Mechanical Engineering	Handling	non-ICT
Mechanical Engineering	Machine tools	non-ICT
Mechanical Engineering	Mechanical elements	non-ICT
Mechanical Engineering	Other special machines	non-ICT
Mechanical Engineering	Textile and paper machines	non-ICT
Mechanical Engineering	Thermal processes and apparatus	non-ICT
Mechanical Engineering	Transport	non-ICT
Other Fields	Civil engineering	non-ICT
Other Fields	Furniture, games	non-ICT
Other Fields	Other consumer goods	non-ICT

Note: Technology fields are non-overlapping IPC codes available from the PatStat dataset.

Figure 1. Yearly citations per patent for ICT and non-ICT sectors.



solely the domain of the electrical-engineering sector, we code each technology class as ICT or non-ICT, following Kim and Hwang.[13c]

We estimate a simple model at the patent level where the total number of forward citations it has received (a count) is regressed against the type of patent (ICT or non-ICT) and other controls X_{it}. These controls include patent office, year of grant, patent family, extended patent family,[d] stock of published patents by sector and year, count of citations by year and sector, and number of citations added by examiners. The controls permit us to alleviate sector-specific concerns (such as the potential ease in categorizing and finding during patent search), which lead to more complete referencing by patent officers, as well as rapid growth of the ICT sector itself, which leads to yet more patents and hence citations. Moreover, they enable us to indirectly control for regional and temporal effects regarding the "ease" of publishing a patent.[e] Self-citations by the same assignee(s) are excluded from the counts. The model is

$$C_i=\beta_i ICT_i+\gamma_i X_{it}+\varepsilon_i$$

where C_i is the count of all citations received by patent *i*, ICT_i is a dummy equal to 1 for ICT patents and 0 otherwise, and X_{it} is a vector of patent characteristics.

To reduce the potential bias of other potentially confounding effects in patent citations we also look at the data as a directed graph that evolves over time. This approach resembles the PageRank algorithm initially proposed by Google as a measure of ranking

c A full matching of sector to International Patent Classification classes is available from author Koutroumpis.

d We follow the European Patent Office, defining a patent family as "all documents having exactly the same priority or combination of priorities belong to one patent family" and a "broad" patent family as including all documents directly or indirectly linked to one specific priority document. Source: European Patent Office, https://www.epo.org/searching-for-patents/helpful-resources/first-time-here/patent-families/definitions.html.

e This control does not alleviate potential concerns that it is "easier" to patent ICT than other technologies; this is a topic for future research, and we thank an anonymous reviewer for suggesting it.

webpage influence.[16] Each patent represents a node in the graph, and each citation represents an edge that originates from the citing patent and ends at the cited patent. Ranking patents in terms of their influence instead of the count of incoming citations (in-degree) requires an iterative process that goes through each node of the graph and determines its rank as the sum of the ranks of other patents citing the focal patent. We compute the PageRank for each patent, then use the same model from the first regression, replacing the dependent variable with PageRank. As the PageRank method requires, we apply a damping factor to reduce the importance of nodes that are farther away.[f] As a further robustness test we compute the PageRank for the entire network by decade and observe the changing influence of ICT patents.

As the size of the dataset exceeds common computing capacities, the bulk of this analysis has taken place using `c4.8xlarge` compute-optimized instances and `r3.8xlarge` memory-optimized instances on the Amazon Cloud.

Results

We first plot the mean citations for ICT and non-ICT sectors over the period of study in Figure 1. ICT patents clearly display higher citation counts for most of the 20th century. We observe a peak in the 1980s and 1990s when the sector difference increased dramatically. Citation counts drop dramatically after 2005; we attribute it to both a smaller window of observations reducing the citation records and right censorship.[g] We thus consider results later than 2005 to be unreliable.

We also conducted a simple t-test on the citation counts to see whether a difference exists between ICT and non-ICT patent citations (see Table 2). The t-test clearly demonstrated the substantial differences between the technology sectors; ICT patents receive 0.842 more citations than other patents. This result can be attributed to a number of factors: ICT fields may produce more patents than others, increasing their citation counts; ICT patents might cite themselves or other ICT patents more frequently, further increasing the counts; and ICT patents might be associated with more developed international presence through larger patent families that might boost citation counts as well.

To account for these confounding factors, we include a number of control variables in our models in Table 3. In model 1, considering the total number of citations received by each patent, we observe that ICT patents now receive only 0.406 more citations than non-ICT; that is, ICT patents receive 27.6% more citations than other technologies.[h] The controls help explain almost half the effect found earlier in the simple t-statistic, which is both expected and reassuring about our method. In model 2 we limit the period of analysis to five years following publication, measuring the immediate impact of each invention rather than the longer-term effects. ICT patents attract 0.31 more citations than other technologies in their first five years, a finding that suggests that approximately three quarters of citation effects in ICT patents have already materialized in just five years following publication. As a

f We applied a number of damping factors that did not change the result.

g This drop is unrelated to the fact that ICT technologies are relatively newer compared to non-ICT technologies, as there is the same drop for all patent classes; see Koutroumpis et al.[15] for details.

h The estimated citation count coefficient is 0.406, or 27.6% of the mean citation count of 1.473.

Table 2. T-test of the sum of citations per patent.

Group	Observations	Mean	Standard Error
Non-ICT	63,318,494	1.221**	0.001
ICT	27,159,889	2.063**	0.002
Difference		-0.842**	0.002

Notes: Statistical significance at * $p<0.05$; ** $p<0.01$.

Table 3. Influence of ICT on patent citations and PageRank.

	(1) All citations	(2) Five-year citations	(3) PageRank
ICT	0.406**	0.310**	2.903**
	(257.04)	(213.49)	(301.34)
Patent stock	0.001**	0.001**	0.001**
	(104.23)	(102.95)	(5.40)
Family size	0.001**	0.001**	-0.003**
	(4.65)	(7.44)	(6.02)
Family size broad	0.001**	0.001**	-0.001**
	(72.94)	(56.41)	(13.14)
Patent claims	0.25**	0.214**	0.482**
	(1,667.87)	(1,473.02)	(528.34)
Citations added by examiner	2.362**	1.726**	10.319**
	(1,637.21)	(1,277.24)	(1,172.89)
Citation stock	0.001**	0.001**	0.001**
	(139.14)	(109.97)	(98.70)
Year fixed effects	Yes	Yes	Yes
Publication-authority fixed effects	Yes	Yes	Yes
Patent family clustering	Yes	Yes	Yes
R^2	0.18	0.16	0.11
N	88,725,978	82,073,428	88,725,978

Notes: Different dependent variables are used across specifications: (1) the count of citations; (2) the count of citations in a five-year window after publication; and (3) the PageRank of each patent multiplied by 10^9. A 0.85 damping factor is used for (3). All models include year, publication-authority fixed effects, and clustering at the patent-family level. Patent family is defined as "all documents having exactly the same priority or combination of priorities belong to one patent family" (DOCDB). A "broad" (INPADOC) patent family retrieves all documents directly or indirectly linked to one specific priority document.

Absolute value of t statistics in parentheses and statistical significance at * $p<0.05$; ** $p<0.01$.

further check, we find that the controls have the expected signs and significance, with the stock of patents positively affecting total counts and more popular sectors experiencing higher citation counts.

We now perform the computationally demanding iterative process of measuring the PageRank of each patent in our 90-million-patent dataset. This involves loading the edge list (approximately 160 million) of all citations and looping over citation networks until the algorithm converges under the specified accuracy thresholds. The PageRank offers a more informative metric compared to simple citation counts. Using the PageRank approach, two patents with identical citation counts are not necessarily equal; the importance of the citing patents instead reflects the importance of the focal invention.

Figure 2 shows the average PageRank by decade for selected technology sectors.[i] The effect we see with ICT patents is clearly reflected in the electrical-engineering sector where there was a distinct increase in importance from the 1970s onward. Moreover, these results seem to suggest the effect is ongoing. Emphasizing the continued effect of electrical engineering, mechanical engineering appears to be decreasing in importance, while chemistry and instruments appear to have remained at a constant (or slightly decreasing) level of importance during the period of study. The other-sectors category, including consumer goods and furniture and games, seem to be gradually increasing in importance, albeit from a more modest starting point.

Table 3 (model 3) lists the regression results of the PageRank analysis against all controls. We find ICT patents received 10% higher PageRanks than other technologies.[j] Although this appears to be a smaller difference than that of simple citations (27.6%), we highlight that the distribution of citation counts and PageRanks differ significantly (see Table 4).

In particular, the distribution of PageRanks is less dispersed than the simple counts, and its coefficient of variation (the standard deviation divided by the mean) highlights this phenomenon, as in Table 4. Simple counts have a coefficient of variation equal to 4.66, while PageRanks had a coefficient of variation of 2.12, or less than half the dispersion of the citation distribution; that is, a 1% increase in PageRank is equivalent to a 2.2% increase in citation count.[k] The 10% increase in PageRank ICT patents receive was thus equivalent to a 22% increase in citation counts, only slightly less than the 27.6% we found, confirming the effect.

Figure 2. Yearly average PageRank by technology sector.

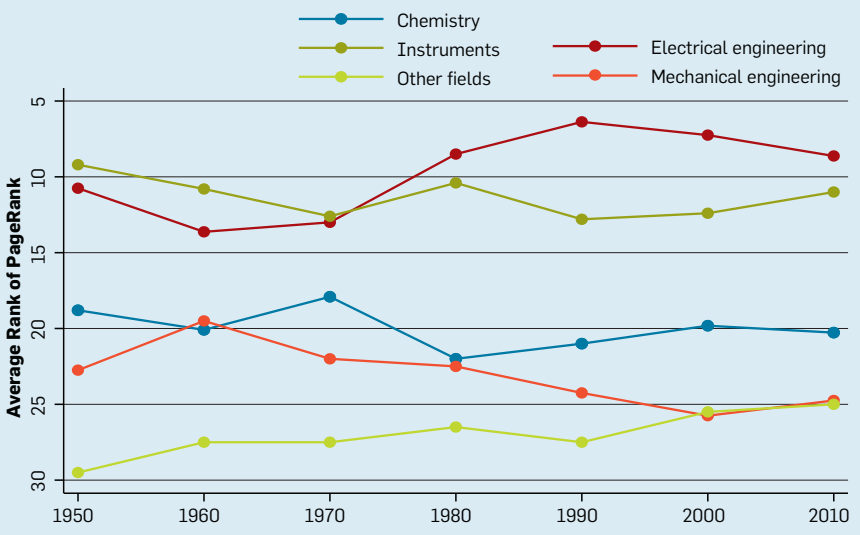
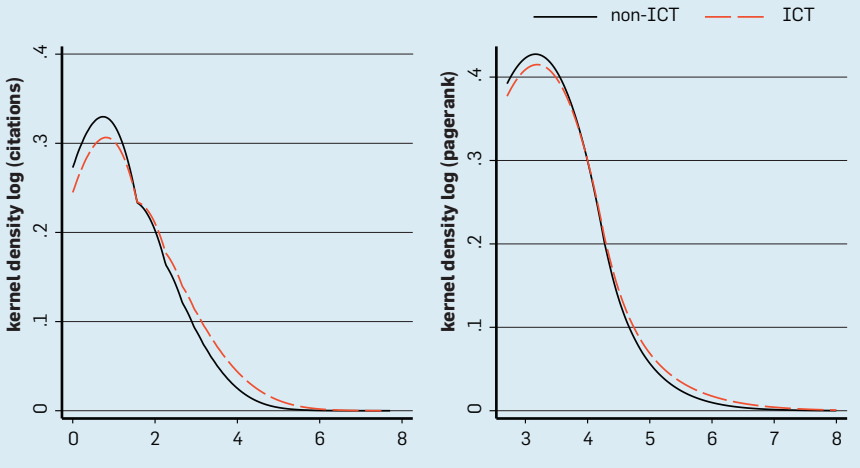


Table 4. Means, standard deviations, and coefficients of variation.

	Mean	Standard Deviation	Coefficient of Variation
Citations (all publications)	1.473	6.873	4.664
PageRank (x 10^9)	28.94	61.29	2.118

Notes: Summary statistics for citations (across publications) and PageRank variables. Mean and standard deviation are reported in the first two columns. The coefficient of variation, a measure of the dispersion, defined as the ratio of the standard deviation to the mean, is reported in the last column.

Figure 3. Kernel densities for (left) distributions of citations and (right) PageRank.



i Although our ICT/non-ICT comparison uses an extended definition of ICT, as in Table 2, we restrict the analysis here to common sectors for clarity.

j The estimated PageRank coefficient is 2.903, or 10.0% of the mean PageRank of 28.939.

k 4.664 divided by 2.118 is 2.202.

Figure 3 and Figure 4 plot the kernel densities for the logged distributions of citations and PageRanks, with the y-axis indicating the probability of each value in the x-axis. For easier presentation and comparison, and, as these scores are positive and non-zero, we use the logarithmic form. Probability distributions help us further understand the difference between the ICT and non-ICT results by presenting the probability of a particular citation count or PageRank value. Figure 3 shows the ICT effects remain strong for both citations and PageRanks, with the gap between ICT and non-ICT widening as we move toward the most influential patents, or those with greater citation counts or PageRanks. Although the difference between PageRank and log(citations) as a measure of influence appears to be relatively minor at the very high end of the distribution, Figure 4, presenting the kernel density of the most influential 1% of patents, shows the difference between ICT & non-ICT is substantial. Figure 4 also shows there is greater probability of ICT patents with high citation counts and PageRank values than non-ICT patents.

Illustrating this effect, Table 5a includes the summary statistics comparison of the PageRank of the top 1%, 0.1%, and 0.01% for ICT and non-ICT patents, and Table 5b shows the PageRank comparison of the top 20 patents in our data. These examples help illustrate the greater PageRanks for ICT patents, as well as the types of patents that drive this effect.

We conclude that our analysis points to an outsized influence of ICT patents on subsequent patents from other technology sectors.

Table 5. PageRank comparison for top patents; (a) top 1%, 0.1%, and 0.01% patents; and (b) top 20 patents (in terms of PageRanks).

	PageRank ($x10^9$)	
Percentile	**ICT**	**non-ICT**
1.00%	265.31	196.79
0.10%	832.91	693.35
0.01%	2,365.63	2,218.22

(a)

Rank	Application ID	Application Title	ICT	PageRank ($x 10^9$)	Year of publication	Technology Sector
1	53256578	Data processing system	Y	41481.03	1962	Electrical engineering
2	47125931	Data processing system	Y	35125.14	1965	Electrical engineering
3	45398511	Process for producing biologically functional molecular chimeras	N	34734.46	1980	Chemistry
4	49093248	Microorganisms having multiple compatible degradative energy-generating plasmids and preparation thereof	N	29760.38	1974	Chemistry
5	46785617	Multiplying and dividing means	Y	26573.25	1952	Electrical engineering
6	49374877	Method for the direct analysis of sickle cell anemia	N	24223.85	1983	Chemistry
7	53246938	Information handling apparatus	Y	23040.61	1962	Electrical engineering
8	53231598	Masers and maser communications system	N	22022.31	1960	Instruments
9	46167804	Diagnostic reagent	N	21880.72	1981	Chemistry
10	53683125	Process for amplifying, detecting, and/or-cloning nucleic acid sequences	N	20769.3	1987	Chemistry
11	53300809	Method for the determination of antigens and antibodies	N	19816.43	1972	Instruments
12	51694170	Atomic or molecular oscillator circuit	Y	19706.05	1958	Electrical engineering
13	53477242	Process for amplifying nucleic acid sequences	N	18441.05	1987	Chemistry
14	50191114	Mobile communication system	Y	18395.47	1972	Electrical engineering
15	50741692	Method of automatically evaluating source language logic condition sets and of compiling machine executable instructions	Y	18153.26	1982	Electrical engineering
16	48279565	Automatic message switching system	Y	17705.71	1954	Electrical engineering
17	51780454	Software version management system	Y	16791.97	1985	Electrical engineering
18	48815898	Controlling arrangements for electronic digital computing machines	Y	16302.78	1957	Electrical engineering
19	47754682	Digital servo	Y	15920.56	1951	Instruments
20	46130692	Three-electrode circuit element utilizing semiconductive materials	Y	15679.73	1950	Electrical engineering

(b)

Conclusion

Using complete patent data available for more than 100 years from 160 countries comprising 90 million patents, we have shown that ICT patents are consistently more influential than patents from other industrial sectors. We used simple, iterative metrics to support our findings using the full lifetime of patents or shorter time spans. Whereas other studies have highlighted the importance of ICT for productivity and economic development, we have quantified the direct influence of ICT inventions on other technologies.

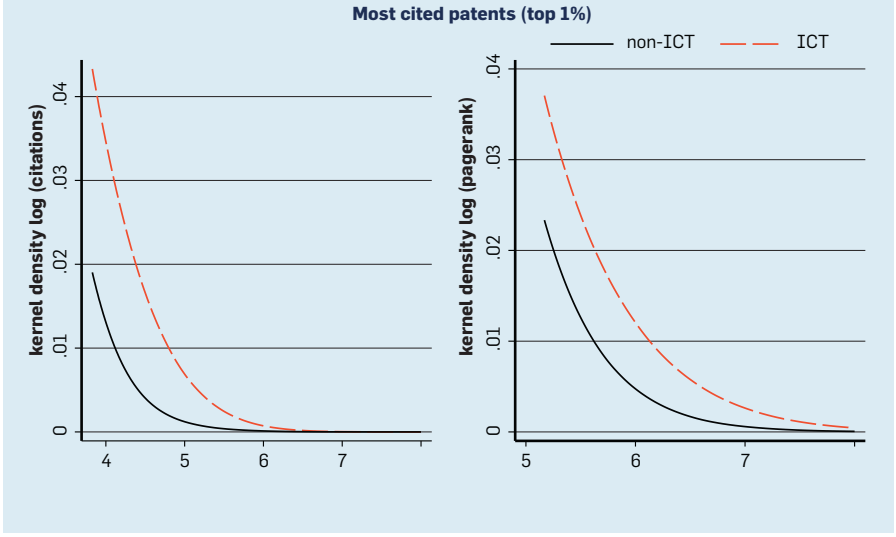
We emphasize the results obtained through the PageRank approach because it most accurately measures the influence of inventions. PageRank highlights the role of specific ICT inventions in enabling the invention of other influential technologies. With it, ICT patents do not appear quite as influential as with the coarser methods of descriptive statistics, but we still find a statistically and economically significant difference between ICT patents and the patents from other sectors. While the propensity for a greater number of patents and patent citation within the ICT sector may exaggerate the influence of ICT, we use PageRank to capture "cumulative" influence and still find ICT patents have an outstanding effect on subsequent technological development. Our findings thus complement and extend earlier studies of the societal effect of ICTs.

Our analysis is not directly able to explain conclusively why ICT inventions are substantially more influential than others but points to three possible factors we hope open avenues for further research: ICTs may be more cumulative than other technology fields, whereby subsequent inventions build more closely on previous inventions; as a field with a strong scientific knowledgebase, ICT inventions may depend on scientific breakthroughs that enable cumulative invention of industrial applications and lead to highly cumulative patterns of patent citations. ICTs may be more "generative" than other technologies; such generativity stems from the openness of ICT systems, by design, to enable complementary applications, and from the inherent flexibility of ICTs that creates technological opportunity for invention. And finally, ICTs may have exceptional influence on invention in a range of technological fields because they enable the capture, manipulation, and communication of information itself, and information is the fundamental ingredient of invention. In a related study, we call ICTs "invention machines" and further explore the nature of ICTs as general-purpose invention technologies.[15] C

References

1. Agrawal, A. and Goldfarb, A. Restructuring research: Communication costs and the democratization of university innovation. *American Economic Review 98*, 4 (Sept. 2008), 1578–1590.
2. Becchetti, L., Bedoya, D.A.L., and Paganetto, L. ICT investment, productivity and efficiency: Evidence at firm level using a stochastic frontier approach (in English). *Journal of Productivity Analysis 20*, 2 (Sept. 2003), 143–167.
3. Bertschek, I., Cerquera, D., and Klein, G.J. More bits, more bucks? Measuring the impact of broadband Internet on firm performance. *Information Economics and Policy 25*, 3 (Sept. 2013), 190–203.
4. Bloom, N., Sadun, R., and Ven Reenen, J. Americans do IT better: U.S. multinationals and the productivity miracle. *American Economic Review 102*, 1 (Feb. 2012), 167–201.
5. Bresnahan, T.F. and Greenstein, S. The economic contribution of information technology: Towards comparative and user studies. *Journal of Evolutionary Economics 11*, 1 (Jan. 2001), 95–118.
6. Bresnahan, T.F. and Trajtenberg, M. General-purpose technologies: Engines of growth? *Journal of Econometrics 65*, 1 (Jan. 1995), 83–108.
7. Brynjolfsson, E. and Hitt, L.M. Computing productivity: Firm-level evidence. *Review of Economics and Statistics 85*, 4 (Nov. 2003), 793–808.
8. Czernich, N., Falck, O., Kretschmer, T., and Woessmann, L. Broadband infrastructure and growth. *The Economic Journal 121*, 552 (May 2011), 505–532.
9. Dechezleprêtre, A., Martin, R., and Mohnen, M. Knowledge spillovers from clean and dirty technologies: A patent-citation analysis. *Centre for Climate Change Economics and Policy Working Paper No. 151 and Grantham Research Institute on Climate Change and the Environment Working Paper No. 135.* London School of Economics and Political Science, London, U.K., 2013.
10. Forman, C. and van Zeebroeck, N. From wires to partners: How the Internet has fostered R&D collaborations within firms. *Management Science 58*, 8 (Aug. 2012), 1549–1568.
11. Hall, B.H., Jaffe, A.B., and Trajtenberg, M. Market value and patent citations. *RAND Journal of Economics 36*, 1 (Spring 2005), 16–38.
12. Jaffe, A.B. and Trajtenberg, M. International knowledge flows: Evidence from patent citations. *Economics of Innovation and New Technology 8*, 1-2 (1999), 105–136.
13. Kim, P.R. and Hwang, S.H. A study on the identification of cutting-edge ICT-based converging technologies. *ETRI Journal 34*, 4 (Aug. 2012), 602–612.
14. Koutroumpis, P. The economic impact of broadband on growth: A simultaneous approach. *Telecommunications Policy 33*, 9 (Oct. 2009), 471–485.
15. Koutroumpis, P., Leiponen, A., and Thomas, L.D.W. Invention machines: How control instruments and information technologies drove global technological progress over a century of invention. Working Paper. Imperial College London, U.K., 2017.
16. Page, L., Brin, S., Motwani, R., and Winograd, T. *The PageRank Citation Ranking: Bringing Order to the Web.* Technical Report. Stanford InfoLab, Mountain View, CA, 1999; http://ilpubs.stanford.edu/422/1/1999-66.pdf
17. Solow, R. We'd better watch out. *The New York Times Book Review* (July 12, 1987).
18. Trajtenberg, M. A penny for your quotes: Patent citations and the value of innovations. *RAND Journal of Economics 21*, 1 (Spring 1990), 172–187.
19. Van Reenen, J., Bloom, N., Draca, M., Kretschmer, T., and Sadun, R. *The Economic Impact of ICT.* In *Final Report*. Center for Economic Performance, London School of Economics, London, U.K., 2010.

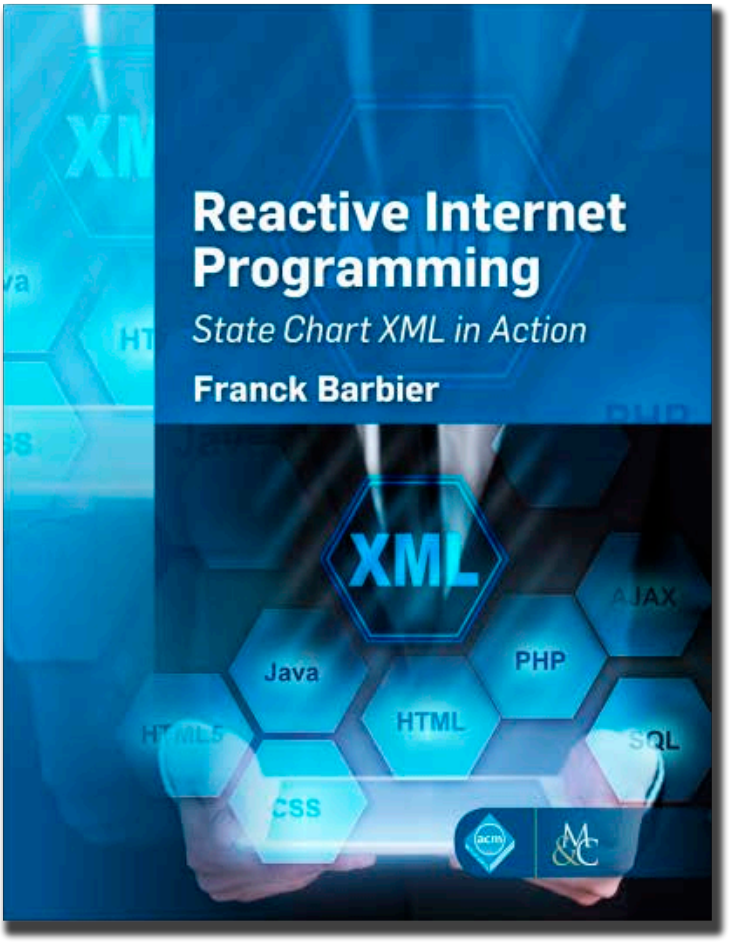
Figure 4. Kernel densities for distributions of (left) top (1%) patent citations and (right) PageRank.



Pantelis Koutroumpis (p.koutroumpis@imperial.ac.uk) is a research fellow at Imperial College Business School, London, U.K., and a fellow of the Columbia Institute of Tele-Information at Columbia University, New York.

Aija Leiponen (aija.leiponen@cornell.edu) is an associate professor in the Dyson School of Applied Economics and Management, Cornell University, Ithaca, NY, and a visiting research fellow in the Research Institute of the Finnish Economy, Helsinki, Finland.

Llewellyn D W Thomas (llewellyn.thomas@imperial.ac.uk) is a visiting professor in the Innovation and Entrepreneurship Department of Imperial College Business School, London, U.K.



You need effective means to put reactive programming into practice.

THEY'RE RIGHT HERE

Is Internet software so different from "ordinary" software? This book practically answers this question through the presentation of a software design method based on the State Chart XML W3C standard along with Java. Web enterprise, Internet-of-Things, and Android applications, in particular, are seamlessly specified and implemented from "executable models."

Internet software puts forward the idea of event-driven or reactive programming, as pointed out in Bonér et al.'s "Reactive Manifesto". It tells us that reactiveness is a must. However, beyond concepts, software engineers require effective means with which to put reactive programming into practice. *Reactive Internet Programming* outlines and explains such means.

The lack of professional examples in the literature that illustrate how reactive software should be shaped can be quite frustrating. Therefore, this book helps to fill in that gap by providing in-depth professional case studies that contain comprehensive details and meaningful alternatives. Furthermore, these case studies can be downloaded for further investigation.

Internet software requires higher adaptation, at run time in particular. After reading *Reactive Internet Programming*, you will be ready to enter the forthcoming Internet era.

DOI:10.1145/3035966

Econometrics is a key component to gauging user satisfaction and advertisers' profits.

BY DENIS NEKIPELOV AND TAMMY WANG

Inference and Auction Design in Online Advertising

THE TRANSITION OF the largest online advertising platforms to auction-based automated real-time designs has transformed the advertising industry. The advertisers had an opportunity to design flexible goal-specific advertising campaigns, target focused small groups of consumers, and perform fast and efficient experimentations. At the same time, consumers can be exposed to a smaller number of higher quality advertisements. However, in order to achieve the design goals, such as optimal placement of ads that maximizes the sum of utilities of users or revenue generated by the auction platform, the auction needs to be carefully optimized. To optimize the auction, the auctioneer typically chooses a relatively small number of auction parameters that determine the allocation of items and prices based on submitted bids. Typical auction parameters include reserve prices, which determine the minimum bid and can correspond to any allocation. Another parameter used in sponsored search auctions is the quality score, which is a positive bidder-specific weight used to discount or inflate submitted bids before they are ranked. The setting of auction parameters requires a knowledge of advertisers' preferences and consumers behavior, which can be acquired from data. This makes econometric inference from observed data of high importance for the design and analysis of online advertising auctions.

The structure of online advertising exchanges is becoming significantly complex, and requires multiple parameters to be input by auction designers. These parameters are required to yield consistent advertising allocations, relevant user ad experiences over time and catering ad placements to advertisers' goals. This operational structure of online ad exchanges has been incorporated into the algorithmic implementation of advertising auctions, see Muthukrishnan,[29] Aggarwal and Muthukrishnan,[30] and Muthukrishnan[31] for details of such an implementation. However, the increased heterogeneity and dynamics of the marketplace call for quick "on demand" adjustments of auction parameters in order to pursue auction platform revenue goals and relevant ad experiences for users. Recent evidence from the sponsored search advertising

» key insights

- **Modern advertising platforms are automated systems powered by auctions that allocate and price advertising slots contemporaneously based on the bids submitted.**
- **Real-time auction-based systems require good predictions of user behavior, need to adapt to the changing marketplace, and require fast and robust methods for evaluating performance.**
- **These challenges can be addressed by applying econometric methods to the data from advertising platforms.**

COLLAGE BY ANDRIJ BORYS ASSOCIATES/SHUTTERSTOCK

marketplace indicates that there are significant gains (both in terms of overall welfare and ad exchange revenue) from adopting data-driven designs. For example, Ostrovsky and Schwarz[34] described the experiment on Yahoo!'s search engine where the advertisers 'per-click values were recovered from observed bids. The distribution of recovered values was then used to set search query-specific reserve prices in the advertising auctions. The data shows a significant increase in the search engine's revenue resulting from this switch.

As with other economic markets, advertising markets balance supply and demand. The webpage content attracts consumers and supplies their "attention." Advertisers demand user attention since it converts advertised products and services to purchases. As in all economic markets, the equilibrium of demand and supply is driven by price. The dominant pricing model for advertising puts payment control on the side of advertisers and lets them determine their preferred prices. At the same time, consumers receive a "free" service. This feature has made advertiser behavior of direct importance for monetization and, therefore, has been studied in detail in recent years.

We survey the work devoted to the analysis of advertiser behavior, starting from market equilibrium models to more recent work where advertisers must adopt learning strategies to set their bids. We then discuss the "supply" side of the market, which to a large extent remains model-free. Recent empirical work has focused on determining actual advertisers' value extracted from users. This work turned out to be significantly more delicate since user behavior is highly correlated with user characteristics, and thus, it is very difficult to distinguish the effect of advertising from the inherent propensity of particular users to purchase particular products. Finally, we describe recent changes in user behavior caused by the transformation of browsing and purchasing patterns generated by new devices, and new ways of accessing the Internet. We predict this trend will generate the need to do more in-depth consumer-side research of advertising markets.

The complexity of advertising markets highlights the importance of inference, as inference informs customized market-specific design and optimization of advertising markets. Here, we review existing and emerging approaches to inference for advertising auctions both aimed at the diagnostic analysis of advertising marketplaces and as an input to decisions regarding marketplace changes.

Platform Design and Advertiser Behavior

Bidder value inference. While the general auction platform design can be based on pure economic theory, platform optimization requires the knowledge of the bidders' underlying preferences and the prediction of user action. This knowledge is derived from the auction outcomes data. The general problem of data-driven auction design can be formulated as the *counterfactual inference*—from observations of agent behavior in the platform, we want to predict how they will behave if the platform design is changed. With the new design, participating agents will re-optimize their strategies, and, thus, this new design leads to the change in bids. Counterfactual inference, therefore, requires the platform to anticipate strategic responses of bidders to the mechanism change.

The counterfactual inference is based on the economic model that characterizes agent behavior on the platform. Based on this model, one can use data to infer the "primitives" of the model (such as agent preferences) that are consistent with the observed behavior. The prediction for the new design will be based on computing bidders' responses to the new auction mechanism, and the changed bids of the other bidders.

In an advertising auction model, the utility of a bidder (that characterizes the expected gain of the bidder from participating in an auction) is typically set equal to the click probability multiplied the bidder's value per-click less the cost. The click probability depends on both the identity of the bidder and the placement of the ad. Therefore, it ultimately depends on the bid that determines where the ad is placed. In this model, we characterize each bidder by a single parameter that expresses this bidder's value per click. Inference of bidders' values per click is, therefore, one of the key ingredients of counterfactual inference. Another ingredient is the computation of the *Bayes-Nash* equilibrium of the auction.

The Bayes-Nash equilibrium of an auction is the set of bids and beliefs of bidders regarding the distribution of uncertainty in the auction such that the bids optimize bidders' expected utilities, and the beliefs of bidders correctly capture the realized distribution of uncertainty. When bidders use Bayes-Nash equilibrium strategies and the equilibrium is unique, the values can be inferred directly from data by inverting bidders' best response functions. The inferred bidder value makes the observed bid the best response to the observable empirical distribution of opponent bids. Athey and Nekipelov[3] adopted this strategy to infer bidders' values in sponsored search auctions. Sponsored links that appear beside Internet search results on major search engines are sold using real-time auctions. Advertisers place standing bids where they are associated with search phrases that form part or all of a user's search query. Each time a user enters a search query, applicable bids are entered to an auction. The ranking of advertisements and the prices paid depend on advertiser bids as well as "quality scores" assigned for each advertisement and user query. These quality scores are traditionally set equal to the predicted probability of a user click on an ad. They vary over time because the statistical algorithms predicting quality produce different predictions based on the features of the user issuing the search query.

In theory, the allocation and payment functions in standalone position auctions are discontinuous with respect to bid, because the price only changes when the bidder goes up in the ranking with an increased bid. However, in ad auctions each bid is typically applied to many auctions. With sufficient uncertainty over auctions, the expected auction outcomes will be continuous. As a result, to accurately characterize the behavior of bidders in this setting, Athey and Nekipelov[3] propose a model that has these properties.

The model proposed by Athey and

Nekipelov[3] deviates from the standard model of a Bayesian auction game where the values of the players are drawn from the distribution. The main reason for this deviation is that in real-world advertising auctions the sets of bidders are fixed and most bidders' bids do not change over long time intervals. Athey and Nekipelov's model rationalizes this observation. In the rest of the survey we adhere to the model of Athey and Nekipelov and assume the values of the bidders are fixed.

Consider the model where there are I bidders whose ads are eligible to be displayed to a user alongside the search results and to be placed in J available advertising positions. The user clicks on the ad i shown in the position j with probability c_{ij}, where $c_{ij} = \alpha_j \cdot s_i$ is the product of the ad-specific click probabillity s_i (for $i = 1, \ldots, I$) and the advertising slot-specific click probability α_j (for $j = 1, \ldots, J$). The bidder-specific click probability s_i for the particular bidder i is a random variable whose realizations vary across search queries of user. The randomness of the bidder-specific click probability reflects different propensities of different users to click on a given ad.[a] The bidder-specific click probability s_i is referred to as the bidder's *quality score* (or, simply, the score).

This description shows how an ad platform conducts a score-weighted auction. In each user query, the platform runs an auction where each advertisement i is assigned the score s_i, and bids are ranked in order of the product $b_i \cdot s_i$. The per-click price p_i that bidder i in position j pays is determined as the minimum price such that the bidder remains in her position,

$$p_i = \min\{b_i : s_i b_i \geq s_k b_k\} = s_k b_k / s_i, \quad (1)$$

where bidder k is in position $j + 1$. Note that advertiser i's bid does not directly influence the final price she gets charged, except when it causes to change positions. In effect, an advertiser's choice of bid determines which position she attains, where the price per click for each position is exogenous to the bid and rises with position. An auction with this type of pricing rule is called the (score-weighted) *generalized second price auction*.

Advertisers demand user attention since it converts advertised products and services to purchases.

Per-click values of bidders v_i for $i = 1, \ldots, I$ are fixed for each bidder across user queries and are supported on $[0, \bar{v}]$. The pay-off of bidder i in a query where this bidder receives a click is the surplus $v_i - p_i$, where p_i is bidder i's price per click defined by Equation (1).

While we assume all I bidders are eligible to be displayed in each user query, the actual sets of participating bidders in a real search query page will vary. For instance, some bidders may be excluded from certain queries based on the targeting settings (that is, advertisers may prefer to advertise in specific geographic locations). Suppose each bidder knows the entire set of bidders I, but can not observe the set of competitors in a given query and can not observe neither her own nor her competitors' scores. In this case, bidders form beliefs regarding the distribution of scores of all bidders (since they affect the prices in individual user queries), and beliefs regarding the distribution of realizations of sets of their competitors across user queries. The standard assumption in the sponsored search auction literature (for example, Edelman et al.[16] and Varian[41]) is that bidders have full information regarding each other's bids. This reflects the idea of a high frequency environment where bidders can quickly determine the bids of their opponents, even if the auction platform does not explicitly provide that information. Bidder i then maximizes the expected pay-off (with per click value) with the bidder's beliefs regarding the distribution of uncertainty of scores and sets of competitors. This pay-off (a.k.a. utility) can be expressed as,

$$U(b_i, b_{-i}; v_i) = E[c_{ij}(v_i - p_{i_j})], \quad (2)$$

where the expectation E is taken over the distribution of scores $s_1, \ldots, s_I$, and the distribution of sets of bidders participating in the auction. We emphasize that the pay-off depends on the bid of the bidder i (b_i) as well as the bids of all competing bidders (b_{-i}) that may determine the rank and the price that bidder i pays depending on a realization of the set of scores.

a In practice, s_i is a prediction from a machine learning algorithm run by the search engine that uses a large set of user features which vary from query to query.

Here, we consider the question of bidders' per-click values that can be recovered from data that contains a large number of queries for a given set of potential bidders. For each query, the data available to the advertising platform contains the bids, a set of entrants, and the bidders' scores relating to each user query.

In this case, we can define function $Q_i(\cdot)$ to be equal to the probability of click on the ad of bidder i in a search query as a function of all bids. We also define function $TE_i(\cdot)$ equal to the expected payment of bidder i in a search query as a function of all bids. The utility of bidder i can be written in terms of these functions as,

$$U(b_i, b_{-i}; v_i) = v_i\, Q_i(b_i, b_{-i}) - TE_i(b_i, b_{-i}).$$

The best-responding bid maximizes the expected utility given the bids of opponent bidders (b_{-i}). Athey and Nekipelov[3] demonstrate that with sufficient smoothness and support size of the distribution of scores, functions $Q_i(\cdot)$ and $TE_i(\cdot)$ are strictly increasing and differentiable with respect to bids. As a result, the value per click for each bidder can be recovered using the necessary first order condition for optimality of the bid.

$$v_i = \frac{\partial\, TE_i(b_i, b_{-i})}{\partial b_i} \Big/ \frac{\partial\, Q_i(b_i, b_{-i})}{\partial b_i}. \quad (3)$$

Functions $Q_i(\cdot)$ and $TE_i(\cdot)$ can be recovered directly from the data[b] (for example, using splines or any other appropriate method). Thus, the per-click value can also be recovered from a data using Equation (3). Smoothness, and monotonicity of $Q_i(\cdot)$ and $TE_i(\cdot)$ can be directly tested in data and has been verified using the auction data by Athey and Nekipelov.[3]

Equation (3) provides a simple practical method for estimating value per click. For each bidder we change bid by a small amount and then compute the change in the outcome of the auctions where the original bid of the bidder was applied. The evaluated change in expenditure will serve as an estimator for the derivative of $TE_i(\cdot)$ function and the evaluated change in number of clicks will serve as an estimator for the derivative of $Q_i(\cdot)$ function.

An advertiser's choice of bid determines which position she attains, where the price per click for each position is exogenous to the bid and rises with position.

Evaluation of new auction mechanisms using data. The approach adopted in the empirical economics literature (see Bajari et al.[4,5]) provides a simple recipe for platform optimization subject to the inferred bidder preferences, *assuming the bids constitute the Nash equilibrium* (defined previously in terms of best responses and beliefs of bidders) under the new settings. To follow this recipe, first we estimate the components of the first-order condition (Equation 3) using data from historical realizations of auctions. Value per click can then be reverse engineered from the bids for each bidder. Second, we plug in the new proposed auction parameters (or consider a new auction format). For each set of bids we can simulate the auction outcomes under the new auction parameters. These new auction outcomes (prices and allocations for hypothetical bids under the new auction rules) will generate *counterfactual* functions $\widetilde{TE}_i(b_i, b_{-i})$ and $\widetilde{Q}_i(b_i, b_{-i})$ for each bid profile. Finally, assuming the system converges to the long-term Nash equilibrium under new auction rules, and the values of the bidders do not change, we can predict the new bids. To do so, we find the set of bids for all participating bidders such that the bid of each bidder maximizes her utility under the new auction rules given the bids of other bidders. Since the maximum utility will be attained at the point where its derivative with respect to the bid is equal to zero, finding the new equilibrium bids is reduced to solving the system of nonlinear equations for $i = 1, \ldots, I$

$$\frac{\partial \widetilde{TE}_i(b_i^*, b_{-i}^*)}{\partial b_i} \Big/ \frac{\partial \widetilde{Q}_i(b_i^*, b_{-i}^*)}{\partial b_i} = v_i, \quad (4)$$

using values v_i inferred from the data (using Equation (3)) and solve for the set of bids $b_1^*, \ldots, b_I^*$ that makes all I equalities hold. Unlike Equation (3) where we use actual bids b_i to obtain the values, in Equation (4) bids b_i^* are the

b Statistical properties of the corresponding estimators, such as the rate of convergence, are determined by the either the complexity of the circuit that is used to compute the allocations and payments from the bids or using their functional properties such as smoothness or concavity directly. The methods that can be used to establish these statistical properties parallel those applied to derive sample complexity of mechanisms such as Cole and Roughgarden,[14] Cummings,[15] and Morgenstern and Roughgarden.[28]

outcome and the values are the input. We note that this strategy only works if functions $TE_i(\cdot, b_{-i})$, $Q_i(\cdot, b_{-i})$, $\widetilde{TE}_i(\cdot, b^*_{-i})$ and $\widetilde{Q}_i(\cdot, b^*_{-i})$ are strictly monotone and the equilibrium is unique.

Monotonicity of payment and allocation functions along with the uniqueness of the Nash equilibrium allow us to use numerical continuation methods to find the new equilibria. The numerical continuation approach to finding this new equilibrium is based on transforming the set of conditions Equation (4) into system of ordinary differential equations. Athey and Nekipelov[3] show that solving this system of differential equations is equivalent to finding the new equilibrium (that is, it will not diverge or generate new extraneous solutions). The solution of numerical approximation of ordinary differential equations will yield the new equilibrium.

Although formulating the equilibrium computation problem in terms of solving the system of differential equations is mathematically elegant, the process can be complicated since it requires many calls to functions $\widetilde{TE}_i(\cdot, b^*_{-i})$ and $\widetilde{Q}_i(\cdot, b^*_{-i})$, which need to be computed from the data for each evaluation. This problem is further accentuated in most advertising markets where the large number of eligible bidders participating in each auction leads to difficulties in computing the new equilibrium, even when sufficient conditions hold for existence and uniqueness of the Nash equilibrium.

A significant simplification in equilibrium settings can be achieved when the object of interest is not the entire vector of bids, but a specific lower dimensional auction outcome, such as revenue or welfare. When a simple outcome is defined, the entire vector of counterfactual equilibrium bids is no longer of direct interest, which effectively reduces the scale of the computational problem. Next, we consider this approach in application to social welfare in an auction.

The price of anarchy bounds. While equilibrium-based analysis can be widely applied, it is largely based on the idea that bidders use equilibrium strategies and the corresponding set of bids constitutes a Nash equilibrium. A common assumption used in the economics literature is the equilibrium is unique in a given market. In practice, however, it is impossible to guarantee that sufficient conditions for Nash equilibrium uniqueness are satisfied in a given auction mechanism. For instance, the non-monotonicity of either the payment or allocation function leads to several possible values for each click that are consistent with observed bids. Then the simple equilibrium-based analysis is no longer valid.

The economics literature has developed an approach to the analysis of data from outcomes of games with multiple equilibria and games in non-equilibrium settings (for example, see Aradillas-Lopez et al.[1] and Beresteanu et al.[6]). The approach is to directly consider the optimality conditions for the players in these new settings. In auctions such optimality conditions lead to sets of possible values of bidders referred to as *identified sets*. This approach, however, has proven to be very difficult with reported computational time significantly exceeding those in the equilibrium framework even for simple games (see Ciliberto and Tamer[13]).

Koutsoupias and Papadimitriou[23] have developed a theoretical framework for analyzing simple auction outcomes such as revenue and welfare. The framework produces bounds for these outcomes comparing them to the welfare-maximizing allocation over all feasible realizations of bidders' values. This approach is referred to as *the price of anarchy* approach, which is defined as the ratio of the worst-case objective function value of the outcome of a game (such as Nash equilibrium outcome) and that of an optimal outcome. It turns out the price of anarchy can be established under minimal assumptions on the underlying preferences for many commonly used auction mechanisms (for instance, see Roughgarden[36]).

There are two difficulties with practical application of the price of anarchy bounds for a given game. First, these bounds can be quite conservative. Given they are derived taking the worst case equilibrium and the value profile, these bounds can be large. However, the values of players and the equilibria that generate the worst-case outcomes may not occur in reality. Even when the data from the mechanism is available, the price of anarchy bounds do not take it into account. Second, the price of anarchy is mechanism-specific and has so far been derived on the case-by-case basis.

Hoy et al.[20] bridge the gap between the robust but coarse theoretical price of anarchy bounds and precise but difficult to compute identified sets. The idea is to integrate data directly into the price of anarchy style analysis that can be applied to large classes of existing auction mechanisms. This new concept is called the *empirical price of anarchy*.

Here, we discuss the idea behind the empirical price of anarchy for auction settings where bidders adhere to playing full information equilibrium best responses in the auction mechanism A, but where those best responses are not unique (for example, due to non-monotonicity of allocations and prices in the auction). We assume, like in the model of Athey and Nekipelov,[3] the auction parameters are random while the profile of bidders' values is fixed and we use $\mathbf{V} = (v_1, \ldots, v_I)$ to denote that profile. For a given profile of values auction A can have multiple Nash equilibria that constitute set $\Sigma(A, \mathbf{V})$. Combined with the distribution of auction parameters across user search queries, an equilibrium (which is an element of $\Sigma(A, \mathbf{V})$) generates the distribution of observable auction outcomes (allocations and payments of bidders) $D(\cdot)$ across those user queries.

If the set of all equilibria $\Sigma(A, \mathbf{V})$ is not a singleton, then there will be many equilibrium distributions of outcomes $D(\cdot)$ that correspond to value profile $\mathbf{V}$. Conversely, for each distribution of auction outcomes $D(\cdot)$ there may be multiple value profiles of the bidders that could have generated that distribution of outcomes. Our next step is based on exploring this information to make statements regarding the counterfactual welfare (the sum of bidder utilities and the revenue of the auctioneer) in auctions.

We define the Empirical Bayesian Price of Anarchy (EPoA) as the worst-case ratio of welfare of the optimal allocation to the welfare of an equilibrium in a given auction A, taken over all value profiles of the bidders $\mathbf{V}$ and Nash equilibria that could have generated the observed distribution of auction outcomes $D(\cdot)$. The notion of EPoA allows us to provide bounds on the (unobserved)

welfare of the optimal allocation (OPT) and the welfare of the auction A that is actually implemented,

$$\text{WELFARE(OPT)} \le \text{EPoA}(A; D) \cdot \text{WELFARE}(A). \quad (5)$$

Thus, the EPoA is the characteristic of an auction that allows us to measure the efficiency of the current auction mechanism without estimating (the set of) values of the bidders.

The empirical price of anarchy (and, subsequently, the bound for optimal welfare) is defined for the true distribution of auction outcomes $D(\cdot)$. Then, the idea is to replace the true distribution of auction outcomes with empirical distribution $\widehat{D}(\cdot)$ of auction outcomes observed in the data. Given that the empirical distribution is a consistent estimator of the true distribution[c] the bound for the welfare constructed using EPoA $(A; \widehat{D})$ will bound the actual auction welfare with probability approaching one (as we get more samples from the distribution of auction outcomes $D(\cdot)$). We call EPoA $(A; \widehat{D})$ the *estimator* for EPoA.

We note that even the estimator for EPoA is defined as a potentially complex constrained optimization problem. It turns out that it is possible to avoid solving this problem by invoking the *revenue covering* approach. The revenue covering approach is based on establishing the ratio of the actual auction revenue and maximum payment that can be extracted from participating bidders in equilibrium. This ratio can be used to establish a simple bound for EPoA. We now describe this approach in more detail in application to the sponsored search auction.

Consider the sponsored search auction model with uncertainty as we described in detail. We can define the average price per click for bidder i with bid b_i as $\text{ppc}_i(b_i) = \text{TE}_i(b_i)/\text{Q}_i(b_i)$. The typical function that provides the expected number of clicks as a function of the bid $\text{Q}_i(b_i)$ in the sponsored search auction is continuous and monotone. As a result, we can construct its inverse $b = \text{Q}_i^{-1}(z)$ that specifies the bid that is needed to get the expected number of clicks z. Then, the average price per click can be redefined in terms of expected clicks as $\text{ppc}_i(z) = \text{ppc}_i(\text{Q}_i^{-1}(z))$, which is the average price per click that bidder i getting z clicks pays. Function $\tau_i(z) = \text{ppc}_i^{-1}(z)$ is called the *threshold* because it corresponds to the minimum price per click that bidder i needs to pay to get z clicks in expectation. The *cumulative threshold* for agent i who gets Q_i expected clicks is $T_i(\text{Q}_i) = \int_0^{\text{Q}_i} \tau_i(z)dz$ $T_i(\text{Q}_i)$ can be interpreted as the total payment that bidder i would have paid, had she purchased each additional click at the average price of previously purchased clicks when she purchases a total of Q_i clicks.

Definition 1. Strategy profile σ of auction A (defining the mapping from bidders' values into their bids) is μ-revenue covered if for any feasible allocation $\{\text{Q}_i\}_{i=1}^I$

$$\mu\text{Revenue}(A, \sigma) \ge \Sigma_i T_i(\text{Q}_i). \quad (6)$$

Auction A is μ-revenue covered if for *any* strategy profile σ, it is μ-revenue covered.

The inequality Equation (6) can be defined in expectation over the realizations of possible equilibrium profiles and all thresholds corresponding to the realizations of bid profiles. Then, the expected revenue from the auction can be measured directly by the auction platform (as a sum of payments of bidders per auction over the observed auction realizations). The thresholds can be computed via simulation from the observed auction realizations given that the allocation and pricing rule is controlled by the auction platform, and thus empirical equivalents of $\text{TE}_i(\cdot)$ and $\text{Q}_i(\cdot)$ can be computed for each bid b_i. As a result, the platform can compute the revenue covering parameter for given auction mechanism A.

The next result, developed in Syrgkanis and Tardos,[37] takes revenue covering parameter and provides the EPoA for the auction mechanism A.

Theorem 1. *The welfare in any μ-revenue covered strategy profile σ of auction A is at least a $\frac{\mu}{1-e^{-\mu}}$ -approximation to the optimal welfare. In other words, EPoA* $(A; D) \le \frac{\mu}{1-e^{-\mu}}$.

The estimator for the EPoA is then obtained by replacing the true revenue covering parameter μ with its estimator recovered from the data. As a result, this approach allows one to establish the bounds for auction welfare bypassing complex computations required in the approaches previously used in the economics literature by combining statistical inference and results from the algorithmic game theory. The Syrgkanis and Tardos[37] approach may potentially be applied to other bounds, for example, comparing the welfare of a given auction mechanism to the welfare of another auction mechanism (instead of the welfare of the optimal allocation). We believe that such an analysis is an important direction for future research.

Advertising markets with learning agents. Real-world advertising platforms are complex systems with thousands of bidders who compete in the same auction with bidders dynamically changing their bids, entering and exiting auctions. In this case, information requirements for bidders to derive their Bayes-Nash equilibrium profiles are truly impractical since they are required to form beliefs over the actions of all of their thousands of opponents, as well as the dynamic adjustment of auction parameters by the advertising platform.

In practice, most bidders in these large advertising platforms use algorithmic tools that allow them to automatically and dynamically update their bids for multiple ads and advertising campaigns. The algorithmic solutions implemented in these tools take the advertisers goals (in terms of yield of auction outcomes) as inputs, and adjust bids using dynamic feedback from the auction outcomes. Such implementations can be associated with algorithmic learning, where the bidding strategy is treated as the goal of online statistical learning procedure.

Recent work by Blum et al.,[8,9] Blum and Y. Mansour,[10,11] Caragiannis et al.,[12] Hartline et al.,[19] Kleinberg et al.,[22] Roughgarden,[36] and Syrgkanis and Tardos[37] shows that some of the worst case outcome properties of full information pure Nash equilibria extend to outcomes when all players use no-regret or low-regret learning strategies, assuming the game itself is stable. The assumption that players use low-regret learning to adjust their strategies is attractive for a number of reasons. First, low-regret learning outcome

c Formally, the infinity norm $\sup_t |\widehat{D}(t) - D(t)|$ converges to zero in probability under mild assumptions regarding $D(\cdot)$.

generalizes the Bayes-Nash equilibrium assumption. Rather than assuming that at any time players' actions form a Bayes-Nash equilibrium, it is only necessary to assume that each player has no-regret for any fixed action over a longer time period. Thus, this assumption reduces the requirement to gather player aggregate behavior over multiple runs of the game rather than the constraint of optimality in each game. If behavior of all agents is stable over the time period, this exactly fits the Nash equilibrium assumption; however, this also allows evolving play. Second, there are many well-known learning strategies that guarantee players achieve no regret, including the weighted majority algorithm[2] also known as Hedge,[17,26] and regret matching[18] just to name a few simple ones.

In the context of bidders using algorithmic strategies, we cannot directly estimate their preferences from observed actions. While in Nash equilibrium, bid is the expression of bidder's value per click, but with algorithmic strategy, the bid is solely the outcome of the algorithm and thus the link between the bid and the value can be lost if the algorithm only approximately optimizes the bidder's objective function. In this case there may be a range of possible values that are compatible with the observed bids.

Nekipelov et al.[33] use the concept of ϵ - *regret learning* to study online learning in sponsored search auctions. In online learning settings the regret of a given learning algorithm measures the difference between cumulative loss achieved by predictions given by the algorithm and loss of the best predictive function (a.k.a. "expert") from a given reference set. This concept can be applied to learning in repeated sponsored search auctions. The loss function will be associated with the negative utility in the sponsored search auction described earlier (that is, the loss minimization will correspond to the utility maximization). Nekipelov et al.[33] suggest set the strategies that adhere to a constant bid over the sequence of auctions. This idea is motivated by the empirical observation of Microsoft's Bing advertising platform, where a large fraction of bidders do not change bids over the lifetime of their advertisements.

Real-world advertising platforms are complex systems with thousands of bidders dynamically changing their bids, entering and exiting auctions.

Then, the average regret of bidder i over T periods against the expert that sets the constant bid b' can be evaluated as,

$$\text{Regret}_i(b') = \frac{1}{T}\sum_{t=1}^{T}(U_i(b', b_{-i,t}; v_i) - U_i(b_{i,t}, b_{-i,t}; v_i)),$$

where index t correspond to the time period. The sequence of bids $\{b_{i,t}\}_{i=1}^{T}$ achieves ϵ_i-average regret if for any expert b' (from the bid space) $\text{Regret}_i(b') \le \epsilon_i$. Based on this definition Nekipelov et al.[33] introduce the notion of a *rationalizable set*. We note that if a given sequence of bids $\{b_{i,t}\}_{t=1}^{T}$ has ϵ_i-average regret, then the value per click v_i of bidder i and the average regret ϵ_i satisfy the set of inequalities

$$\frac{1}{T}\sum_{t=1}^{T}(U_i(b', b_{-i,t}; v_i) - U_i(b_{i,t}, b_{-i,t}; v_i)) \le \epsilon_i$$

for each b'. Since bid sequences are observed in data, and the components of bidder utility (expected clicks and expected payment) can be simulated, these inequalities impose restrictions on pairs (v_i, ϵ_i). The set of these pairs is the rationalizable set is denoted as $\mathcal{NR}$. The range of bidder values contained in this set characterizes all possible values per click that rationalize the data from this bidder. Expected regret of a player reflects the properties of the learning algorithm used.

Under continuity and monotonicity of expected payment $\text{TE}_i(\cdot)$ and expected click $\text{Q}_i(\cdot)$ functions, Nekipelov et al.[33] establish basic geometric properties of the rationalizable set, such as its convexity and closeness. They also present an elegant geometric characterization that suggests an efficient algorithm for computing the rationalizable set. Since closed convex bounded sets are fully characterized by their boundaries, one can use the notion of the *support function* to represent the boundary of set $\mathcal{NR}$. The support function of a closed convex set $\mathcal{NR}$ is $h(\mathcal{NR}, \mathbf{u}) = \sup_{x \in \mathcal{NR}} \langle x, \mathbf{u} \rangle$. Geometrically, the support function is the distance and hyperplane that it is tangent to the set $\mathcal{NR}$ with normal vector $\mathbf{u}$. Since $\mathcal{NR}$ is a two-dimensional set, then $\mathbf{u}$ is a two-dimensional vector of unit length. An important result from convex analysis shows that support functions fully characterize closed convex sets.

Nekipelov at. al.[33] derive the support

function of the rationalizable set by defining the following functions that can be computed directly from the auction data via simulation

$$\Delta P(b') = \frac{1}{T}\sum_{t=1}^{T}(\mathrm{TE}_i(b', b_{-i,t}) - \mathrm{TE}_i(b_{i,t}, b_{-i,t})),$$

$$\Delta C(b') = \frac{1}{T}\sum_{t=1}^{T}(\mathrm{Q}_i(b', b_{-i,t}) - \mathrm{Q}_i(b_{i,t}, b_{-i,t})).$$

These functions characterize how an average outcome in T auctions changes when bidder i switches to a fixed alternate bid b' from an actually applied bid sequence. The characterization of the rationalizable set is given in the following theorem.

THEOREM 2. *Under monotonicity and continuity of $\Delta P(\cdot)$ and $\Delta C(\cdot)$ the support function of $\mathcal{NR}$ with $\mathbf{u} = (u_1, u_2)^T$ and $\|\mathbf{u}\| = 1$ is* $h(\mathcal{NR}, \mathbf{u}) = |u_2| \Delta C\left(\Delta P^{-1}\left(\frac{u_1}{|u_2|}\right)\right)$.

This theorem establishes that valuations and algorithm parameters for ϵ-regret algorithms can be recovered from the data (by computing functions $\Delta C(\cdot)$ and $\Delta P(\cdot)$). If the bids constitute the Nash equilibrium, we can pinpoint the bidders' value per click. As explained earlier, the initial stage of learning may not be a Nash equilibrium and there will be an entire range of values compatible with the data. At the same time, the characterization of this range of values and the entire rationalizable set for learning bidders are reduced to evaluation of two one-dimensional functions from the data. We can use efficient numerical approximation for such an evaluation. The corresponding error in the estimated rationalizable set will be determined by the estimation error of functions $\Delta C(\cdot)$ and $\Delta P(\cdot)$.

Platform Design and User Behavior

Inference of user actions. As discussed, the clickthrough rate measuring the probability that a given user will click on a given ad is a crucial input in pricing and allocation rules for advertising auctions. Advertising platforms use sophisticated machine learning tools to make such predictions on a user-by-user basis. However, despite a significant amount of effort and sophisticated statistical approaches, measuring the casual impact of advertising on the actions of users has been shown to be exceedingly difficult.[25] In addition, since a larger emphasis has been placed on the demand side of the advertising market (that yields revenue to advertising platforms), the work on the analysis of user behavior is significantly more sparse. Here, we briefly discuss some recent empirical results on the analysis of user behavior. This aspect of the market leaves a plethora of opportunities for future research.

Despite a significant amount of effort and sophisticated statistical approaches, measuring the causal impact of advertising on the actions of users has been shown to be exceedingly difficult.

The basic idea that explains the emergence of this problem is that user actions (such as the clicking on an ad) are correlated with observable and unobservable user characteristics. At the same time, ad delivery ensures ads are, by their very design, delivered to customers or during time periods that have different purchasing probabilities as compared to the baseline population. As a result, the straightforward approach based on predicting user action by using user characteristics can lead to overestimating the impact of advertising as shown in Blake et al.[7] and Lewis et al.[35]

To validate observational estimates of user behavior (such as the clickthrough rates for different advertising slots), advertising platforms rely on fully randomized experiments. These experiments are used by advertising platforms to infer important quantities of interest. For example, search engines exogenously re-order ad ranking and limit the number of ads shown in order to produce training data for many algorithms that govern ad delivery systems. These experiments are conducted on a small percentage of traffic so as to limit revenue risk, but nonetheless, still capture millions of searches. The experiments provide ground truth estimates of user behavior such as the causal effect of an ad as well as the impact of "organic" results on ad effectiveness. However, given their size, these estimates often need to be aggregated over many advertisers to obtain reasonable precision.

New directions for further work. Recently, advertising platforms have begun shifting toward advertiser goal-based pricing. In other words, if an advertiser's ultimate goal is sales to customers, then an advertiser may not be interested in the clicks per se. However, the noise in advertiser-specific estimates makes "reasonable size" experiments inadequate for supporting market design solutions that would have advertiser only pay for marginal clicks (i.e., the clicks that were actually

caused by the ad). This is because search engines, quite naturally, do not want to rest a pricing solution on the estimates with a large amount of noise with millions, if not billions, of dollars at risk. The current practice is for advertisers to figure their true return to advertising for themselves. This approach is inefficient because the platform possesses far more information to conduct the inference, and thus is in a much better position conduct price-to-value conversion. New generations of platform design will break through the bottleneck of the inefficiency leading to a decreased loss of social welfare and a potential increase in the platform revenue.

The detailed logging of digital advertising, combined with scale achieved on major platforms, opens the potential for new observational methods that could fulfill the promise of routinely measuring advertising effectiveness in an unbiased, precise manner. Inference of user intentions and prediction of user actions are exceedingly difficult outside of search engines where advertisers explicitly specify queries as proxy of intention. In social-, entertainment-, or task-oriented contexts, rendering potential advertisers in front of a user via dynamic mechanism is the most efficient, but not an easily implementable approach.[38]

While advertisers and advertising platforms clearly move to performance-based pricing, the user behavior is becoming more complex. Current trends show that device usage will continue to change how users consume information, enjoy leisure time, and communicate with each other (see Fulgoni and Lipsman[39] and Xu et al.[40]). In this context, the advertising ecosystem is evolving. Brands have changed ad formats to engage and interact with users in-video, in-game, and in other dynamic content. The hyper-local nature of mobile applications provides a new type of a signal about the user. This more refined user information, in turn, changes the composition of advertisers. The typical persona an advertiser has shifted from large business entities like big-name brands or e-commerce platforms on Google and Facebook, to deep vertically integrated professionals. For example, on Yelp, Zillow, or Grubhub, advertisers are small business owners or individual professionals like real estate agents or plumbers. These business users do not have the sophisticated business knowledge or technology skills to aid them in ad campaign management. The user actions relevant for these new breeds of advertisers shifted away from clicks to in-app text, direct call, or email communication. Ideally, new vertical marketplaces will provide users with better experiences and advertisers with more accurate user intent and direct access to relevant users. But, it requires advertising platforms to adapt and evolve. Advanced machine learning technology is essential for these new platforms to accurately account for user actions with a large volume of data points across devices. Another key necessity is the design of new auction mechanisms that encompass the need to provide the advertisers with simple and easy ways to manage work flows. The technological change that has been occurring over the past decade creates the need for new platforms that are both grounded in the game theory but also account for complex behavioral responses of users and advertisers.

Acknowledgments

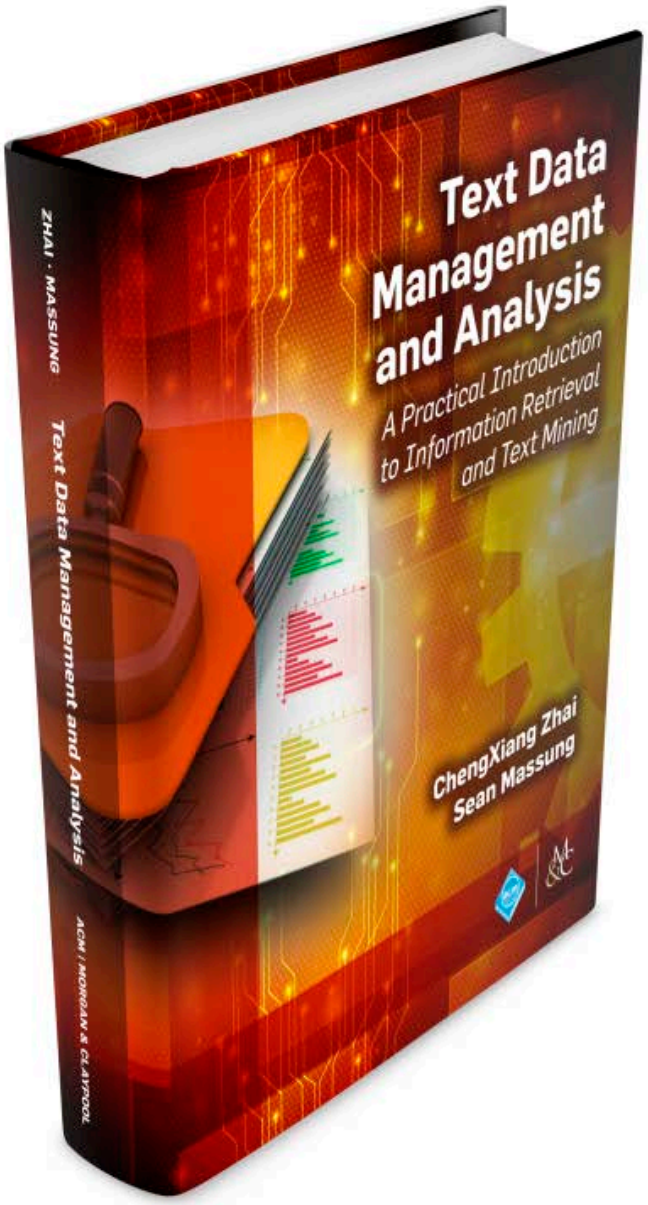
The authors are indebted to Éva Tardos who suggested the theme of this article and helped them shape it.

References

1. Aradillas-Lopez, A. Tamer, E. The identification power of equilibrium in simple games. *J. Bus. Econ. Stat. 26*, 3 (2008), 261–283.
2. Arora, S., Hazan, E., Kale, S. The multiplicative weights update method: A meta-algorithm and applications. *Theory of Computing 8*, (2012), 121–164.
3. Athey, S. Nekipelov, D. *A Structural Model of Sponsored Search Advertising Auctions*. AdAuctions'10, 2010.
4. Bajari, P., Hong, H., Nekipelov, D. Game theory and econometrics: a survey of some recent results, *Advances in Economics and Econometrics 3*, (2013), 3–52.
5. Bajari, P., Hong, H., Krainer, M., Nekipelov, D. Estimating static models of strategic interactions, *J. Bus. Econ. Stat*, (2011).
6. Beresteanu, A., Molchanov, I., Molinari, F. Sharp identification regions in models with convex moment predictions. *Econometrica 79*, (2011), 1785–1821.
7. Blake, T., Nosko, C., Tadelis, S. Customer heterogeneity and paid search effectiveness: a large scale field experiment. *Econometrica 83*, 1 (2015), 155–174.
8. Blum, A., Even-Dar, E., Ligett, K. Routing without regret: on convergence to Nash equilibria of regret-minimizing algorithms in routing games. *PODC '06*, 45–52,
9. Blum, A., Hajiaghayi, M., Ligett, K., Roth, A. Regret minimization and the price of total anarchy. *ACM STOC*, (2008), 373–382.
10. Blum, A., Mansour, Y. From external to internal regret. *J. Mach. Learn. Res. 8*, (2007), 1307–1324.
11. Blum, A. Mansour, Y. Learning, regret minimization, and equilibria. In *Algorithmic Game Theory*. N. Nisan et al. eds. Cambridge University Press, 2007, 79–101.
12. Caragiannis, I., Kaklamanis, C., Kanellopoulos, P., Kyropoulou, M., Lucier, B., Paes Leme, R., Tardos, E. Bounding the inefficiency of outcomes in generalized second price auctions. *J. Econ. Theo. 156*, (2015), 343–388.
13. Ciliberto, F., Tamer, E. Market structure and multiple equilibria in airline markets. *Econometrica 77*, 6 (2009), 1791–1828.
14. Cole, R., Roughgarden, T. The sample complexity of revenue maximization. *ACM STOC*, (2014), 243–252.
15. Cummings, R., Ligett, K., Nissim, K., Roth, A., Wu, Z. Adaptive learning with robust generalization guarantees. In *29th Annual Conference on Learning Theory*, (2016), 772–814.
16. Edelman, B., Ostrovsky, M., Schwarz, M. Internet advertising and the generalized second-price auction. *American Economic Review 97*, 1 (2007), 242–259.
17. Freund, Y., Hsu, D. A new Hedging algorithm and its application to inferring latent random variables. ArXiv:0806.4802
18. Hart, S., Mas-Colell, A. A simple adaptive procedure leading to correlated equilibrium. *Econometrica 68*, (2000), 1127–1150.
19. Hartline, J., Syrgkanis, V., Tardos, É. *No-Regret Learning in Repeated Bayesian Games, Advances in Neural Information Processing Systems* 28 (NIPS 2015).
20. Hoy, D., Syrgkanis, V., Nekipelov, D. Robust data-driven efficiency guarantees in auctions. *arXiv:1505.00437*, 2015
21. Kesselheim, T., Kleinberg, R., Tardos, E. Smooth online mechanisms: a game-theoretic problem in renewable energy markets, EC'15, 203–220, 2015.
22. Kleinberg, R., Piliouras, G., Tardos, E. Load balancing without regret in the bulletin board model. *Distributed Computing 24*, 1 (2011), 21–29.
23. Koutsoupias, E., Papadimitriou, C. Worst-case equilibria, In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science* (1999), 404–413.
24. Lahaie, S. Mcafee, P. Efficient Ranking in Sponsored Search, Internet and Network Economics (2011), 254–265.
25. Lewis, R. Rao, J. The unfavorable economics of measuring the returns to advertising, SSRN 2367103, 2014.
26. Littlestone, N. Warmuth, M. The weighted majority algorithm. *Information and Computation 108*, 2 (1994), 212–260.
27. Lykouris, T., Syrgkanis, V., Tardos, É. Learning and efficiency in games with dynamic population. SODA'16, 120–129, 2016.
28. Morgenstern, J., Roughgarden, T. On the pseudo-dimension of nearly optimal auctions. NIPS'15, 136–144, 2015.
29. Muthukrishnan, S. Ad exchanges: research issues. *Internet and Network Economics*, 1–12, Springer, 2009.
30. Aggarwal, G. Muthukrishnan, S. Theory of sponsored search auctions. foundations of computer science, 2008, FOCS'08. IEEE 49th Annual IEEE Symposium on. IEEE, 2008.
31. Muthukrishnan, S. Internet ad auctions: insights and directions. *Automata, Languages and Programming*, 14–23, Springer, 2008.
32. Myerson, R. Optimal auction design. *Mathematics of Operations Research 6*, 1 (1981), 58–73.
33. Nekipelov, D., Syrgkanis, V., Tardos, E. Econometrics for learning agents, EC '15, 1–18, 2015.
34. Ostrovsky, M., Schwarz, M. Reserve prices in internet advertising auctions: a field experiment. EC'11, 59–60, 2011.
35. Lewis, R., Rao, J., Reiley, D. *Measuring the Effects of Advertising: The Digital Frontier, Economic Analysis of the Digital Economy*. University of Chicago Press, 2015, 191–218.
36. Roughgarden, T. Intrinsic robustness of the price of anarchy, *Journal of the ACM 62*, 5 (2015), article No. 32.
37. Syrgkanis, V., Tardos, É. Composable and efficient mechanisms STOC '13, 211–220, 2013.
38. Ballings, M., Van den Poel, D. CRM in social media: predicting increases in facebook usage frequency. *Eur. J. Oper. Res. 244*, 1 (2015), 248–260.
39. Fulgoni, G., Lipsman, A. Numbers, please: digital game changers: how social media will help usher in the era of mobile and multi-platform campaign-effectiveness measurement. *J. Advert. Res. 54*,1 (2014), 11–16.
40. Xu, J., Forman, C., Kim, J.B., Van Ittersum, K. News media channels: complements or substitutes? evidence from mobile phone usage. *J. Mark. 78*, 4 (2014), 97–112.
41. Varian, H. Online ad auctions. *Am. Econ. Rev. 99*, 2 (2009), 430–434.

Denis Nekipelov (denis@virginia.edu) is an associate professor in the departments of Economics and Computer Science at the University of Virginia, Charlottesville.

Tammy Wang (twang@rivierapartners.com) is VP of Data Science and Analytics at Riviera Partners, San Francisco, CA.



The most useful and practical knowledge for building a variety of text data applications.

CS STUDENTS (Undergrad & Graduate)
LIBRARY & INFORMATION SCIENTISTS
TEXT DATA PRACTITIONERS

ChengXiang Zhai & Sean Massung (Authors)
University of Illinois at Urbana-Champaign

Text Data Management and Analysis covers the major concepts, techniques, and ideas in **information retrieval** and **text data mining**. It focuses on the practical viewpoint and **includes many hands-on exercises designed with a companion software toolkit** (i.e., MeTA) to help readers learn how to apply techniques of information retrieval and text mining to real-world text data. It also shows readers how to **experiment** with and **improve** some of the algorithms for interesting application tasks. The book can be used as a **text** for computer science undergraduates and graduates, library and information scientists, or as a **reference** for practitioners working on relevant problems in **managing and analyzing text data**.

http://books.acm.org
http://www.morganclaypoolpublishers.com/text

research highlights

DOI:10.1145/3068584

To view the accompanying paper, visit doi.acm.org/10.1145/3068608 rh

Technical Perspective
IronFleet Simplifies Proving Safety and Liveness Properties

By Fred B. Schneider

AS WITH CLAIMS about beauty, claims about system correctness are relative to expectations. Such expectations for a computing system are called *specifications*. And while claims about beauty are necessarily subjective ("in the eye of the beholder"), claims about correctness need not be—they can be conveyed as proofs in a formal logic, which can be communicated to anyone and then independently checked.

Whether a given correctness claim is useful will depend on how completely the specification characterizes a system's behaviors. Obviously, we are limited by what can be said using our specification language and what specifications can be proved using our formal logic. Over the last half-century, a consensus has emerged that specifying sets of behaviors called *trace properties* is a sweet spot. Trace properties suffice for describing most of the important aspects of a system's behaviors. Moreover, trace properties for a system are straightforward to deduce from trace properties for its components, so we can reason about trace properties compositionally.

The description of a trace property must, by definition, characterize a set of system behaviors, where whether a system behavior σ is included in that set depends only on σ and not on other system behaviors in the set or on other properties of the set. So trace properties formalize partial and total correctness, mutual exclusion, deadlock freedom, starvation freedom, among others.

Moreover, trace properties are easily specified. The text of a program specifies a trace property comprising the set of that program's possible executions, since an execution might depend on what has come before but not on what occurs in a different execution. By modeling a system behavior as a sequence of states and/or actions, an automaton can be used to specify the trace property comprising sequences the automaton accepts. And formulas of linear time Temporal Logic or its derivatives like TLA are satisfied by individual sequences, so the set of sequences that satisfy a given formula also is a trace property.

In a 1977 paper, Leslie Lamport used the terms *safety property* ("bad things" don't happen) and *liveness property* ("good things" do happen) in connection with reasoning about mutual exclusion protocols and other concurrent algorithms.

These two kinds of specifications seemed to cover the landscape. In the decade that followed, Alpern and I showed that Lamport was right. We proved that safety properties and liveness properties are basic building blocks for all trace properties; invariants suffice for verifying safety properties; variant functions are also needed for verifying liveness properties. Thus, not only are safety properties and liveness properties a natural way to specify program behaviors, but such a decomposition provides insight into the approach needed for constructing a proof.

That's the theory. The following paper represents an important step forward for the practice. It describes mechanically checked proofs for two non-trivial distributed services: A Paxos-based library to support replication and a shared key-value store. Appropriate safety properties and liveness properties are proved for each. These are not the first mechanically checked proofs for non-trivial systems software. But prior work just proved safety properties, where invariants suffice; additional techniques (as noted above) must be employed for proving liveness properties.

IronFleet is interesting because it uses refinement and reduction to simplify proof construction.

IronFleet is also interesting because it uses refinement and reduction to simplify proof construction. Refinement is a way to show that some high-level specification is satisfied by systems satisfying a low-level specification; reduction allows reasoning about coarse-grained atomic actions to support inferences about systems having fine-grained actions. These techniques are not new, but with IronFleet we see them in action on real system software. For example, we see how difficulties with reduction for proving liveness properties can be avoided by restricting the internal design of each individual step that a service may take. We thus learn about an engineering effort that, by necessity, encompasses both system construction and proof construction.

The authors also report performance impact and the expansion in code size needed to accommodate annotations required for the mechanical proof checking.

We are now closer to having the capability for software systems to be accompanied by correctness claims that are grounded in mechanically checked proofs. The need is great; the progress reported herein should be an inspiration. C

Fred B. Schneider (fbs@cs.cornell.edu) is Samuel B. Eckert Professor of Computer Science at Cornell University, Ithaca, NY.

DOI:10.1145/3068608

IronFleet: Proving Safety and Liveness of Practical Distributed Systems

By Chris Hawblitzel, Jon Howell, Manos Kapritsos, Jacob R. Lorch, Bryan Parno, Michael L. Roberts, Srinath Setty, and Brian Zill

Abstract

Distributed systems are notorious for harboring subtle bugs. Verification can, in principle, eliminate these bugs, but it has historically been difficult to apply at full-program scale, much less distributed system scale. We describe a methodology for building practical and provably correct distributed systems based on a unique blend of temporal logic of actions-style state-machine refinement and Hoare-logic verification. We demonstrate the methodology on a complex implementation of a Paxos-based replicated state machine library and a lease-based sharded key-value store. We prove that each obeys a concise safety specification as well as desirable liveness requirements. Each implementation achieves performance competitive with a reference system. With our methodology and lessons learned, we aim to raise the standard for distributed systems from "tested" to "correct."

1. INTRODUCTION

Distributed systems are notoriously hard to get right. Protocol designers struggle to reason about concurrent execution on multiple machines, which leads to subtle errors. Engineers implementing such protocols face the same subtleties and, worse, must improvise to fill in gaps between abstract protocol descriptions and practical constraints such as "real logs cannot grow without bound." Thorough testing is considered best practice, but its efficacy is limited by distributed systems' combinatorially large state spaces.

In theory, formal verification can categorically eliminate errors from distributed systems. However, due to the complexity of these systems, previous work has primarily focused on formally specifying,[1,8,18] verifying,[20] or at least bug-checking[9] distributed protocols, often in a simplified form, without extending such formal reasoning to the implementations. In principle, one can use model checking to reason about the correctness of both protocols[15] and implementations.[17] In practice, however, model checking is incomplete—the accuracy of the results depends on the accuracy of the model—and does not scale.[1]

This paper presents IronFleet, the first methodology for automated machine-checked verification of the safety and liveness of nontrivial distributed system implementations. The IronFleet methodology is practical: it supports complex, feature-rich implementations with reasonable performance, and a tolerable proof burden.

Ultimately, IronFleet guarantees that the implementation of a distributed system meets a high-level, centralized specification. For example, a sharded key-value store acts as a key-value store, and a replicated state machine acts as a state machine. This guarantee categorically rules out race conditions, violations of global invariants, integer overflow, disagreements between packet encoding and decoding, and bugs in rarely exercised code paths such as failure recovery. Moreover, it not only rules out bad behavior but also tells us exactly how the distributed system will behave at all times.

The IronFleet methodology supports proving both *safety* and *liveness* properties of distributed system implementations. A safety property says that the system cannot perform incorrect actions; for example, replicated-state-machine linearizability says that clients never see inconsistent results. A liveness property says that the system eventually performs a useful action, for example, that it responds to each client request. In large-scale deployments, ensuring liveness is critical, since a liveness bug may render the entire system unavailable.

IronFleet takes the verification of safety properties further than prior work (Section 7), mechanically verifying two full-featured systems. The verification applies not just to their protocols but to actual imperative implementations that achieve good performance. Our proofs reason all the way down to the bytes of the UDP packets sent on the network, guaranteeing correctness despite packet drops, reorderings, or duplications.

Regarding liveness, IronFleet breaks new ground: to our knowledge, IronFleet is the first system to mechanically verify liveness properties of a practical protocol, let alone an implementation.

IronFleet achieves comprehensive verification of complex distributed systems via a methodology for structuring and writing proofs about them, as well as a collection of generic verified libraries useful for implementing such systems. Structurally, IronFleet's methodology uses a *concurrency containment* strategy (Section 3) that blends two distinct verification styles within the same automated theorem-proving framework, preventing any semantic gaps between them. We use temporal logic of actions (TLA)-style state-machine refinement[13] to reason about protocol-level concurrency, ignoring implementation complexities, then use

The original version of this paper was published as "IronFleet: Proving Practical Distributed Systems Correct" in the *25th ACM Symposium on Operating Systems Principles (SOSP)*, Oct. 2015.

Floyd–Hoare-style imperative verification[5, 7] to reason about those complexities while ignoring concurrency. To simplify reasoning about concurrency, we impose a machine-checked *reduction-enabling obligation* on the implementation. Finally, we structure our protocols using *always-enabled actions* (Section 4) to greatly simplify liveness proofs.

To illustrate IronFleet's applicability, we have built and proven correct two rather different distributed systems: IronRSL, a Paxos-based[12] replicated-state-machine library, and IronKV, a sharded key-value store. All IronFleet code is publicly available.

IronRSL, our first application, has a complex implementation including many details often omitted by prior work, such as state transfer, log truncation, dynamic view-change timeouts, batching, and a reply cache. We prove full functional correctness and the key liveness property: if the network is eventually synchronous for a live quorum of replicas, then clients that persist in sending requests eventually get replies.

Unlike IronRSL, which uses distribution for reliability, IronKV uses it for improved throughput by moving "hot" keys to dedicated machines. For IronKV, we prove complete functional correctness and an important liveness property: if the network is fair then the reliable-transmission component eventually delivers each message.

While verification rules out a host of problems, it is not a panacea. IronFleet's correctness relies on several assumptions (Section 2.4). Also, verification requires more up-front development effort: the automated tools we use fill in many low-level proof steps automatically, but still require considerable assistance from the developer. Finally, we focus on verifying newly written code in Dafny, a verification-friendly language (Section 2.2), rather than verifying existing code.

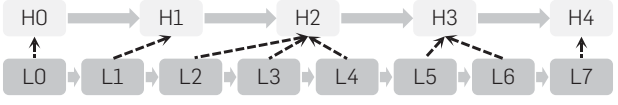
2. BACKGROUND AND ASSUMPTIONS

We briefly describe the existing verification techniques that IronFleet draws upon, as well as our assumptions.

2.1. State machine refinement

State machine refinement[11] is often used to reason about distributed systems.[1, 8, 18] The developer describes the desired system as a simple abstract state machine with potentially infinitely many states and with nondeterministic transition predicates. She then creates a series of increasingly complex (but still declarative) state machines, and proves that each one *refines* the one "above" it (Figure 1). State machine L refines H if each of L's possible *behaviors*, that is, each (potentially infinite) sequence of states the machine may visit, corresponds to an equivalent behavior of H. State machine refinement in a distributed-system context (e.g., TLA-style refinement[13]) typically considers declarative specifications, not imperative code.

Figure 1. State machine refinement. The low-level state machine behavior L0…L7 refines the high-level behavior H0…H4. Each low-level state corresponds to a high-level state; for each such correspondence, shown as a dashed line, the two states must satisfy the spec's refinement conditions. Each low-level step maps to one high-level step (e.g., L0→L1 maps to H0→H1) or no high-level steps (e.g., L2→L3).



2.2. Floyd–Hoare verification

Many program verification tools support Floyd–Hoare style[5, 7] first-order predicate logic reasoning about imperative programs. That is, the programmer annotates a program with assertions about the program's state, and the verifier checks that the assertions hold for all possible program inputs. For example, the code in Figure 2 asserts a condition about its input via a *precondition* and asserts a condition about its output via a *postcondition*.

We use Dafny,[14] a high-level language that automates verification via the Z3[2] SMT solver. This enables it to fill in many low-level proofs automatically; for example, it easily verifies the program in Figure 2 for all possible inputs x without any assistance.

However, many proposition classes are not decidable in general, so Z3 uses heuristics. For example, propositions involving universal quantifiers (∀) and existential quantifiers (∃) are undecidable. Thus, it is possible to write correct code in Dafny that the solver nevertheless cannot prove automatically. In such cases, the developer may insert annotations to guide the verifier's heuristics to a proof.

Once a program verifies, Dafny compiles it to C# and has the .NET compiler produce an executable. Other languages (e.g., C++) are currently unsupported, but it would be possible to compile Dafny to them to, for example, simplify integration with existing code. Our previous work[6] shows how to compile Dafny to verifiable assembly to avoid depending on the Dafny compiler, .NET, and Windows.

Like most verification tools, Dafny only considers one single-threaded program, not a collection of concurrently executing hosts. Indeed, some verification experts estimate that the state-of-the-art in concurrent program verification lags that of sequential verification by a decade.[19]

2.3. Temporal logic of actions (TLA)

Temporal logic and its extension TLA[11] are standard tools for reasoning about safety and liveness. Temporal logic *formulas* are predicates about the system's current and future states. The simplest type of formula ignores the future; for example, a formula P could be "host h holds the lock now." Other formulas involve the future; for example, $\Diamond P$ means P eventually holds, and $\Box P$ means P holds now and forever. Thus, $\forall h \in \text{Hosts}: \Box\Diamond P$ means that for any host, it is always true that h will eventually hold the lock.

Figure 2. Simple Floyd–Hoare verification example.

```
method halve(x:int) returns (y:int)
  requires x > 0;
  ensures  y < x;
{
  y := x / 2;
}
```

Although Dafny does not directly support the temporal logic □ and ◊ operators, Dafny's logic is powerful enough to encode □ and ◊ using universal and existential quantifiers (∀ and ∃). Section 4 describes our encoding, which is a simple library written in Dafny that does not require any extensions to the Dafny language. Thus, we do not need a separate tool for reasoning about TLA, nor do we modify Dafny; instead, we use the existing Dafny language to reason about both our executable implementation and our high-level TLA-style specifications. Using a single language avoids any semantic gaps between implementation and specification.

2.4. Assumptions

Our guarantees rely on the following assumptions.

A small amount of our code is assumed, rather than proven, correct. Thus, to trust the system, a user must read this code. Specifically, the spec for each system is trusted, as is the brief main-event loop that runs `ImplInit` and `ImplNext` (see Section 3). We do not assume reliable packet delivery, so the network may arbitrarily delay, drop, or duplicate packets. We *do* assume the network does not tamper with packets, and that addresses in packet headers are trustworthy. These integrity assumptions can be enforced within, say, a datacenter or VPN, and could be relaxed by modeling the necessary cryptographic primitives to talk about keys instead of addresses.[6]

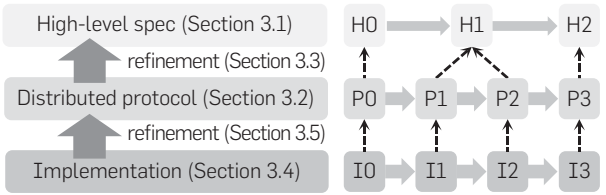
We assume the correctness of Dafny, the .NET compiler and runtime, and the underlying Windows OS. Our previous work[6] shows how to compile Dafny code into verifiable assembly code to avoid these dependencies. We also rely on the correctness of the underlying hardware.

Our liveness properties depend on further assumptions. For IronRSL, we assume a quorum of replicas run their respective main loops with a minimum frequency, never running out of memory, and the network eventually delivers messages synchronously among them. For IronKV, we assume that each host's main loop executes infinitely often and that the network is fair, that is, a message sent infinitely often is eventually delivered.

3. VERIFICATION METHODOLOGY

IronFleet organizes a distributed system's implementation and proof into layers (Figure 3), all of which are expressed in Dafny. This layering avoids the intermingling of subtle distributed protocols with implementation complexity. At the top (Section 3.1), we write a simple spec for the system's behavior. We then write an abstract distributed protocol layer (Section 3.2) and use TLA-style techniques to prove that it refines the spec layer (Section 3.3). Then we write an imperative implementation layer to run on each host (Section 3.4) and prove that, despite the complexities introduced when writing real systems code, the implementation correctly refines the protocol layer (Section 3.5). Section 4 extends this methodology to liveness properties.

Figure 3. Verification overview. IronFleet divides a distributed system into carefully chosen layers. We use TLA-style verification to prove that any behavior of the protocol layer (e.g., P0...P3) refines some behavior of the high-level spec (e.g., H0...H2). We then use Floyd–Hoare style to prove that any behavior of the implementation (e.g., I0...I3) refines a behavior of the protocol layer.



To avoid complex reasoning about interleaved execution of low-level operations at multiple hosts, we use a *concurrency containment* strategy: the proofs above assume that every implementation step performs an atomic protocol step. Since the real implementation's execution is not atomic, we use a verified reduction argument to show that a proof assuming atomicity is equally valid as a proof for the real system. This argument imposes a mechanically verified property on the implementation.

3.1. The high-level spec layer

What does it mean for a system to be *correct?* One can informally enumerate properties and hope they suffice to provide correctness. A more rigorous way is to define a *spec*, a succinct description of all allowable behaviors of the system, and prove that an implementation always generates outputs consistent with the spec.

With IronFleet, the developer writes the system's spec as a state machine expressed in Dafny (Section 2.2): starting with some initial state, the spec succinctly describes how that state can be transformed. The spec defines the state machine via three *predicates*, that is, functions that return true or false. `SpecInit` describes acceptable starting states, `SpecNext` describes acceptable ways to move from an old to a new state, and `SpecRelation` describes the required conditions on the relation between an implementation state and its corresponding spec state. For instance, in Figure 3, `SpecInit` constrains H0, `SpecNext` constrains steps such as H0→H1 and H1→H2, and `SpecRelation` constrains corresponding state pairs such as (I1, H1) and (I3, H2). To avoid unnecessary constraints on implementations of the spec, `SpecRelation` should only talk about the externally visible behavior of the implementation, for example, the set of messages it has sent so far.

As a toy example, the Dafny spec in Figure 4 describes a simple distributed lock service with a single lock that passes among the hosts. It defines the system's state as a history: a sequence of host IDs such that the *n*th host in the sequence held the lock in epoch *n*. Initially, this history contains one valid host. The system can step from an old state to a new state by appending a valid host to the history. An implementation is consistent with the spec if all lock messages for epoch *n* come from the *n*th host in the history.

By keeping the spec simple, a skeptic can study the spec to understand the system's properties. In our example, she can easily conclude that the lock is never held by more than one host. Since the spec captures all permitted system behaviors, she can later verify additional properties of the implementation just by verifying they are implied by the spec.

Figure 4. A toy lock specification.

```
datatype SpecState = SpecState(history:seq<HostId>)

predicate SpecInit(ss:SpecState) {
  |ss.history|==1 && ss.history[0] in AllHostIds()
}

predicate SpecNext(ss_old:SpecState,
                   ss_new:SpecState) {
  exists new_holder :: new_holder in AllHostIds() &&
  ss_new.history == ss_old.history + [new_holder]
}

predicate SpecRelation(is:ImplState,ss:SpecState) {
  forall p :: p in is.sentPackets && p.msg.lock? ==>
              p.src == ss.history[p.msg.epoch]
}
```

Figure 5. Simplified host state machine for a lock service.

```
datatype Host = Host(held:bool,epoch:int)

predicate HostInit(s:Host,id:HostId,held:bool) {
  s.held==held && s.epoch==0
}

predicate HostGrant(s_old:Host,s_new:Host,
                    spkt:Packet) {
    s_old.held && !s_new.held && spkt.msg.transfer?
  && spkt.msg.epoch == s_old.epoch+1
}

predicate HostAccept(s_old:Host,s_new:Host,
                     rpkt:Packet,spkt:Packet) {
    !s_old.held && s_new.held && rpkt.msg.transfer?
  && s_new.epoch == rpkt.msg.epoch == spkt.msg.epoch
  && rpkt.msg.epoch > s_old.epoch && spkt.msg.lock?
}

predicate HostNext(s_old:Host,s_new:Host,
                   rpkt:Packet,spkt:Packet) {
    HostGrant(s_old,s_new,spkt)
  || HostAccept(s_old,s_new,rpkt,spkt)
}
```

3.2. The distributed-protocol layer

At the untrusted distributed-protocol layer, the IronFleet methodology introduces the concept of independent hosts that communicate only via network messages. To manage the subtle concurrency, we keep this layer simple and abstract.

In more detail, we formally specify, in Dafny, a distributed system state machine. This state machine consists of *N* host state machines and a set of network packets. In each step of the distributed system state machine, one host's state machine takes a step, allowing it to atomically read messages from the network, update its state, and send messages to the network; our reduction argument relaxes this atomicity assumption (see full paper).

The developer must specify each host's state machine: the structure of the host's local state, how that state is initialized (`HostInit`), and how it is updated (`HostNext`). Within the protocol layer, IronFleet reduces the developer's effort in the following three ways.

First, we use a simple, abstract style for the host state and network interface; for example, the state uses unbounded mathematical integers (ignoring overflow issues), unbounded sequences of values (e.g., tracking all messages ever sent or received), and immutable types (ignoring memory management and heap aliasing). The network allows hosts to send and receive high-level, structured packets, hence excluding the challenges of marshalling and parsing from this layer.

Second, we use a declarative predicate style. In other words, `HostNext` merely describes how host state can change during each step; it gives no details about how to effect those changes, let alone how to do so with good performance.

Third, from the protocol's perspective, each of the steps defined above takes place atomically, greatly simplifying the proof that the protocol refines the spec layer (Section 3.3). In our reduction argument we connect this atomicity-assuming proof to a real execution.

Continuing our lock example, the protocol layer might define a host state machine in Dafny as in Figure 5. During the distributed system's initialization of each host via `HostInit`, exactly one host is given the lock via the `held` parameter. `HostNext` then says that a host may step from an old to a new state, given some incoming and outgoing packets, if the new state is the result of one of two *actions*, each represented by its own predicate. The two actions are giving away the lock (`HostGrant`) and receiving the lock from another host (`HostAccept`). A host may grant the lock if in the old state it holds the lock, and if in the new state it no longer holds it, and if the outbound packet (`spkt`) represents a transfer message to another host. Accepting a lock is analogous.

3.3. Connecting protocol to specification

The first major theorem we prove about each system is that the distributed protocol layer refines the high-level spec layer; that is, given a behavior of IronFleet's distributed system in which *N* hosts take atomic protocol steps defined by `HostNext`, we provide a corresponding behavior of the high-level state machine spec.

We use the standard approach to proving refinement, as illustrated in Figure 3. First, we define a *protocol abstraction function* `PAbs` that takes a state of the distributed protocol state machine and returns the corresponding state of the centralized spec. We could use a relation instead of a function, but the proof is easier with a function. Second, we prove that `PAbs` of the initial state of the distributed protocol satisfies `SpecInit`. Third, we prove that if a step of the protocol takes the state from `ps_old` to `ps_new`, then either `PAbs(ps_old) = PAbs(ps_new)` or `SpecNext(PAbs(ps_old), PAbs(ps_new))`.

The challenge of proving the protocol-to-spec theorem comes from reasoning about global properties of the distributed system. One key tool is to establish *invariants*: predicates that should hold throughout the execution of the distributed protocol. In the lock example, we might use

the invariant that the lock is either held by exactly one host or granted by one in-flight lock-transfer message. We can prove this invariant inductively by showing that every protocol step preserves it. Showing refinement of the spec is then simple.

3.4. The implementation layer

Unlike in the declarative protocol layer, in the implementation layer the developer writes single-threaded, imperative code to run on each host. This code must cope with all of the ugly practicalities we abstracted away in the protocol layer. For instance, it must handle real-world constraints on how hosts interact: since network packets must be bounded-sized byte arrays, we need to prove the correctness of our routines for marshalling high-level data structures into bytes and for parsing those bytes. We also write the implementation with performance in mind by, for example, using mutable arrays instead of immutable sequences and using `uint64`s instead of infinite-precision integers. The latter requires us to prove the system correct despite the potential for integer overflow.

Dafny does not natively support networking, so we extend the language with a trusted UDP specification that exposes `Init`, `Send`, and `Receive` methods. For example, `Send` expects an IP address and port for the destination and an array of bytes for the message body. When compiled, calls to these Dafny methods invoke the .NET UDP network stack.

The trusted network interface maintains a ghost variable (a variable used only for verification, not execution) that represents a "journal" of every `Send` and `Receive` that the implementation might make, including all of the arguments and return values. We use this journal when connecting the implementation to the protocol.

3.5. Connecting implementation to protocol

The second major theorem we prove about each IronFleet system is that the implementation layer correctly refines the protocol. To do this, we prove that even though the implementation operates on concrete local state, which uses heap-dependent, bounded representations, it is still a refinement of the protocol layer, which operates on abstract types and unbounded representations.

First, we prove that the host implementation refines the host state machine described in the protocol layer. This refinement proof is analogous to the one in Section 3.3, though simplified by the fact that each step in the implementation corresponds to exactly one step of the host state machine. We define an abstraction function `HAbs` that maps a host's implementation state to a host protocol state. As shown in Figure 6, we prove that the code `ImplInit`, which initializes the implementation state, ensures `HostInit` for the abstraction of that state. Similarly, we prove that the code `ImplNext`, which executes one host step, ensures `HostNext`. Note that `HostNext` refers to the journal of network events, thus connecting the implementation's low-level network actions to the protocol's abstract description of how the host should handle packets it sends and receives.

We then use our proof about one host implementation to prove that a distributed system comprising N host implementations, which is what we actually intend to run, refines the distributed protocol of N hosts. We use an *implementation abstraction function* `IAbs` that maps states of the distributed implementation to states of the distributed protocol. The refinement proof is largely straightforward because each step of the distributed implementation in which a host executes `ImplNext` corresponds to one step of the distributed protocol where a host takes a `HostNext` step. The difficult part is proving that the network state in the distributed system implementation refines the network state in the protocol layer. Specifically, we must prove that every send or receive of a UDP packet corresponds to a send or receive of an abstract packet. This involves proving that when host A marshals a data structure into an array of bytes and sends it to host B, B parses out the identical data structure.

The last major theorem we prove is that the distributed implementation refines the abstract centralized spec. For this, we use the abstraction functions from our two major refinement theorems, composing them to form our final abstraction function `PAbs(IAbs(·))`. The key part of this proof is establishing that the specified relation conditions hold, that is, that for all implementation states `is`, `SpecRelation (is, PAbs (IAbs (is)))` holds.

4. VERIFYING LIVENESS

Section 3 describes the high-level spec as a state machine. Such a spec says what the implementation *must not* do: it must never deviate from the state machine's behavior. However, we also often want to specify what the implementation *must* do; properties of this form are called *liveness* properties. For example, we might specify that the lock implementation eventually grants the lock to each host (Figure 7). Thus, a

Figure 6. Mandatory host event-handler loop.

```
method Main() {
  var s := ImplInit();
  assert HostInit(HAbs(s));
  while (true)
    invariant ImplInvariant(s);
  {
    ghost var journal_old := get_event_journal();
    ghost var old_s := s;
    ghost var ios_performed:seq<IoEvent>;
    s, ios_performed := ImplNext(old_s);
    assert HostNext(HAbs(old_s), HAbs(s), ios_performed);
    assert get_event_journal() ==
                journal_old + ios_performed;
    assert ReductionObligation(ios_performed);
  }
}
```

Figure 7. Desired liveness property for the lock service.

```
predicate LockBehaviorFair(b:map<int,SpecState>) {
  forall h:Host, i:int :: h in AllHostIds()&& i >= 0
    ==> exists j :: j >= i && h == last(b[j].history)
}
```

spec will typically include not just a state machine but also liveness properties.

Some researchers have proposed heuristics for detecting and quashing likely sources of liveness violations,[9] but it is better to definitively *prove* their absence. With such a proof, we do not have to reason about, for example, deadlock or livelock; such conditions and any others that can prevent the system from making progress are provably ruled out.

Liveness properties are much harder to verify than safety properties. Safety proofs need only reason about two system states at a time: if each step between two states preserves the system's safety invariants, then we can inductively conclude that all behaviors are safe. Liveness, in contrast, requires reasoning about infinite series of system states. Such reasoning creates challenges for automated theorem provers (Section 4.2), often causing the prover to time out rather than return a successful verification or a useful error message.

With IronFleet, we address these challenges by writing a library in Dafny that defines standard TLA operators and proves standard TLA rules from first principles. This library is a useful artifact for proving liveness properties of arbitrary distributed systems: its rules allow both the human developer and Dafny to operate at a high level by taking large proof steps with a single call to a lemma from the library. Finally, by structuring our protocols with *always-enabled actions*, we significantly simplify the task of proving liveness properties.

4.1. TLA library

As discussed in Section 2.3, TLA[11] is a standard mathematical formalism for reasoning about liveness. IronFleet encodes TLA in Dafny by expressing a TLA behavior, an infinite sequence of system states, as a Dafny mapping b from integers to states, where $b[0]$ is the initial state and $b[i]$ is the ith subsequent state. A liveness property is a constraint on the behavior of the state machine. For example, the Dafny code in Figure 7 says that for every host h, there is always a later time when h will hold the lock.

Our encoding hides key definitions from the prover except where truly needed, and instead provides verified lemmas that relate them to one another. For example, we represent temporal logic formulas as opaque objects (i.e., objects Dafny knows nothing about) of type `temporal`, and TLA transformations like □ as functions that convert `temporal` objects to `temporal` objects.

Of course, in some contexts we actually do need to reason about the internal meaning of □ and ◇. State-of-the-art SMT solvers, such as Z3, do not yet provide decision procedures for temporal operators like □ and ◇ directly. However, we can encode these operators using explicit quantification over steps: □ universally quantifies over all future steps, while ◇ existentially quantifies over some future step. We can then provide the SMT solver with heuristics to control these quantifiers using the solver's support for *triggers*.[3] One simple heuristic proved effective in many situations: when the solver is considering a future step j for one formula, such as $\Diamond Q$, the heuristic requests that the solver also consider j as a candidate step for other formulas starting with □ or ◇, such as $\Box P$ and $\Diamond (P \wedge Q)$. This allows the solver to automatically prove formulas like $(\Diamond Q) \wedge (\Box P) \Rightarrow \Diamond (P \wedge Q)$.

This heuristic is effective enough to automatically prove 40 fundamental TLA proof rules, that is, rules for deriving one formula from other formulas.[11] The heuristic allows us to prove complicated rules efficiently; for example, we state and prove a key rule about invariants in only 27 lines of Dafny, and a key rule about fairness in only 16 lines. Our liveness proofs then use these fundamental proof-rule lemmas to justify temporal formula transformations.

4.2. Always-enabled actions

Liveness properties depend on *fairness assumptions*, that is, assumptions that the underlying environment will enable progress. For instance, in IronRSL our liveness property depends on a quorum of participants continuing to run, and on the network delivering packets among that quorum and the client in a timely fashion. Fairness assumptions let us prove *fairness properties*: properties indicating that our protocol makes progress. An example fairness property is "Each host executes `HostGrant` infinitely often."

Lamport[13] suggests that fairness properties take the form "if action A becomes always *enabled*, that is, always possible to do, the implementation must eventually do it." However, reasoning about such properties is challenging. For instance, it is difficult to verify that an implementation's scheduler really has such a property. Also, to use such a property one must prove that A will always be enabled as long as some condition C holds, that is, that $\forall s.\ C(s) \Rightarrow \exists s' \mid A(s, s')$. Proving statements with alternating universal and existential quantifiers is notoriously challenging for automated theorem provers.

We thus adopt *always-enabled actions*; that is, we only use actions that are always possible to do. For instance, we would not use `HostGrant` from Figure 5 since it is impossible to perform without the lock. Instead, we might use "if you hold the lock, grant it to the next host; otherwise, do nothing," which can always be done. This means we can write a method that always does `HostGrant` no matter what state the host is in. Then, the fairness property "Each host executes `HostGrant` infinitely often" can be proven by showing that each host runs the method infinitely often; we accomplish this by invoking `HostGrant` inside a round-robin scheduler that itself sits inside an infinite loop.

Since our approach deviates from Lamport's standard fairness formulas, it can admit specifications that are not machine closed.[13] Machine closure ensures that liveness conditions do not combine with safety conditions to create an unimplementable spec, such as that the implementation must both grant a lock (to be fair) and not grant a lock (to be safe, because it does not hold the lock). Fortunately, machine closure is no concern in IronFleet: the existence of an implementation that meets a fairness

property is itself proof that the property does not prevent implementation!

4.3. Liveness proof strategies

Most of a liveness proof involves demonstrating that if some condition C_i holds then eventually another condition C_{i+1} holds. By chaining such proofs together, we can prove that if some assumed initial condition C_0 holds then eventually some useful condition C_n holds. For instance, in IronRSL, we prove that if a replica receives a client's request, it eventually suspects its current view; if it suspects its current view, it eventually sends a message to the potential leader of a succeeding view; and, if the potential leader receives a quorum of suspicions, it eventually starts the next view.

Most steps in this chain require an application of a variant of Lamport's WF1 rule.[11] This variant involves a starting condition C_i, an ending condition C_{i+1}, and an always-enabled action predicate Action. It states that C_i leads to C_{i+1} if the following three requirements are met:

1. If C_i holds, it continues to hold as long as C_{i+1} does not.
2. If a transition satisfying Action occurs when C_i holds, it causes C_{i+1} to hold.
3. Transitions satisfying Action occur infinitely often.

We use this in Dafny as follows. Suppose we need a lemma that shows C_i leads to C_{i+1}. We first find the action transition Action intended to cause this. We then establish each of requirements 1 and 2 with an invariant proof that considers only pairs of adjacent steps. We then establish requirement 3, a fairness property, as discussed in Section 4.2. Finally, having established the three preconditions for the WF1 lemma from our verified library, we call that lemma.

5. SYSTEM IMPLEMENTATION

We use the IronFleet methodology to implement two practical distributed systems. All IronFleet code is publicly available.

5.1. IronRSL

IronRSL replicates a deterministic application on multiple machines to make that application fault-tolerant. Such replication is commonly used for critical services, such as Chubby and Zookeeper, on which many other services depend.

IronRSL guarantees safety and liveness while supporting complex implementation features. For instance, it uses batching to amortize consensus costs, log truncation to constrain memory usage, and state transfer to let nodes recover from extended network disconnection. The spec for IronRSL is simply *linearizability*: it must generate the same outputs as a system that runs the application sequentially on a single node. Our implementation achieves linearizability via the MultiPaxos[12] consensus protocol. It is worth noting that our spec does not enforce *exactly once* semantics, as it is a matter of much debate whether linearizability implies such semantics or not. If required, exactly-once semantics can be implemented—and formally proven—at the application level. We also prove that our implementation is live: if a client repeatedly sends a request to all replicas, it eventually receives a reply. No consensus protocol can be live under arbitrary conditions,[4] so we prove liveness of IronRSL under a set of fairness assumptions about the network and nodes.

5.2. IronKV

IronKV uses distribution for a completely different purpose: to scale its throughput by dynamically sharding a key-value store across a set of nodes. The high-level spec of IronKV's state machine is concise: it is simply a map (Figure 8).

In IronKV's distributed-protocol layer, each host's state consists of a map storing a subset of the key space and a "delegation map" mapping each key to the host responsible for it. To gain throughput and to relieve hot spots, IronKV allows an administrator to delegate key ranges to other hosts. When a host receives such an order, it sends the corresponding key-value pairs to the intended recipient and updates its delegation map to reflect the new owner. If such a message is lost, the protocol layer cannot be shown to refine the high-level spec, since the corresponding key-value pairs vanish. To avoid this, we design a reliable-transmission component that requires each host to acknowledge messages it receives, track its own set of unacknowledged messages, and periodically resend them. We prove desirable safety and liveness properties of this component.

We then prove a key invariant—every key is claimed either by exactly one host or in-flight packet—that we use in conjunction with the semantics ensured by the reliable-transmission component to show that the protocol layer refines the high-level spec. Finally, we implement the protocol and prove it refines the protocol layer.

Figure 8. Complete high-level spec for IronKV state machine.

```
type Map = map<Key,Value>
type OptValue = ValuePresent(v:Value) | ValueAbsent

predicate SpecInit(h:Map) {
  h == map []
}

predicate Set(h:Map,h':Map,
              k:Key,ov:OptValue) {
  h' == if ov.ValuePresent? then h[k := ov.v]
        else map ki | ki in h && ki!=k :: h[ki]
}

predicate Get(h:Map,h':Map,
              k:Key,ov:OptValue) {
  h' == h && ov == if k in h then ValuePresent(h[k])
                   else ValueAbsent()
}

predicate SpecNext(h:Map,h':Map) {
  exists k, ov :: Set(h,h',k,ov) || Get(h,h',k,ov)
}
```

5.3. Common libraries

We wrote several libraries when building IronRSL and IronKV.

Marshalling and parsing. All distributed systems need to marshal and parse network packets, a tedious task prone to bugs. Hence, we have written and verified a generic grammar-based parser and marshaller to hide this pain from developers. For each distributed system, the developer specifies a high-level grammar for her messages. The library automatically converts byte arrays to and from a datatype conforming to the grammar.

Collection properties. We have developed a library proving many useful relationships about collections such as sequences, sets, maps, etc. These are common for reasoning about distributed systems, for example, to reason about whether a set of nodes form a quorum.

Generic refinement. We also built a library for reasoning about refinement between collections, for example, to prove the refinement from protocol-layer collections containing abstract node identifiers to implementation-layer collections containing IP addresses.

6. EVALUATION

IronFleet's premise is that automated verification is a viable engineering approach, ready for developing real distributed systems. We evaluate that hypothesis by answering the following questions: (1) How does verification affect the development of distributed systems? (2) How does the performance of a verified system compare with an unverified one?

6.1. Developer experience

To assess practicality, we evaluate the developer experience as well as the effort required to produce verified systems. The experience of producing verified software shares some similarities with that of unverified software. Dafny provides near-real-time integrated development environment feedback. Hence, as the developer writes a given method or proof, she typically sees feedback in 1–10 s indicating whether the verifier is satisfied. To ensure the entire system verifies, our build system tracks dependencies across files and outsources, in parallel, each file's verification to a cloud virtual machine. Thus, while a full integration build done serially requires 6 h, in practice, the developer rarely waits more than 6–8 min, which is comparable to a traditional large system integration build and test pass.

An IronFleet developer must write a formal trusted spec, a distributed protocol layer, and proof annotations to help the verifier see the refinements between them. Table 1 quantifies this effort by reporting the amount of proof annotation required for each layer of the system. We count all non-spec, non-executable code as proof annotation; this includes, for example, preconditions and postconditions, loop invariants, and all lemmas and invocations thereof. Our ratio of proof annotation to implementation is 7.7:1 (5.4:1 if liveness proof annotations are excluded). In total, developing the IronFleet methodology and applying it to build and verify two real systems required approximately 3.7 person-years.

Table 1. Code sizes and verifi cation times

	Spec	Impl	Proof	Time to verify (minutes)
	Source lines of code			
High-level spec	327			
Distributed protocol				
IronRSL	202	–	12,450	145
IronKV	134	–	6817	37
TLA library	–	–	1824	2
Implementation	737	5114	18,162	207
Total	1400	5114	39,253	395

In exchange for this effort, IronFleet produces a provably correct implementation with desirable liveness properties. Indeed, except for unverified components like our C# client, both IronRSL (including view changes, log truncation, etc.) as well as IronKV (including delegation and reliable delivery) worked the first time we ran them.

6.2. Performance

We run IronRSL on three replicas on three separate machines, each equipped with an Intel Xeon 2.13 GHz processor and connected over a 1 Gbps network. Our IronKV experiments use two such machines connected over a 10 Gbps network. In all our experiments the bottleneck was the CPU (not the memory, disk, or network).

IronRSL. Workload is offered by 1–256 parallel client threads, each making a serial request stream and measuring latency. As an unverified baseline, we use the MultiPaxos Go-based implementation from the EPaxos codebase.[16] For both systems, we measure with and without batching, and we use the same application state machine: it maintains a counter and it increments the counter for every client request. Figure 9 summarizes our results. We find that IronRSL's peak throughput is within 2.4× of the baseline.

IronKV. To measure the throughput of IronKV, we preload the server with 1000 keys, then run a client with 1–256 parallel threads; each thread generates a stream of Get (or Set) requests in a closed loop. As an unverified baseline, we use Redis, a popular key/value store written in C and C++, with the client-side write buffer disabled. For both systems, we use 64-bit unsigned integers as keys and byte arrays of varying sizes as values. Figure 10 summarizes our results. We find that IronKV's performance is competitive with that of Redis.

While our systems achieve respectable performance, they do not yet match that of the unverified baselines. Since verifying mutable data structures is challenging, we sometimes employ immutable data structures instead; our measurements indicate that these create significant bottlenecks. The baselines we compare against are highly optimized; we have also optimized our code, but each optimization must be proven correct rather than just implemented and tested. Hence, given a fixed time budget, IronFleet may produce fewer optimizations. IronFleet also suffers from compiling to C#, which imposes run-time

Figure 9. IronRSL's performance is competitive with an unverified MultiPaxos system. Results averaged over three trials.

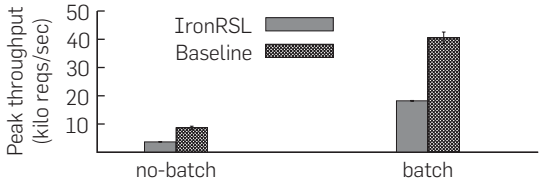
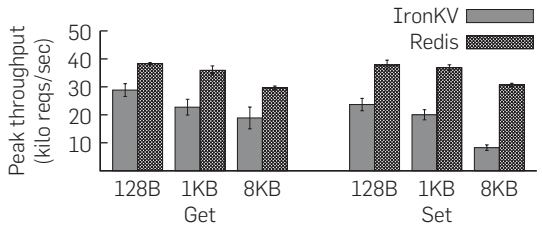


Figure 10. IronKV's performance is competitive with Redis, an unverified key-value store. Results averaged over three trials.



overhead to enforce type safety on code that provably does not need it.

7. RELATED WORK

The recent increase in the power of software verification has emboldened several research groups to use it to prove the correctness of single-machine implementations, for example, the seL4 microkernel.[10] Our Ironclad project[6] shows how to completely verify the security of sensitive services all the way down to the assembly code.

Distributed systems are known to harbor subtle design and implementation errors. Researchers have recently started generating machine-checkable proofs of correctness for their protocols, since paper proofs, no matter how formal, can contain serious errors.[25] In some cases, the proof of correctness encompasses the implementation, as well. In all cases, the systems proven correct have been much smaller and simpler than ours.

Ridge[21] proves the correctness of a persistent message queue; however, his system is substantially smaller in scale than ours and has no proven liveness properties. Schiper et al.[22] verify the correctness, but no liveness properties, of a Paxos implementation. However, they do not verify the state machine replication layer of this Paxos implementation, only the consensus algorithm, ignoring complexities such as state transfer. In contrast to IronFleet, which exploits multiple levels of abstraction and refinement, their approach posits a language below which all code generation is automatic, and above which a human can produce a one-to-one refinement. It is unclear if this approach will scale up to complex distributed systems.

Verdi[23,24] implements verified distributed systems. Its verified system transformers convert a developer's implementation in a simplified environment into an equivalent implementation that is robust in a more hostile environment, offering a clean approach to composition. Unlike IronRSL, Verdi does not prove any liveness properties and its current implementation of Raft does not support verified marshalling and parsing, state transfer, log truncation, dynamic view-change timeouts, a reply cache, or batching.

8. SUMMARY AND FUTURE WORK

The IronFleet methodology slices a system into specific layers to make verification of practical distributed system implementations feasible. The high-level spec gives the simplest description of the system's behavior. The protocol layer deals solely with distributed protocol design; we connect it to the spec using TLA+[13] style verification. At the implementation layer, the programmer reasons about a single-host program without worrying about concurrency. Reduction and refinement tie these individually feasible components into a methodology that scales to practically-sized concrete implementations. This methodology admits conventionally structured implementations capable of processing up to 18,200 requests/s (IronRSL) and 28,800 requests/s (IronKV), performance competitive with unverified reference implementations.

In the future, we plan to address two of IronFleet's limitations. First, the performance of even state-of-the-art verification tools limits the scale of the systems we can easily verify. For instance, for every system invariant, we must prove that no action can invalidate that invariant. Automated reasoning handles this with little developer burden when there are tens of actions, but likely not when there are thousands. To fix this, we will require stronger modularity, for example, to enable efficient verification that one component's actions do not interfere with another component's invariants. Another limitation of IronFleet is that it allows concurrency only among processes, not among threads that share memory. The software verification community provides a variety of approaches, such as ownership and separation logic, to address this problem. We plan to make such approaches practical in the context of automated verification of large-scale systems. C

References

1. Bolosky, W.J., Douceur, J.R., Howell, J. The Farsite project: a retrospective. *ACM SIGOPS Oper. Syst. Rev. 41*, 2 (Apr. 2007), 17–26.
2. de Moura, L.M., Bjørner, N. Z3: An efficient SMT solver. In *Proceedings of the Conference on Tools and Algorithms for the Construction and Analysis of Systems* (2008).
3. Detlefs, D., Nelson, G., Saxe, J.B. Simplify: A theorem prover for program checking. *J. ACM 52* (2003), 365–473.
4. Fischer, M.J., Lynch, N.A., Paterson, M.S. Impossibility of distributed consensus with one faulty process. *J.ACM 32*, 2 (Apr. 1985), 374–382.
5. Floyd, R. Assigning meanings to programs. In *Proceedings of Symposia in Applied Mathematics* (1967). American Mathematical Society, 19–32.
6. Hawblitzel, C., Howell, J., Lorch, J.R., Narayan, A., Parno, B., Zhang, D., Zill, B. Ironclad apps: End-to-end security via automated full-system verification. In *Proceedings of USENIX OSDI* (Oct. 2014).
7. Hoare, T. An axiomatic basis for computer programming. *Commun. ACM 12* (1969), 576–580.
8. Joshi, R., Lamport, L., Matthews, J., Tasiran, S., Tuttle, M., Yu, Y. Checking cache coherence protocols with TLA+. *J. Formal Methods Syst. Des. 22*, 2 (2003), 125–131.
9. Killian, C.E., Anderson, J.W., Braud, R., Jhala, R., Vahdat, A.M. Mace: Language support for building

distributed systems. In *Proceedings of ACM PLDI* (2007).
10. Klein, G., Andronick, J., Elphinstone, K., Murray, T., Sewell, T., Kolanski, R., Heiser, G. Comprehensive formal verification of an OS microkernel. *ACM Trans. Comput. Syst. 32*, 1 (2014), 1–70.
11. Lamport, L. The temporal logic of actions. *ACM Trans. Program. Lang. Syst. 16*, 3 (May 1994), 872–923.
12. Lamport, L.. The part-time parliament. *ACM Trans. Comput. Syst. 16*, 2 (May 1998), 133–169.
13. Lamport, L. *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers*. Addison-Wesley, Boston, MA, 2002.
14. Leino, K.R.M. Dafny: An automatic program verifier for functional correctness. In *Proceedings of the LPAR Conference* (2010).
15. Lu, T., Merz, S., Weidenbach, C., Bendisposto, J., Leuschel, M., Roggenbach, M., Margaria, T., Padberg, J., Taentzer, G., Lu, T., Merz, S., Weidenbach, C. Model checking the Pastry routing protocol. In *10th International Workshop Automated Verification of Critical Systems* (Düsseldorf, Germany, Sep. 2010).
16. Moraru, I., Andersen, D.G., Kaminsky, M. There is more consensus in egalitarian parliaments. In *Proceedings of the ACM SOSP* (2013).
17. Musuvathi, M., Park, D., Chou, A., Engler, D., Dill, D.L. CMC: A pragmatic approach to model checking real code. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation* (2002).
18. Newcombe, C., Rath, T., Zhang, F., Munteanu, B., Brooker, M., Deardeuff, M. How Amazon Web Services uses formal methods. *Commun. ACM 58*, 4 (Apr. 2015), 66–73.
19. Parkinson, M. The next 700 separation logics. In *Proceedings of IFIP VSTTE* (Aug. 2010).
20. Pek, E., Bogunovic, N. Formal verification of communication protocols in distributed systems. In *Proceedings of the Joint Conferences on Computers in Technical Systems and Intelligent Systems, MIPRO* (2003).
21. Ridge, T. Verifying distributed systems: The operational approach. In *Proceedings of the ACM POPL* (Jan. 2009).
22. Schiper, N., Rahli, V., van Renesse, R., Bickford, M., Constable, R. Developing correctly replicated databases using formal tools. In *Proceedings of IEEE/IFIP DSN* (June 2014).
23. Wilcox, J.R., Woos, D., Panchekha, P., Tatlock, Z., Wang, X., Ernst, M.D., Anderson, T. Verdi: A framework for implementing and formally verifying distributed systems. In *Proceedings of ACM PLDI* (June 2015).
24. Woos, D., Wilcox, J.R., Anton, S., Tatlock, Z., Ernst, M.D., Anderson, T. Planning for change in a formal verification of the Raft consensus protocol. In *ACM Conference on Certified Programs and Proofs (CPP)* (Jan. 2016).
25. Zave, P. Using lightweight modeling to understand Chord. *ACM SIGCOMM Comput. Comm. Rev. 42*, 2 (Apr. 2012), 49–57.

Chris Hawblitzel, Manos Kapritsos, Jacob R. Lorch, Bryan Parno, Michael L. Roberts, Srinath Setty, and Brian Zill ([chrishaw, emkaprit, lorch, parno, mirobert, srinath, bzill]@microsoft.com), Microsoft Research.

Jon Howell (jonh@jonh.net), Google.

Watch the authors discuss their work in this exclusive *Communications* video. https://cacm.acm.org/videos/ironfleet

DOI:10.1145/3068770

To view the accompanying paper, visit doi.acm.org/10.1145/3068772 rh

Technical Perspective
Building a Better Hash Function

By Michael Mitzenmacher

HASHING IS EVERYWHERE. To start, hash tables are one of the most widely used primitive data structures, with numerous variations (open address, chained, linear probing, multiple-choice, cuckoo, and so on). Hashing is also frequently used for sampling; hash all items and keep only those with certain hash values as the sample. Hashing further plays a major role in a variety of algorithms and data structures for data sketches for both streaming and non-streaming data, such as Bloom filters and approximate counting structures.

For much of the early history of hashing, there was a clear divide between theory and practice. The mathematical analysis of hashing and hashing algorithms was (and often still is) based on perfect randomness. You assume that for each input x, the hash value $h(x)$ is uniformly distributed over all possible values it could take on, and that each value $h(x)$ is independent of all other hash values $h(y)$ for $y \neq x$. Such perfect hash functions make mathematical analysis much simpler, as every new hash value looks completely random.

Of course, nobody actually uses perfect hash functions; they would take exponential space to store under any reasonable model of computation. Instead, in practice, people use various approaches to obtain pseudo-random hash functions. Knuth provides an early guide to various hashing methods of this type, such as multiplying by a constant and shifting to obtain the higher order bits.[2] Some people turn to cryptographic hash functions, although such functions may not actually be suitably random for many hashing purposes, and can be slow enough to become a bottleneck in systems that use them.

There are a large variety of real-world hash functions that come with no provable guarantees. Perhaps the biggest danger with such hash functions is that they may work deceptively well in a huge number of tests, creating a false sense of security, but they may fail miserably when faced with real data that is not random. They often seem to behave just as the analysis assuming perfect randomness predicts—that is, until they do not. An unfortunate, structured dataset can break such hash functions, in turn breaking the systems that rely on them.

Theoretical computer science has tried to develop frameworks that can provide the best of both worlds: practical hash functions along with provable guarantees. The key insight is that perfect randomness, while easier to analyze, is usually not necessary to guarantee the desired result. By taking more care in the analysis, we can often determine what we really need from our hash function, and tailor our choice of hash function to those needs. This line of work appears to have begun with the seminal work of Carter and Wegman,[1] who argued that for well-performing hash tables, hash values did not all need to be completely independent. Instead, suppose we choose a hash function randomly from a family of hash functions with range $[0, B)$ so that for any two elements x and y, the probability they end up with the same hash value is only $1/B$. This is enough to show that standard chained hash tables perform well. More generally, there are other settings where the analysis may only require we choose a hash function randomly from a family of hash functions so that any collection of k hash values are independent, for a small value of k. Fortunately, there are families of such k-independent hash functions that require only a small amount of space and computation time, each proportional to k, and these are suitable for many applications. Unfortunately, many other applications we care about appear to require logarithmic independence (or more), and for such applications the evaluation time for the hash function may become prohibitive. Still, these fundamental ideas have clarified that we can indeed obtain practical hash functions with provable guarantees in many situations, as long as we have a clear understanding of what exactly we need from our hash functions.

In the following paper, Mikkel Thorup describes another variation of simple but surprisingly effective and powerful hash functions based on using small tables of random hash values. The approach is referred to as tabular hashing. Tabular hashing actually dates back to the late 1960s, where it was used by Zobrist to create identifiers for board positions in computer games.[3] For decades, its power remained essentially unnoticed, until Thorup (and his colleagues) revived the approach. He shows how tabular hashing provides the types of general concentration guarantees often needed in practice, as well as specific guarantees for certain key algorithms and data structures, including cuckoo hashing, linear probing, and bucket-based sampling. In some cases these results arise from simple tabular hashing, but for some problems he also shows how certain improvements can provide even stronger guarantees without too much of a price in space and running time. One of the "twists" he introduces is even called twisted tabular hashing.

In short, Thorup's work has shown that tabular hashing provides great potential for more situations where we can have our cake and eat it too: that is, we can have the security of knowing our hashing offers theoretically sound guarantees, while also having the efficiency of a practical hash function that does not become a system bottleneck. C

References

1. Carter, J.L. and Wegman, M.N. Universal classes of hash functions. *J. Computer and System Sciences 18*, 2 (1979), 143–154.
2. Knuth, D.E. *The Art of Computer Programming, Volume 3: Sorting and Searching.* Addison-Wesley Publishing, Reading, MA, 1973.
3. Zobrist, A.L. A new hashing method with application for game playing. *ICCA Journal 13*, 2 (1970), 69–73.

Michael Mitzenmacher is a professor at Harvard University, School of Engineering and Applied Sciences. His work is supported in part by NSF grants NSF grants CNS-1228598, CCF-1320231, CCF-1535795, and CCF-1563710.

DOI:10.1145/3068772

Fast and Powerful Hashing Using Tabulation

By Mikkel Thorup

Abstract

Randomized algorithms are often enjoyed for their simplicity, but the hash functions employed to yield the desired probabilistic guarantees are often too complicated to be practical. Here, we survey recent results on how simple hashing schemes based on tabulation provide unexpectedly strong guarantees.

Simple tabulation hashing **dates back to Zobrist (A new hashing method with application for game playing. Technical Report 88, Computer Sciences Department, University of Wisconsin). Keys are viewed as consisting of c characters and we have precomputed character tables $h_1, \ldots, h_c$ mapping characters to random hash values. A key $x = (x_1, \ldots, x_c)$ is hashed to $h_1[x_1] \oplus h_2[x_2] \ldots \oplus h_c[x_c]$. This schemes is very fast with character tables in cache. Although simple tabulation is not even four-independent, it does provide many of the guarantees that are normally obtained via higher independence, for example, linear probing and Cuckoo hashing.**

Next, we consider ***twisted tabulation*** **where one input character is "twisted" in a simple way. The resulting hash function has powerful distributional properties: Chernoff-style tail bounds and a very small bias for minwise hashing. This is also yields an extremely fast pseudorandom number generator that is provably good for many classic randomized algorithms and data-structures.**

Finally, we consider ***double tabulation*** **where we compose two simple tabulation functions, applying one to the output of the other, and show that this yields very high independence in the classic framework of Wegman and Carter.[26] In fact, w.h.p., for a given set of size proportional to that of the space consumed, double tabulation gives fully random hashing. We also mention some more elaborate tabulation schemes getting near-optimal independence for given time and space.**

Although these tabulation schemes are all easy to implement and use, their analysis is not.

1. INTRODUCTION

A useful assumption in the design of randomized algorithms and data structures is the free availability of fully random hash functions, which can be computed in unit time. Removing this unrealistic assumption is the subject of a large body of work. To implement a hash-based algorithm, a concrete hash function has to be chosen. The space, time, and random choices made by this hash function affects the overall performance. *The generic goal is therefore to provide efficient constructions of hash functions that for important randomized algorithms yield probabilistic guarantees similar to those obtained assuming fully random hashing.*

To fully appreciate the significance of this program, we note that many randomized algorithms are very simple and popular in practice, but often they are implemented with too simple hash functions without the necessary guarantees. This may work very well in random tests, adding to their popularity, but the real world is full of structured data, for example, generated by computers, that could be bad for the hash function. This was illustrated in Ref.[21] showing how simple common inputs made linear probing fail with popular hash functions, explaining its perceived unreliability in practice. The problems disappeared when sufficiently strong hash functions were used.

In this paper, we will survey recent results from Refs.[6–9, 21, 22, 25] showing how simple realistic hashing schemes based on tabulation provide unexpectedly strong guarantees for many popular randomized algorithms, for example, linear probing, Cuckoo hashing, minwise independence, treaps, planar partitions, power-of-two-choices, Chernoff-style concentration bounds, and even high independence. The survey is from a users perspective, explaining how these tabulation schemes can be applied. While these schemes are all very simple to describe and use, the analysis showing that they work is nontrivial. For this analysis, the reader is referred to the above papers. The reader is also referred to these papers for a historical account of previous work.

1.1. Background

Generally a hash function maps a key universe $\mathcal{U}$ of keys into some range $\mathcal{R}$ of hash values. A random hash function h is a random variable from $\mathcal{R}^{\mathcal{U}}$, assigning a random hash value $h(x) \in \mathcal{R}$ to every $x \in \mathcal{U}$. A truly random hash function is picked uniformly from $\mathcal{R}^{\mathcal{U}}$, assigning a uniform and independent hash value $h(x) \in \mathcal{R}$ to each key $x \in \mathcal{U}$. Often randomized algorithms are analyzed assuming access to truly random hash functions. However, just storing a truly random hash function requires $|\mathcal{U}| \log_2 |\mathcal{R}|$ bits, which is unrealistic for large key universes.

The concept of k-independence was introduced by Wegman and Carter[26] in FOCS'79 and has been the cornerstone of our understanding of hash functions ever since. As above, we think of a hash function $h: [u] \to [m]$ as a random variable distributed over $[m]^{[u]}$. We say that h is k-independent if (a) for any distinct keys $x_1, \ldots, x_k \in [u]$, the hash values $h(x_1), \ldots, h(x_k)$ are independent random variables; and (b) for any fixed x, $h(x)$ is uniformly distributed in $[m]$.

As the concept of independence is fundamental to probabilistic analysis, k-independent hash functions are both

A previous version of this paper was published in the *Proceedings of the 36th IARCS Conference on Foundations of Software Technology and Theoretical Computer Science* (2016).

natural and powerful in algorithm analysis. They allow us to replace the heuristic assumption of truly random hash functions that are uniformly distributed in $[m]^{[u]}$, hence needing $u \lg m$ random bits ($\lg = \log_2$), with real implementable hash functions that are still "independent enough" to yield provable performance guarantees similar to those proved with true randomness. We are then left with the natural goal of understanding the independence required by hashing-based algorithms.

Once we have proved that k-independence suffices for a hashing-based randomized algorithm, we are free to use *any* k-independent hash function. The canonical construction of a k-independent hash function is based on polynomials of degree $k-1$. Let $p \geq u$ be prime. Picking random $a_0, \ldots, a_{k-1} \in \{0, \ldots, p-1\}$, the hash function is defined by:

$$h(x) = \left((a_{k-1}x^{k-1} + \cdots + a_1 x + a_0) \bmod p\right) \quad (1)$$

If we want to limit the range of hash values to $[m]$, we use $h(x) \bmod m$. This preserves requirement (a) of independence among k hash values. Requirement (b) of uniformity is close to satisfied if $p \gg m$. As suggested in Ref.,[5] for a faster implementation, we can let p be a Mersenne prime, for example, to hash 64-bit integers, we could pick $p = 2^{81} - 1$.

Sometimes two-independence suffices. For example, two-independence implies the so-called universality[5]; namely that the probability of two keys x and y colliding with $h(x) = h(y)$ is $1/m$; or close to $1/m$ if the uniformity of (b) is only approximate. Universality implies expected constant time performance of hash tables implemented with chaining. Moreover, generally, Mitzenmacher and Vadhan[15] have proved that two-independent hashing in many applications works almost like truly random hashing if the input has enough entropy. However, structured, low-entropy data, are very common in the real world.

In this paper, we shall focus on applications that require higher independence, for example, linear probing that requires five-independent hashing[18, 20] and minwise hashing that for $o(1)$ bias requires $o(1)$-independence.[12, 20] Here, when we say that k-independence is required, we mean that there are $(k-1)$-independent hash functions that do not suffice for some input. At the high end of the spectrum, when dealing with problems involving n objects, $O(\lg n)$-independence suffices in a vast majority of applications. One reason for this is the Chernoff bounds of Ref.[23] for k-independent events, whose probability bounds differ from the full-independence Chernoff bound by $2^{-\Omega(k)}$.

When it comes to high independence, we note that the polynomial method from Equation (1) takes $O(k)$ time and space for k-independence. This is no coincidence. Siegel[24] has proved that to implement k-independence with less than k memory accesses, we need a representation using $u^{1/k}$ space. He also gives a solution that for any c uses $O(u^{1/c})$ space, $c^{O(c)}$ evaluation time, and achieves $u^{\Omega(1/c^2)}$ independence (which is superlogarithmic, at least asymptotically, for $c = O(1)$). The construction is nonuniform, assuming a certain small expander which gets used in a graph product. Siegel[24] states about his scheme that it is "far too slow for any practical application."

This paper surveys a family of "tabulation"-based hash function that like Siegel's hash function use $O(u^{1/c})$ space. Their query time is only $O(c)$ and they are both simple and practical. Despite having low independence, they offer strong probabilistic guarantees for many popular randomized algorithms. We start with the simplest and fastest tabulation scheme, and move later to more complicated schemes with stronger probabilistic guarantees.

2. SIMPLE TABULATION

The first scheme we consider is *simple tabulation* hashing where the hash values are r-bit numbers. Our goal is to hash keys from $\mathcal{U} = [u]$ into the range $\mathcal{R} = [2^r]$. In tabulation hashing, a key $x \in [u]$ is interpreted as a vector of $c > 1$ *characters* from the alphabet $\Sigma = [u^{1/c}]$, that is, $x = (x_0, \ldots, x_{c-1}) \in \Sigma^c$. As a slight abuse of notation, we shall sometimes use Σ instead of $|\Sigma|$ to denote the size of the alphabet when the context makes this meaning clear. This matches the classic recursive set-theoretic definition of a natural as the set of smaller naturals.

For "simple tabulation hashing" we initialize independent random character tables $h_0, \ldots, h_{c-1} : \Sigma \to \mathcal{R}$. The hash $h(x)$ of a key $x = (x_0, \ldots, x_{c-1})$ is computed as:

$$h(x) = \bigoplus_{i \in [c]} h_i[x_i]. \quad (2)$$

Here $\oplus$ denotes bit-wise exclusive-or. This is a well-known scheme dating back at least to Zobrist.[27] For him a character position corresponds to a position on a game board, and the character is the piece at the position. If the piece at a position i changes from x_i to x_i', he updates the overall hash value h to $h' = h \oplus h_i[x_i] \oplus h_i[x_i']$.

It is easy to see that simple tabulation is three-independent, for if we have a set X of two or three keys, then there must be a position $i \in [c]$ where one key x has a character x_i not shared with any other key in X. This means that x is the only key in X whose hash value depends on $h_i[x_i]$, so the hash value of x is independent of the other hash values from X. However, simple tabulation is not four-independent. Given four keys $(a_0, b_0), (a_1, b_0), (a_0, b_1), (a_1, b_1)$, no matter how we fill our tables, we have

$$h(a_0, b_0) \oplus h(a_1, b_0) \oplus h(a_0, b_1) \oplus h(a_1, b_1) = 0.$$

Thus, given the hash values of any three of the keys, we can uniquely determine the fourth hash value.

In our context, we assume that the number $c = O(1)$ of character positions is constant, and that character tables fit in fast cache. Justifying this assumption, note that if we have n keys from a very large universe, we can first do a universe reduction. Expecting no collisions, we apply a universal hash function,[5] mapping the original large keys into an intermediate polynomial universe, say of size $u = n^3$. Simple tabulation is only applied to the intermediate universe $[u]$. To get space $O(n^\varepsilon)$, we just set $c = 3/\varepsilon$. We shall refer to the lookups in the h_i as "character lookups," emphasizing that they are expected to be much faster than a general lookup in a table of size n.

Putting things into a practical perspective (this paper claims practical schemes with strong theoretical guarantees),

in the experiments from Ref.,[21] for 32-bit keys, simple tabulation with four character lookups took less than 5 ns, whereas a single memory lookup in a 4MB table took more than 120 ns. Character lookups were thus about 100× faster than general lookups. In fact, simple tabulation is three-independent and in experiments from Ref.,[21] it was found to be more than three times faster than three-independent hashing implemented as in Equation (1) by a degree 2 polynomial tuned over the Mersenne prime field $\mathbb{Z}_{2^{61}-1}$. Because cache is so critical to computation, most computers are configured with a very fast cache, and this is unlikely to change.

Usually, it is not a problem to fill the character tables $h_0, \ldots, h_{c-1}$ with random numbers, for example, downloading them from http://random.org which is based on atmospheric noise. However, for the theory presented here, it would suffice to fill them with a (lg u)-independent pseudo-random number generator (PRG). The character tables just need to point to an area in memory with random bits, and this could be shared across many applications. One could even imagine computers configured with random bits in some very fast read-only memory allowing parallel access from multiple cores.

In Ref.,[21] simple tabulation hashing was proved to have much more power than suggested by its three-independence. This included fourth moment bounds, minwise hashing, random graph properties necessary in cuckoo hashing,[19] and Chernoff bounds for distributing balls into *many* bins. The details are described in the following subsections.

2.1. Concentration bounds

First, we consider using simple tabulation hashing to distribute n balls into $m = 2^r$ bins, that is, assuming that the balls have keys from $[u]$, we are using a simple tabulation hash function $h : [u] \to [m]$. In a hash table with chaining, the balls in a bin would be stored in a linked list.

Consider the number X of balls landing in a given bin. We have $\mu = \mathrm{E}[X] = n/m$. Pătraşcu and Thorup[21] have proved that, w.h.p., we get a Chernoff-style concentration on X. First recall the classic Chernoff bounds[16, section 4] for full randomness. On the upper bound side, we have[16, Theorem 4.1]

$$\Pr\left[X \geq (1+\delta)\mu\right] \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu} \left[\leq e^{-\delta^2\mu/3} \text{ for } \delta \leq 1\right] \quad (3)$$

The corresponding probabilistic lower bound[16, Proof of Theorem 4.2] for $\delta \leq 1$ is

$$\Pr\left[X \leq (1-\delta)\mu\right] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\right)^{\mu} \left[\leq e^{-\delta^2\mu/2} \text{ for } \delta \leq 1\right] \quad (4)$$

We note that in connection with hash tables, we are often not just interested in a given bin, but rather we care about the bin that a specific query ball lands in. This is why the hash of the query ball is involved in the theorem below with Chernoff-style bounds for simple tabulation.

THEOREM 1 (Pătraşcu and Thorup[21]). *Consider hashing n balls into $m \geq n^{1-1/(2c)}$ bins by simple tabulation (recall that $c = O(1)$ is the number of characters). Define X as the number of regular balls that hash into a given bin or a bin chosen as a function of the bin $h(q)$ of an additional* query *ball q. Let $\mu = \mathrm{E}[X] = \frac{n}{m}$. The following probability bounds hold for any constant γ:*

$$\Pr\left[X \geq (1+\delta)\mu\right] \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\Omega(\mu)} + 1/m^{\gamma} \quad (5)$$

$$\Pr\left[X \leq (1-\delta)\mu\right] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\right)^{\Omega(\mu)} + 1/m^{\gamma}. \quad (6)$$

With $m \leq n$ bins (incl. $m < n^{1-1/(2c)}$), every bin gets

$$n/m \pm O(\sqrt{n/m}\log^{c} n). \quad (7)$$

balls with probability $1 - n^{-\gamma}$.

Contrasting the standard Chernoff bounds, we see that Equations (5) and (6) in Theorem 1 can only provide polynomially small probability, that is, at least $m^{-\gamma}$ for any desired constant γ. This corresponds to if we had $\Theta(\log m)$-independence in the Chernoff bound from Ref.[23] In addition, the exponential dependence on μ is reduced by a constant which depends (exponentially) on the constants γ and c.

The upper bound (5) implies that any given bin has $O(\lg n/\lg\lg n)$ balls w.h.p., but then this holds for all m bins w.h.p. Simple tabulation is the simplest and fastest constant time hash function to achieve this fundamental property.

Complementing the above Chernoff-style bound, Dahlgaard et al.[7] have proved that we also get the kth moment bounds normally associated with k-independence.

THEOREM 2 (Dahlgaard et al.[7]). *With the same setup for simple tabulation as in Theorem 1, for any constant $k = O(1)$,*

$$\mathrm{E}\left[(X-\mu)^k\right] = O\left(\mu + \mu^{k/2}\right).$$

Since kth moment bounds is one of the main ways k-independence is used, it is nice that they are achieved by simple tabulation which is only three-independent. As an example, the classic AMS sketches[1] were first implemented with four-independent hashing, but Braverman et al.[3] noticed that only a fourth moment bound is needed. They proved that simple tabulation suffices by proving Theorem 2 for $k = 4$.

2.2. Linear probing

Theorem 1 is used in Ref.[21] to get bounds for linear probing. Linear probing is a classic implementation of hash tables. It uses a hash function h to map a set of n keys into an array of size m. When inserting x, if the desired location $h(x) \in [m]$ is already occupied, the algorithm scans $h(x)+1, h(x)+2, \ldots, m-1, 0, 1, \ldots$ until an empty location is found, and places x there. The query algorithm starts at $h(x)$ and scans until it either finds x, or runs into an empty position, which certifies that x is not in the hash table. When the query search is unsuccessful, that is, when x is not stored, the query algorithm scans exactly the same locations as an insert of x. A general bound on the query time is hence also a bound on the insertion time.

This classic data structure is one of the oldest and most popular implementations of hash tables, due to its

unmatched simplicity and efficiency. On modern architectures, access to memory is done in cache lines (of much more than a word), so inspecting a few consecutive values typically translates into a single memory access. Even if the scan straddles a cache line, the behavior will still be better than a second random memory access on architectures with prefetching.

Linear probing was shown to take expected constant time for any operation in 1963 by Knuth,[13] in a report which is now regarded as the birth of algorithm analysis. This analysis, however, assumed a truly random hash function. However, Pagh et al.[18] showed that just five-independence suffices for this expected constant operation time. In Ref.,[20] five-independence was proved necessary with a concrete combination of keys and a four-independent random hash function where searching certain keys takes $\Omega(\log n)$ expected time.

In Ref.,[21] the result from Ref.[18] is strengthened for more filled linear probing tables, showing that if the table size is $m = (1 + \varepsilon)n$, then the expected time per operation is $O(1/\varepsilon^2)$, which asymptotically matches the bound of Knuth[13] with truly random hashing. More important for this paper, Pătraşcu and Thorup[21] proved that this performance bound also holds with simple tabulation hashing.

In fact, for simple tabulation, we get quite strong concentration results for the time per operation, for example, constant variance for constant ε. For contrast, with five-independent hashing, the variance is only known to be $O(\log n)$.[18]

Experiments are done in Ref.[21] comparing simple tabulation with standard two-independent hashing schemes in linear probing. For simple inputs such as consecutive integers, the performance was extremely unreliable with the two-independent hashing, but with simple tabulation, everything worked perfectly as expected from the theoretical guarantees.

2.3. Cuckoo hashing

In cuckoo hashing,[19] we use two tables of size $m \geq (1 + \varepsilon)n$ and independent hash functions h_0 and h_1 mapping the keys to these two tables. Cuckoo hashing succeeds if we can place every key in one of its two hash locations without any collision. We can think of this as a bipartite graph with a set for each table and an edge $(h_0(x), h_1(x))$ for each key x. Cuckoo hashing fails exactly if this graph has a component with more edges than vertices. With truly random hashing, this bad event happens with probability $\Theta(\frac{1}{n})$. Pătraşcu and Thorup[21] study the random graphs induced by simple tabulation, and obtain a rather unintuitive result: the worst failure probability is inversely proportional to the *cube root* of the set size.

THEOREM 3 (Pătraşcu and Thorup[21]). *Any set of n keys can be placed in two tables of size $m = (1 + \varepsilon)$ by cuckoo hashing and simple tabulation with probability $1 - O(n^{-1/3})$. There exist sets on which the failure probability is $\Omega(n^{-1/3})$.*

Thus, cuckoo hashing with simple tabulation is an excellent construction for a static dictionary. The dictionary can be built (in linear time) after trying $O(1)$ independent hash functions w.h.p., and later every query runs in constant worst-case time with exactly two probes. We note that even though cuckoo hashing requires two independent hash functions, these essentially come for the cost of one in simple tabulation: the pair of hash codes can be stored consecutively, in the same cache line so that we can look up both for the price of one.

In the dynamic case, Theorem 3 implies that we expect $\Omega(n^{4/3})$ updates between failures requiring a complete rehash with new hash functions.

2.4. ε-Minwise independence

A hash function $h : [u] \to [m]$ is ε-minwise independent, or minwise with bias ε, if for any set $S \subseteq [u]$ and $q \in S$, $\Pr\left[h(q) = \min h(S)\right] = \frac{1 \pm \varepsilon}{n}$. The classic application of ε-minwise hashing of Broder et al.[4] is the estimation of Jaccard set similarity $|A \cap B|/|A \cup B|$. More precisely, ignoring the probability of collisions in the minimum hash value, we get

$$\begin{aligned}\Pr\left[\min h(A) = \min h(B)\right] &= \sum_{x \in A \cap B} \Pr\left[h(x) = \min h(A \cup B)\right] \\ &= (1 \pm \varepsilon)|A \cap B|/|A \cup B|.\end{aligned}$$

For better concentration on the estimate, we would make multiple experiments with independent hash functions, yet this cannot eliminate the bias ε.

To get minwise bias ε, we generally need a $\Theta(\log 1/\varepsilon)$-independent hash function.[12, 20] However, Pătraşcu and Thorup[21] shows

THEOREM 4 (Pătraşcu and Thorup[21]). *Consider a set $S \subseteq \Sigma^c$ of $n = |S|$ keys and $q \in S$. If $h : \Sigma^c \to [m]$, $m \geq n^{1+1/c}$, is implemented by simple tabulation, then* $\Pr[h(q) = \min h(S)] = (1 \pm \tilde{O}(1/n^{1/c}))/n$.

2.5. The power of two choices

The power of two choices is a standard scheme for placing balls into bins where each ball hashes to two bins, and is placed in the lightest loaded one. When placing n balls into n bins, using the two-choice paradigm with truly random hash functions, the maximum load of any bin is $\lg \lg n + O(1)$ w.h.p.[2] Dahlgaard et al.[8] have proved that simple tabulation gives a maximum load which is $\lg \lg n + O(1)$ in expectation and $O(\log \log n)$ w.h.p. This is the simplest constant time hashing scheme known to offer such strong two-choice load balancing.

2.6. Weakness with small numbers

As described above, simple tabulation has much more power than suggested by its three-independence. However, there are also some weaknesses. For example, in the Chernoff-style bounds (5) and (6) from Theorem 1, we have an additive error probability of $1/m^\gamma$ when hashing into m bins. Here γ is an arbitrarily large constant, so this is fine when m is large. However, this is not good if m is small, for example, $m = 2$ as when we toss a coin. A related problem from Theorem 4 is that the minwise bias $\tilde{O}(1/n^{1/c})$ depends on the size n of the set considered. This is fine if the set is large, but not if the set

is small. Both of these problems and more will be addressed by twisted tabulation described below.

3. TWISTED TABULATION

We will now consider *twisted tabulation* proposed by Pătraşcu and Thorup.[22] It adds a quick twist to simple tabulation, leading to more general distributional properties, including Chernoff bounds that also work for few bins and better minwise hashing that also works well for small sets. For $i = 1, \ldots, c-1$, we expand the entries of h_i with a random character called the *twister*. More precisely, for $i > 0$, we now have random tables $h_i^\star : \Sigma \to \Sigma \times \mathcal{R}$. The table $h_0 : \Sigma \to \mathcal{R}$ is kept unchanged. The hash function is now computed in two steps:

$$(t, h_{>0}) = \bigoplus_{i=1}^{c-1} h_i^\star[x_i]; \qquad h(x) = h_{>0} \oplus h_0[x_0 \oplus t]. \tag{8}$$

Figure 1 contains the C-code for simple and twisted tabulation. The twister adds lg Σ bits to each entry of the tables h_i, $i > 0$, but in practice, we want entries to have bit lengths such as 32 or 64, so for 32-bit hash codes as in Figure 1, we double the length. However, for a random 32-bit floating point in (0; 1), we only need 23 bits for a random mantissa, and then we have free bits for an 8-bit twister.

The highlight of twisted tabulation is its minimalistic nature, adding very little to the cost of simple tabulation while gaining significantly stronger guarantees. Twisted tabulation uses exactly c character lookups into tables with Σ entries, just like simple tabulation, though with larger entries. Essentially twisted tabulation only differs from simple tabulation by two AC^0 operations, so we would expect it to be almost as fast as simple tabulation (whose practicality has long been established). This was confirmed experimentally in Ref.,[22] where twisted tabulation was less than 30% slower than simple tabulation.

Figure 1. C-code for simple and twisted tabulation for 32-bit keys to 32-bit hash codes assuming a pointer H to some randomly filled storage (4KB for simple and 8KB for twisted).

```
#include <stdint.h>
//defines uintX_t as unsigned X-bit integer.

uint32_t SimpleTab32(uint32_t x, uint32_t[4][256] H) {
  uint32_t i;
  uint32_t h=0;
  uint8_t c;
  for (i=0;i<4;i++) {
    c=x;
    h^=H[i][c];
    x = x >> 8;
  }
  return h;
}

uint32_t TwistedTab32(uint32_t x, uint64_t[4][256] H) {
  uint32_t i;
  uint64_t h=0;
  uint8_t  c;
  for (i=0;i<3;i++) {
    c=x;
    h^=H[i][c];
    x = x >> 8;
  }
  c=x^h;          // extra xor compared with simple
  h^=H[i][c];
  h>>=32;         // extra shift compared with simple
  return ((uint32_t) h);
}
```

When we discuss properties of twisted tabulation, we view keys $x = (x_0, \ldots, x_{c-1})$ as composed of a head, $\mathsf{head}(x) = x_0$ and a tail, $\mathsf{tail}(x) = (x_1, \ldots, x_{c-1})$. We refer to the following implementation of twisted tabulation which is less efficient but mathematically equivalent to (8):

1. Pick a simple tabulation hash function $h^\tau : \Sigma^{c-1} \to \Sigma$ from $c-1$ characters to 1 character. This corresponds to the twister components of $h_1^\star, \ldots, h_{c-1}^\star$. Applying h^τ to the tail of a key x, we get the combined twister $t = h^\tau(\mathsf{tail}(x))$, the "twisted head" $x_0 \oplus t$, and the "twisted key" $h^\mathcal{T}(x) = (x_0 \oplus t, x_1, \ldots, x_{c-1})$.
2. Pick a simple tabulation hash function $h^\mathcal{S} : \Sigma^c \to \mathcal{R}$ (where $\mathcal{R}$ was the desired output range). This corresponds to h_0 and for $i > 0$, the nontwister component h_i of $h_i^\star$ (remember that tails are not touched by twisting). The twisted tabulation hash function is then $x \mapsto h^\mathcal{S}(h^\mathcal{T}(x))$.

For all the results presented here, it does not matter which character we view as the head. Above it is the first character, but sometimes it is more efficient if it is the last least significant character.

As noted in Ref.,[22] the twisting by $h^\mathcal{T}$ can be seen as a single-round Feistel permutation where the hash function is simple tabulation. Because twisting is a permutation, twisted tabulation inherit from simple tabulation any (probabilistic) property that holds regardless of concrete key values, for example, the above Cuckoo hashing and the power of two choices. Like simple tabulation, twisted tabulation is only three-independent, but it does have some more stronger more general distributional guarantees, which we explain in detail below.

3.1. Chernoff bounds

Chernoff bounds play a prominent role in the design of randomized algorithms[16, section 4]. The Chernoff-style bounds from Theorem 1 where limited in that they only really worked for throwing balls into a large number m of bins. Pătraşcu and Thorup[22] prove the following far more general Chernoff-style bounds for twisted tabulation.

THEOREM 5 (Pătraşcu and Thorup[22]). *Choose a random c-character twisted tabulation hash function $h = h^\mathcal{S} \circ h^\mathcal{T} : [u] \to [u]$, $[u] = \Sigma^c$. For each key $x \in [u]$ in the universe, we have an arbitrary "value function" $v_x : [u] \to [0, 1]$ assigning a value $V_x = v_x(h(x)) \in [0, 1]$ to x for each hash value. Define $V = \sum_{x \in [u]} V_x$ and $\mu = \sum_{x \in [u]} \mu_x$ where $\mu_x = \mathrm{E}[v_x(h(x))]$. Let γ, c, and $\varepsilon > 0$ be constants. Then for any $\mu < \Sigma^{1-\varepsilon}$ and $\delta > 0$, we have:*

$$\Pr[V \geq (1+\delta)\mu] \leq \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^{\Omega(\mu)} + 1/u^\gamma \tag{9}$$

$$\Pr[V \leq (1-\delta)\mu] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\right)^{\Omega(\mu)} + 1/u^\gamma. \tag{10}$$

Moreover, for any $\mu \geq \sqrt{\Sigma}$ *(including* $\mu \geq \Sigma^{1-\varepsilon}$*):*

$$V = \mu \pm \tilde{O}\left(\sqrt{\mu}\right). \tag{11}$$

If we have a given distinguished query key $q \in [u]$*, the above bounds hold even if we condition everything on* $h(q) = a$ *for any given* $a \in [u]$[a].

The statement of Theorem 5 may seem a bit cryptic, so before proceeding, we will show that it improves the simple tabulation bounds from Theorem 1. Those bounds considered a set S of n balls or keys mapped into m bins, and had an error bound of $m^{-\gamma}$. The error is here improved to $u^{-\gamma}$. This is important when m is small, for example, if $m = 2$ corresponding to unbiased coin tosses.

We now do the translation. Discounting all irrelevant keys $x \in [u]\backslash S$, we zero their value functions, setting $v_x(\cdot) = 0$. Also, we define the bin hash function from Theorem 1 as $h'(x) = h(x) \bmod m$, noting that $m \uparrow u$ since both are powers of two. Theorem 1 studies the number of balls landing a bin b which may be a function of the bin $h'(q)$ of a query ball $q \in [u]\backslash S$. Thanks to the last statement of Theorem 5, we can condition on any value a of $h(q)$, which determines $h'(q)$ and hence b. Now, for $x \in S$, define $V_x = v_x(h(x)) = 1$ if $h'(x) = b$; 0 otherwise. Now, $V = \Sigma_{x\in[u]} V_x$ is the variable X from Theorem 1, and (5) and (6) follow from (9) and (10), but with the improved error $u^{-\gamma}$.

A different illustration of the versatility of Theorem 5 is if we want each key $x \in [u]$ to be sampled with some individual sampling probability p_x. In this case, we have no distinguished query key, and we can just define $v_x(y) = 1$ if $y < p_x \cdot u$; 0 otherwise. Since $h(x)$ is uniform in $[u]$, we have that x is sampled with $V_x = v_x(h(x)) = 1$ with probability $\tilde{p}_x = \lceil p_x m \rceil / m$. The number $V = \Sigma_{x\in[u]} V_x$ of samples is now concentrated according to (9) and (10).

3.2. Minwise independence

Concerning minwise hashing, Dahlgaard and Thorup[9] have proved that twisted tabulation yields the following strengthening of Theorem 4 for simple tabulation.

THEOREM 6 (Dahlgaard and Thorup[9]). *Consider a set* $S \subseteq \Sigma^c$ *of* $n = |S|$ *keys and* $q \in S$*. If* $h : \Sigma^c \to [m]$*,* $m \geq nu^{1/c}$*, is implemented with twisted tabulation, then* $\Pr[h(q) = \min h(S)] = (1 + \tilde{O}(1/u^{1/c}))/n$.

The important difference is that the bias $\tilde{O}(1/n^{1/c})$ from Theorem 4 is replaced by $\tilde{O}(1/u^{1/c})$ which is small regardless of the set size. Such an absolutely small bias generally requires $\Omega(\log u)$-independence.[20]

3.3. Short range amortization for hash tables

We now switch to a quite different illustration of the power of twisted tabulation hashing from Ref.[22] Consider linear probing in a half-full hash table with n keys. Out of $\sqrt{n}$ operations, we expect some to take $\Omega(\log n)$ time. Nevertheless, we show that any window of $\log n$ operations on distinct keys is executed in $O(\lg n)$ time with high probability. This also holds for the simpler case of chaining.

The general point is that for any set of stored keys and any set of window keys, the operation times within the window are sufficiently independent that the average concentrates nicely around the expected constant operation time. Such concentration of the average should not be taken for granted with real hash functions. In Ref.[20] are input examples for linear probing with fast two-independent hashing such that if one operation is slow, then most operations are slow. Ref.[22] presented a parallel universe construction causing similar problems for simple tabulation. As stated above, twisted tabulation does, however, provide sufficient independence, and we expect this to prove useful in other applications.

3.4. Pseudorandom numbers generators

Like *any* hash function, twisted tabulation naturally implies a PRG with the pseudorandom sequence $h(0)$, $h(1)$, … For maximal efficiency, we use the last and least significant character as the head. Thus, a key $x = (x_{c-1}, \ldots, x_0) \in \Sigma^c$ has $\mathsf{head}(x) = x_0$ and $\mathsf{tail}(x) = x_{>0} = (x_{c-1}, \ldots, x_1)$. For twisted tabulation, we use a simple tabulation function $h^\star : \Sigma^{c-1} \to \Sigma \times [2^r]$ and a character function $h_0 : \Sigma \to [2^r]$, and then $h(x)$ is computed, setting $(t, h_{>0}) = h^\star(x_{>0})$ and $h(x) = h_0[x_0 \oplus t] \oplus h_{>0}$. We now think of x as the pair $(x_{>0}, x_0) \in \Sigma^{c-1} \times \Sigma$. As we increase the index $x = 0, 1, \ldots, \Sigma - 1, \Sigma, \Sigma + 1, \ldots = (0, 0), (0, 1), \ldots, (0, \Sigma - 1)$, (1, 0), (1, 1), …, the tail $x_{>0}$ only increases when x_0 wraps around to 0—once in every Σ calls. We, therefore, store $(t, h_{>0}) = h^\star(x_{>0})$ in a register, recomputing it only when $x_{>0}$ increases. Otherwise, we compute $h(x) = h_0[x_0 \oplus t] \oplus h_{>0}$ using just one character lookup and two $\oplus$-operations. In Ref.[22] this was found to be exceedingly fast: as fast as a single multiplication and four times faster than the standard random number generator `random()` from the GNU C library which has almost no probabilistic guarantees. Besides being faster, the twisted PRG offers the powerful distributional guarantees discussed above.

As an alternative implementation, we note that $h^\star$ is itself applied to consecutive numbers $x_{>0} = 0, 1, 2, \ldots$, so $h^\star$ can also be implemented as a PRG. The $h^\star$-PRG is only applied once for every Σ numbers generated by h, so the $h^\star$-PRG can be much slower without affecting the overall performance. Instead of implementing $h^\star$ by simple tabulation, we could implement it with any logarithmically independent PRG, thus not storing any tables for $h^\star$, but instead generating each new value $h^\star(x_{>0})$ on the y as $x_{>0}$ increases. We can view this as a general conversion of a comparatively slow but powerful PRG into an extremely fast one preserving the many probabilistic properties of twisted tabulation.

3.5. Randomized algorithms and data structures

When using the twisted PRG in randomized algorithms,[16] we get the obvious advantage of the Chernoff-style bounds from Theorem 5 which is one of the basic techniques needed.[16, Section 4] The ε-minwise hashing from Theorem 6 with $\varepsilon = \tilde{O}(1/u^{1/c})$ is important in contexts where we want to

[a] The last statement conditioning on $h(q) = a$ was not proved in Ref.,[22] but is an easy extension using ideas from Ref.[21]

assign randomized priorities to items. A direct example is treaps.[16, Section 8.2] The analysis of the expected operation time is based on each key in an interval having the same chance of getting the lowest priority. Assigning the priorities with an ε-minwise hash function, expected cost is only increased by a factor $(1+\varepsilon)$ compared with the unrealistic case of true randomness. In static settings, we can also use this to generate a random permutation, sorting items according to priorities. This sorting itself takes linear time since we essentially only need to look at the $\log n$ most significant bits to sort n priorities. Using this order to pick items, we get that classic algorithms like QuickSort[16, Section 1] and Binary Planar Partitions[16, Section 1.3] perform within an expected factor $(1+\varepsilon)$ of what they would with true randomness. With $\varepsilon = \tilde{O}(1/u^{1/c})$ as with our twisted PRG, this is very close the expected performance with true randomness.

4. DOUBLE TABULATION AND HIGH INDEPENDENCE

Thorup[25] has shown that simple tabulation can also be used to get high independence if we apply it twice. More precisely, we consider having two independent simple tabulation functions $h_0 : \Sigma^c \to \Sigma^d$ and $h_1 : \Sigma^d \to [2^r]$, and then the claim is that $h_1 \circ h_0$ is likely to be highly independent. The main point from Ref.[25] is that the first simple tabulation h_0 is likely to have an expander-type property.

More precisely, given a function $f : [u] \to \Sigma^d$, a key set $X \subseteq [u]$ has a *unique output character* if there is a key $x \in X$ and a $j \in [d]$ and such that for all $y \in X \backslash \{x\}$, $h(y)_j \neq h(x)_j$, that is, the jth output character is unique to some x in X. We say that f is *k-unique* if each nonempty key set $Y \subseteq [u]$ of size at most k has a unique output character. Siegel[24] noted that if f is k-unique and $h_1 : \Sigma^d \to [2^r]$ is a random simple tabulation function, then $h_1 \circ f : [u] \to [2^r]$ is k-independent. The main technical result from Ref.[25] is

THEOREM 7 (Thorup[25]). *Consider a random simple tabulation function $h_0 : \Sigma^c \to \Sigma^d$. Assume $c = \Sigma^{o(1)}$ and $(c+d)c = \Sigma^{o(1)}$. Let $k = \Sigma^{1/(5c)}$. With probability $1 - o(\Sigma^{2-d/(2c)})$, the function h_0 is k-unique. More concretely for 32-bit keys with 16-bit characters, h_0 is 100-unique with probability $1 - 1.5 \times 10^{-42}$.*

Assuming that h_0 is k-unique, if $h_1 : \Sigma^d \to [2^r]$ is a random simple tabulation function, then $h_1 \circ h_0$ is k-independent.

This construction for highly independent hashing is much simpler than that of Siegel[24] mentioned in Section 1, and for $d = O(c)$, the evaluation takes $O(c)$ time as opposed to the $O(c)^c$ time used by Siegel.

Complementing the above result, Dahlgaard et al.[7] have proved that double tabulation is likely to be truly random for any specific set S with less than $(1 - \Omega(1))\Sigma$ keys:

THEOREM 8 (Dahlgaard et al.[7]). *Given a set $S \subseteq [u]$ of size $(1 - \Omega(1))\Sigma$, consider two random simple tabulation function $h_0 : \Sigma^c \to \Sigma^d$ and $h_1 : \Sigma^d \to [2^r]$. With probability $1 - O(\Sigma^{1-\lceil d/2 \rceil})$, every nonempty subset $X \subseteq S$ gets a unique output character with h_0, and then the double tabulation function $h_1 \circ h_0$ is fully random over S.*

It is interesting to compare Theorem 8 with Theorem 7. Theorem 8 holds for one large set whereas Theorem 7 works for all small sets. Also, Theorem 8 with $d = 4$ "derived" characters gets essentially the same error probability as Theorem 7 with $d = 6c$ derived characters.

Siegel[24] has proved that with space Σ, we cannot in constant time hope to get independence higher than $\Sigma^{1-\Omega(1)}$, which is much less than the size of the given set in Theorem 8.

Theorem 8 provides an extremely simple $O(n)$ space implementation of a constant time hash function that is likely uniform on any given set S. This is much simpler than the previous linear space uniform hashing of Pagh and Pagh,[17, Section 3] which needs the high independence of Theorem 7 as a subroutine. We note that[17, Section 4] presents a general trick to reduce the space from linear, that is, $O(n(\lg n + \lg |\mathcal{R}|))$ bits, downto $(1+\varepsilon)n \lg |\mathcal{R}| + O(n)$ bits, preserving the constant evaluation time. This reduction can also be applied to Theorem 8 so that we also get a simpler overall construction for a succinct dictionary using $(1+\varepsilon)n \lg |\mathcal{R}| + O(n)$ bits of space and constant evaluation time.

Very recently, Christiani et al.[6] have shown that we using a more elaborate recursive tabulation scheme can get quite to Siegel's lower-bound.

THEOREM 9 (Christiani et al.[6, Corollary 3]). *For word-size w, and parameters k and $c = O(w/(\log k))$, with probability $1 - u^{1/c}$, we can construct a k-independent hash function $h : [2^w] \to [2^w]$ in $O(cku^{1/c})$ time and space, that is, evaluated in $O(c \log c)$ time.*

In Theorem 8, we used the same space to get independence $\Sigma^{1/(5c)}$ and evaluation time $O(c)$. The construction of Theorem 7 is also simpler. We should thus use Theorem 9 if we need its very high independence, but if, say, logarithmic independence suffices, then Theorem 7 is the better choice.

A major open problem is get the space and independence of Theorem 9 but with $O(c)$ evaluation time, matching the lower bound of Ref.[24] In its full generality, the lower bound from Ref.[24] says that we for independence k with $c < k$ cell probes need space $\Omega(k(u/k)^{1/c}c)$.

4.1. Invertible bloom filters with simple tabulation

Theorem 8 states that if a random simple tabulation function $h_0 : \Sigma^c \to \Sigma^d$ is applied to a given set S of size $(1 - \Omega(1))\Sigma$, then with probability $1 - O(\Sigma^{1-\lfloor d/2 \rfloor})$, every nonempty subset $X \subseteq S$ gets a unique output character. This is not only relevant for fully random hashing. This property is also sufficient for the hash function in Goodrich and Mitzenmacher's Invertible Bloom Filters,[11] which have found numerous applications in streaming and data bases.

4.2. *k*-Partitions via mixed tabulation

The general goal of Dahlgaard et al.[7] is a hash function for k-partitioning a set into bins so that we get good concentration bounds when combining statistics from each bin.

To understand this point, suppose we have a fully random hash function applied to a set X of red and blue balls. We want to estimate the fraction f of red balls. The idea of minwise hashing is to sample the ball with the smallest hash value. This sample is uniformly random and is

red with probability f. If we repeat the experiment k times with k independent hash functions, we get a multiset S of k samples with replacement from X and the fraction red balls in S concentrates around f as we increase the number of samples.

Consider the alternative experiment using a single hash function, where we use some bits of the hash value to partition X into k bins, and then use the remaining bits as a local hash value. We pick the ball with the smallest hash value in each bin. This is a sample S from X without replacement, and again, the fraction of red balls is concentrated around f.

The big difference between the two schemes is that the second one runs $\Omega(k)$ times faster. In the first experiment, each ball participated in k independent experiments, but in the second one with k-partitions, each ball picks its bin, and then only participates in the local experiment for that bin. Thus with the k-partition, essentially, we get k experiments for the price of one.

This generic idea has been used for different types of statistics. Flajolet and Martin[10] introduced it to count the number of distinct items in a multiset, and recently, Li et al.[14] used it for Minwise estimation of the Jaccard Similarity of two sets.

The issue is that no realistic hashing scheme was known to make a good enough k-partition for the above kind of statistics to make sense. The point is that the contents of different bins may be too correlated, and then we get no better concentration with a larger k. In the independence paradigm of Wegman and Carter,[26] it would seem that we need independence at least k to get sufficiently independent statistics from the different bins.

An efficient solution is based on a variant of double tabulation described below.

4.3. Mixed tabulation

For Theorem 8, we may use $d = 4$ even if c is larger, but then h_0 will introduce many collisions. To avoid this problem we mix the schemes in *mixed tabulation*. Mathematically, we use two simple tabulation hash functions $h_1 : [u] \to \Sigma^d$ and $h_2 : \Sigma^{c+d} \to [2^r]$, and define the hash function $h(x) \mapsto h_2(x \circ h_1(x))$, where $\circ$ denotes concatenation of characters. We call $x \circ h_1(x)$ the *derived key*, consisting of c original characters and d derived characters. Since the derived keys includes the original keys, there are no duplicate keys.

We note that mixed tabulation only requires $c+d$ lookups if we instead store simple tabulation functions $h_{1,2} : \Sigma^c \to \Sigma^d \times [r]$ and $h_2' : \Sigma^d \to [r]$, computing $h(x)$ by $(v_1, v_2) = h_{1,2}(x)$; $h(x) = v_1 \oplus h_2(v_2)$. This efficient implementation is similar to that of twisted tabulation, and is equivalent to the previous definition. As long as we have at least one derived character, mixed tabulation has all the distribution properties of twisted tabulation, particularly, the Chernoff-style concentration bound from Theorem 5. At the same time, we get the full randomness from Theorem 8 for any given set S of size $(1 - \Omega(1))\Sigma$. Based on these properties and more, it is proved in Ref.[7] that mixed tabulation, w.h.p., gets essentially the same concentration bounds as full randomness for all of the abovementioned statistics based on k-partitions.

Acknowledgments

Research partly supported by Advanced Grant DFF-0602-02499B from the Danish Council for Independent Research under the Sapere Aude research carrier program.

References

1. Alon, N., Matias, Y., Szegedy, M. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci. 58*, 1 (1999), 209–223, 1999. Announced at STOC'96.
2. Azar, Y., Broder, A.Z., Karlin, A.R., Upfal, E. Balanced allocations. *SIAM J. Comput. 29*, 1 (1999), 180–200. Announced at STOC'94.
3. Braverman, V., Chung, K.-M., Liu, Z., Mitzenmacher, M., Ostrovsky, R. AMS without 4-wise independence on product domains. In *Proceedings of the 27th STACS* (2010), 119–130.
4. Broder, A.Z., Charikar, M., Frieze, A.M., Mitzenmacher, M. Min-wise independent permutations. *J. Comput. Syst. Sci. 60*, 3 (2000), 630–659. Announced at STOC'98.
5. Carter, L., Wegman, M.N. Universal classes of hash functions. *J. Comput. Syst. Sci. 18*, 2 (1979), 143–154. Announced at STOC'77.
6. Christiani, T., Pagh, R., Thorup, M. From independence to expansion and back again. In *Proceedings of the 47th STOC* (2015), 813–820.
7. Dahlgaard, S., Knudsen, M.B.T., Rotenberg, E., Thorup, M. Hashing for statistics over k-partitions. In *Proceedings of the 56th FOCS* (2015), 1292–1310.
8. Dahlgaard, S., Knudsen, M.B.T., Rotenberg, E., Thorup, M. The power of two choices with simple tabulation. In *Proceedings of the 27th SODA* (2016), 1631–1642.
9. Dahlgaard, S., Thorup, M. Approximately minwise independence with twisted tabulation. In *Proceedings of the 14th SWAT* (2014), 134–145.
10. Flajolet, P., Martin, G.N. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci. 31*, 2 (1985), 182–209.
11. Goodrich, M.T., Mitzenmacher, M. Invertible bloom lookup tables. In *Proceedings of the 49th Allerton Conference on Communication, Control, and Computing* (2011), 792–799.
12. Indyk, P. A small approximately min-wise independent family of hash functions. *J. Algorithms 38*, 1 (2001), 84–90. Announced at SODA'99.
13. Knuth, D.E. Notes on open addressing. Unpublished memorandum. See http://citeseer.ist.psu.edu/knuth63notes.html, 1963.
14. Li, P., Owen, A.B., Zhang, C.-H. One permutation hashing. In *Proceedings of the 26th NIPS* (2012), 3122–3130.
15. Mitzenmacher, M., Vadhan, S.P. Why simple hash functions work: Exploiting the entropy in a data stream. In *Proceedings of the 19th SODA* (2008), 746–755.
16. Motwani, R., Raghavan, P. *Randomized Algorithms*. Cambridge University Press, 1995.
17. Pagh, A., Pagh, R. Uniform hashing in constant time and optimal space. *SIAM J. Comput. 38*, 1 (2008), 85–96.
18. Pagh, A., Pagh, R., Ružić, M. Linear probing with constant independence. *SIAM J. Comput. 39*, 3 (2009), 1107–1120. Announced at STOC'07.
19. Pagh, R., Rodler, F.F. Cuckoo hashing. *J. Algorithms 51*, 2 (2004), 122–144. Announced at ESA'01.
20. Pătraşcu, M., Thorup, M. On the *k*-independence required by linear probing and minwise independence. In *Proceedings of the 37th ICALP* (2010), 715–726.
21. Pătraşcu, M., Thorup, M. The power of simple tabulation-based hashing. *J. ACM 59*, 3 (2012), Article 14. Announced at STOC'11.
22. Pătraşcu, M., Thorup, M. Twisted tabulation hashing. In *Proceedings of the SODA* (2013), 209–228.
23. Schmidt, J.P., Siegel, A., Srinivasan, A. Chernoff-Hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math. 8*, 2 (1995), 223–250. Announced at SODA'93.
24. Siegel, A. On universal classes of extremely random constant-time hash functions. *SIAM J. Comput. 33*, 3 (2004), 505–543. Announced at FOCS'89.
25. Thorup, M. Simple tabulation, fast expanders, double tabulation, and high independence. In *Proceedings of the 54th FOCS* (2013), 90–99.
26. Wegman, M.N., Carter, L. New classes and applications of hash functions. *J. Comput. Syst. Sci. 22*, 3 (1981), 265–279. Announced at FOCS'79.
27. Zobrist, A.L. A new hashing method with application for game playing. Technical Report 88, Computer Sciences Department, University of Wisconsin, 1970.

Mikkel Thorup (mikkel2thorup@gmail.com), Department of Computer Science, University of Copenhagen, Denmark.

California Institute of Technology
Lecturer in Computing and Mathematical Sciences

The Department of Computing and Mathematical Sciences (CMS) at the California Institute of Technology invites applications for the position of Lecturer in Computing and Mathematical Sciences. This is a (non-tenure-track) career teaching position, with full-time teaching responsibilities. The start date for the position ideally is **September 1, 2017** and the initial term of appointment can be up to three years.

The lecturer will teach introductory computer science courses including data structures, algorithms and software engineering, and will work closely with the CMS faculty on instructional matters. The ability to teach intermediate-level undergraduate courses in areas such as software engineering, computing systems and/or compilers is desired. The lecturer may also assist in other aspects of the undergraduate program, including curriculum development, academic advising, and monitoring research projects. The lecturer must have a track record of excellence in teaching computer science to undergraduates. In addition, the lecturer will have opportunities to participate in research projects in the department. An advanced degree in Computer Science or related field is desired but not required.

Applications will be accepted on an ongoing basis until the position is filled.

Please view the application instructions and apply on-line at https://applications.caltech.edu/job/cmslect

The California Institute of Technology is an Equal Opportunity/Affirmative Action Employer. Women, minorities, veterans, and disabled persons are encouraged to apply.

der Bundeswehr
Universität München

The Universität der Bundeswehr München (Bundeswehr University Munich) is significantly expanding its Cyber Defence Research Center (CODE). CODE was established in 2013 with the objective to bring together experts from different faculties and scientific disciplines as well as expertise from industry and government agencies to conduct research in the cyber and information space. CODE pursues a comprehensive, integrated, and interdisciplinary approach to implement technical innovations and concepts for the protection of data, software, and ICT infrastructures in accordance with legal and commercial framework conditions. It has already established important strategic partnerships in this area. The objective of the expansion is to unite the Bundeswehr's and the Federal Government's research initiatives in the area of Cyber Defence and Smart Data and to establish the CODE Research Center as the primary point of contact in the cyber and information domain of the Bundeswehr and the Federal Government.

Research and teaching in the area of cyber security is already being carried out as part of the bachelor's and master's programs in the Computer Science Department. According to current planning, a new international master's program in Cyber Security will be launched on January 1st, 2018.

The Universität der Bundeswehr München will therefore be appointing **four professors** for its Computer Science Department **on April 1st, 2018.**

The Universität der Bundeswehr München is looking for personalities with outstanding scientific qualifications to fill these professorial positions, who will also contribute actively to the CODE research center. Besides excellent research work, the new professors are expected to develop demanding lectures, practicals, and seminars for the new master's program in Cyber Security and to provide excellent teaching in their respective specialist area. Applicants are also expected to carry out teaching in the bachelor's programs in computer science and business informatics, and to work closely with the other departments at the Universität der Bundeswehr München.

The professorships will be provided with excellently equipped laboratories housed in a new building that is to be completed in the near future.

The candidates must have an excellent scientific track record, as demonstrated by a habilitation or equivalent scientific achievements, as well as significant excellent publications in academic journals. Proven teaching experience in their respective specialist area is highly desired. The new professors should have an international perspective, e.g., based on participation in international research projects, and experience in acquiring third-party funding. The duties will also include active participation in the university's academic self-administration.

The Computer Science Department at the Universität der Bundeswehr München is seeking professors for the following specialist areas of its Cyber Defence und Smart Data Research Center:

University Professorship (W3) in Digital Forensic

The reconstruction and investigation of offenses involving fixed or mobile electronic devices and complex software systems requires methodologies for verifying or falsifying hypotheses. The challenge of analyzing security incidents as well as using seized IT components as court-type evidence is increased by the requirement to systematically and accurately extract and document tiny digital traces from steadily increasing amounts of data. As IT forensics experts often cannot know about the details they should be looking for at the beginning of a case, specifying legally compliant digital-forensics methods and supporting them with tools while ensuring the chain of custody is essential.

We are looking for an excellent, internationally oriented personality that is particularly well-known in the field of digital forensics, e.g., storage media analysis and multimedia forensics. Given the increasing amounts of data that needs to be analyzed, also research experience regarding novel approaches for scalable and automated forensics methods is required.

University Professorship (W3) in Data Science

Big Data denotes huge amounts of data that cannot be analyzed or processed using conventional methodologies. By applying intelligent processing steps and semantics-based technologies, big data (raw data) can be transformed into valuable information, which is the foundation for developing innovative applications and business models as well as optimizing existing business models.

We are looking for an excellent, internationally oriented personality that is particularly well-known in the field of big data analytics, predictive analytics, data quality, and data security. Experience in the application of big data analytics to the cyber-security domain is highly desired.

University Professorship (W3) in Machine Learning

Machine learning as a segment of artificial intelligence strives for the development of methodologies and algorithms to implement adaptive technical systems. Machine learning algorithms are the basis for the future development of smart systems, which adapt to new situations and thus can generate knowledge from their own experiences.

We are looking for an excellent, internationally oriented personality that is particularly well-known in the field of machine learning for pattern recognition, prognostic maintenance and decision making, deep learning, and data-based adaptive, self-learning and self-optimizing systems. Experience in the application of machine learning to the cyber-security domain is highly desired.

University Professorship (W3) in IT System Hardening

Errors made during programming, while adapting a system to its operating environment, or negligence when using IT systems lead to vulnerabilities that allow attackers to gain unauthorized access to data or take complete control over a system. In addition to company networks, networks for the control of industrial plants (SCADA), highly secure networks, but also complex military and defence systems, are still severely endangered by sophisticated and high-quality attacks (Advanced Persistent Threats, APT). For example, manipulations on the hardware level can lead to vulnerabilities that are hard to identify. The systematic analysis of vulnerabilities in IT systems and test procedures (for example, penetration tests) for their identification and evaluation is the basis for increasing the security level of networked applications. It is important to develop novel holistic approaches for the identification of IT vulnerabilities. Furthermore, novel protection concepts for identified vulnerabilities are to be developed, taking into account hardware limitations, real-time constraints or requirements regarding certifications in the field of industrial control systems and critical infrastructures.

We are looking for an excellent, internationally oriented personality that is particularly well-known in the field of IT vulnerability management and penetration testing in research and teaching. In addition, experience in the hardening of COTS (Commercial off the Shelf) products is expected.

The Universität der Bundeswehr München offers academic programs directed primarily at officer candidates and officers, who can obtain bachelor's and master's degrees within a trimester system. Depending on spare capacity, civilian students are allowed to enroll. The study programs are complemented by interdisciplinary elements in an integrated program entitled "studium plus".

Preconditions of employment and the legal duty positioning of professors are based upon the "Bundesbeamtengesetz". Employment as a "Beamte/r" requires that the candidate be no older than 50 at the date of appointment.

The University seeks to increase the number of female professors and thus explicitly invites women to submit applications. Severely disabled candidates with equal qualifications will receive preferential consideration.

Please submit your application documents marked as Confidential Personnel Matter to the Department Head of the Computer Science Department at the Universität der Bundeswehr München, DE-85577 Neubiberg, by July 24th, 2017.

DOI:10.1145/3098273

Dennis Shasha

Upstart Puzzles
Ruby Risks

YOU HAVE THREE covered boxes of Burmese rubies before you. You know there are a total of 30 identical seven-carat rubies in the three boxes. You can ask for a certain number of rubies from each box. If you ask for more than there are, you get none from that box. Otherwise, you get what you asked for from that box. For now, suppose you must state your requests in advance for all three boxes and have no chance to change your mind; that is, with no feedback.

Warm-Up. Given no further constraints, how many rubies do you ask for from each box?

Solution to Warm-Up. If you ask for, say, 10 from each box, then one box may have 30 and the others 10, so you may get only 10. If you ask for x, where $x < 10$, you might get only x, if one box has all 30. On the other hand, if you ask for more than 10, then they might evenly be distributed (10 rubies in each box), meaning you get none. You should thus ask for 10, because 10 is the maximum you are guaranteed to get.

Suppose you know one box has four more rubies than some other box and the remaining box could have any number. Again, you must make requests for all boxes at once, so you cannot use your winnings from earlier boxes to help you determine what to do later. How many rubies in total can you guarantee to receive?

Solution. The boxes have x, $x+4$, and y. Here are the possibilities, in ascending order, from what is in the first box:

(i) 0, 4, 26; (ii) 1, 5, 24; (iii) 2, 6, 22; (iv) 3, 7, 20; (v) 4, 8, 18; (vi) 5, 9, 16; (vii) 6, 10, 14; (vii) 7, 11, 12; (vii) 8, 12, 10; (viii) 9, 13, 8; (xi) 10, 14, 6; (xii) 11, 15, 4; and (xiii) 12, 16, 2

How many of these red beauties could you obtain if you properly curbed your greed?

You are guaranteed to receive 12 rubies by asking for 12 from all three. No other scenario guarantees you more rubies.

Upstart 1. (Feedback) Suppose the boxes are laid out in left-to-right order and you may state your request for the leftmost box first, then, based on how many rubies you receive, state your request for the middle box, and then the rightmost one. When you are done, you can repeat this process (left, then middle, then right). For both the warm-up scenario and the scenario outlined in the first question—where one box has four more rubies than some other box—how many more rubies can you guarantee to get in this "2-feedback" scenario?

Upstart 2. Suppose you are told there are b boxes with r rubies overall and that one box has d more rubies than some other box. Can you formulate an algorithm for both the f-feedback (meaning you go left, middle, right f times) and no-feedback scenario that is optimal? If so, please explain your algorithm in pseudocode and send links to your software.

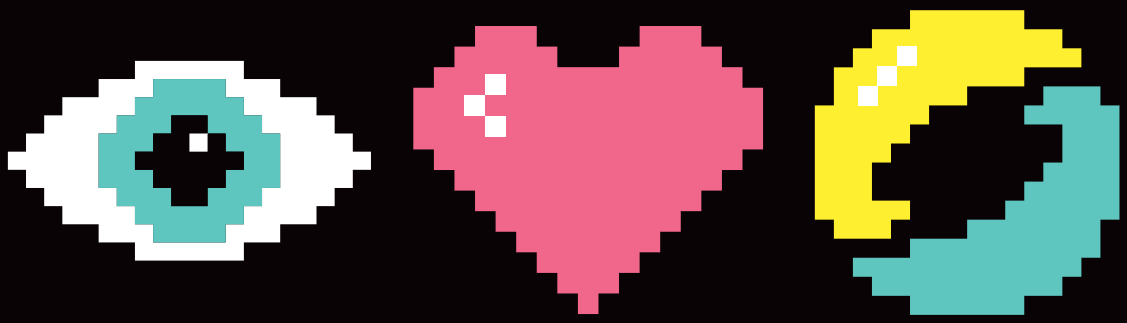
All are invited to submit their solutions to upstartpuzzles@cacm.acm.org; solutions and discussion will be posted at http://cs.nyu.edu/cs/faculty/shasha/papers/cacmpuzzles.html

Dennis Shasha (dennisshasha@yahoo.com) is a professor of computer science in the Computer Science Department of the Courant Institute at New York University, New York, as well as the chronicler of his good friend the omniheurist Dr. Ecco.

PHOTO BY VICTOR MOUSSA

SIGGRAPH2017

AT THE  of COMPUTER GRAPHICS & INTERACTIVE TECHNIQUES



REGISTER TODAY

30 JULY – 3 AUGUST | *Los Angeles, California*

S2017.SIGGRAPH.ORG/REGISTRATION

Sponsored by ACMSIGGRAPH