

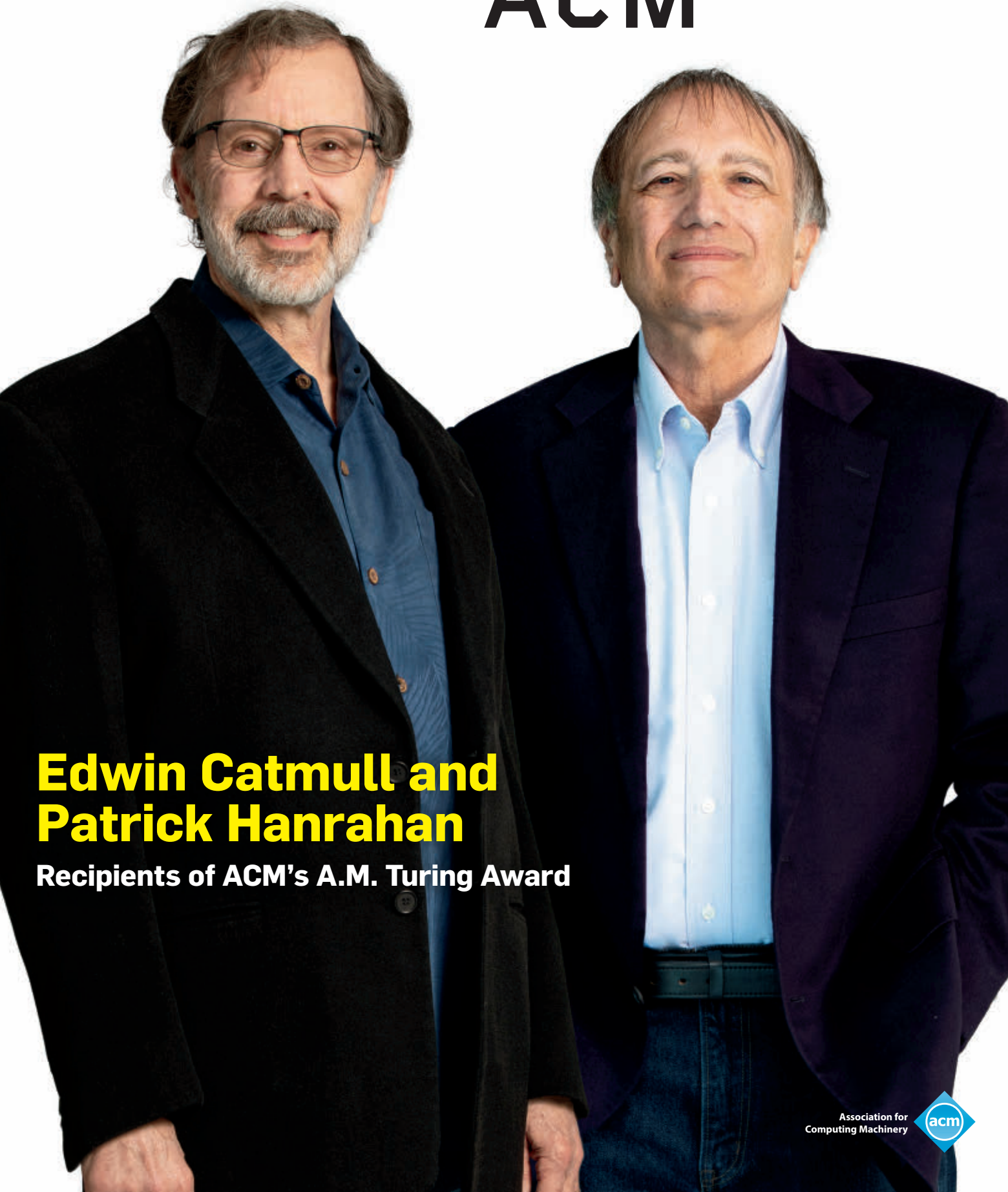
COMMUNICATIONS

CACM.ACM.ORG

OF THE

ACM

06/2020 VOL.63 NO.06



Edwin Catmull and Patrick Hanrahan

Recipients of ACM's A.M. Turing Award

Association for
Computing Machinery



ACM SYMPOSIUM ON VIRTUAL REALITY SOFTWARE & TECHNOLOGY

www.vrst.acm.org/vrst2020



VRST | 2020
OTTAWA

OTTAWA
NOV 1-4 2020

Preceded by
ACM SUI 2020

IMPORTANT DATES

- JUN 29 PAPERS
- AUG 03 POSTERS AND DEMOS
- AUG 17 NOTIFICATIONS
- NOV 1-4 CONFERENCE IN OTTAWA, CANADA

OUR SPONSORS



*Thank you, Ed,
for always reminding us that
we can never achieve the unexpected by
sticking to the familiar.*



The University of Utah and its 125-year-old College of Engineering extend our heartfelt congratulations to distinguished alumnus, advisory board member and friend, Ed Catmull, for a lifetime of stunning technical and creative achievements.

Ed began making computer graphics “magic” as a graduate student at the U, where he was awarded a PhD in 1974 for his groundbreaking techniques.



“Getting the right people and the right chemistry is more important than getting the right idea.”



COLLEGE OF ENGINEERING
THE UNIVERSITY OF UTAH

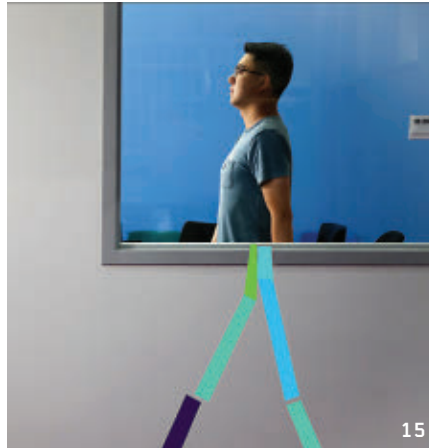
Departments

- 5 **Editor's Letter**
What Do DDT and Computing Have in Common?
By Andrew A. Chien
-
- 6 **Letters to the Editor**
Safety Proposal Points in Same Direction
-
- 7 **Cerf's Up**
Implications of the COVID-19 Pandemic
By Vinton G. Cerf
-
- 8 **BLOG@CACM**
Detecting/Preventing Infections, and Moving Instruction Online
Terrence DeFranco suggests the Internet of Things could be keeping us safer, and Jeremy Roschelle airs issues related to online instruction.
-
- 23 **Calendar**

Last Byte

- 96 **Q&A**
Attaining The Third Dimension
ACM A.M. Turing Award recipients Ed Catmull and Pat Hanrahan discuss how they helped to bring the power of three-dimensional imagery to computer graphics.
By Leah Hoffmann

News



- 10 **An Animating Spirit**
This year's ACM A.M. Turing Award recipients, Ed Catmull and Pat Hanrahan, overcame industry indifference to found Pixar and put their computer graphics expertise to work.
By Neil Savage



Watch the recipients discuss their work in the exclusive *Communications* video.
<https://cacm.acm.org/videos/2019-acm-turing-award>

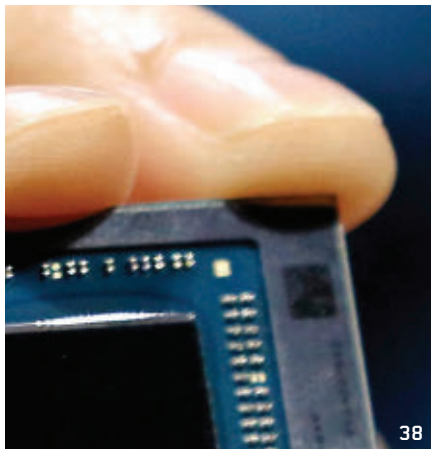
- 13 **Leveraging Unlabeled Data**
Deep learning looks for better pretexts.
By Chris Edwards
-
- 15 **Seeing Through Walls**
Artificial intelligence makes sense of radio signals to understand what someone in another room is doing.
By Neil Savage
-
- 17 **Hiring from the Autism Spectrum**
Companies increasingly are looking to hire people who are on the autism spectrum to fill IT roles.
By Esther Shein

Viewpoints

- 20 **Inside Risks**
How to Curtail Oversensing in the Home
Limiting sensitive information leakage via smart-home sensor data.
By Connor Bolton, Kevin Fu, Josiah Hester, and Jun Han
-
- 25 **Kode Vicious**
Kode Vicious Plays in Traffic
With increasing complexity comes increasing risk.
By George V. Neville-Neil
-
- 27 **The Profession of IT**
Technology Adoption
The S-shaped curve of technology adoption is a welcome recurrence in an otherwise chaotic adoption world.
By Peter J. Denning and Ted G. Lewis
-
- 30 **Viewpoint**
Studying Programming in the Neuroage: Just a Crazy Idea?
Programming research has entered the Neuroage.
By Janet Siegmund, Norman Peitek, André Brechmann, Chris Parnin, and Sven Apel
-
- 35 **Viewpoint**
AI and Accessibility
A discussion of ethical considerations.
By Meredith Ringel Morris



Practice



38

38 **Commit to Memory**
Chipping away at Moore's Law.
By Jessie Frazelle

42 **Communicate Using the Numbers 1, 2, 3, and More**
Leveraging expectations for better communication.
By Thomas A. Limoncelli

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

About the Cover:

This issue spotlights the recipients of the 2019 ACM A.M. Turing Award—Edwin E. Catmull (left) and Patrick M. Hanrahan—for their fundamental contributions to 3D computer graphics and the revolutionary impact of these techniques on computer-generated imagery in filmmaking and other applications. Photographed by Richard Morgenstein; www.morgenstein.com



Contributed Articles

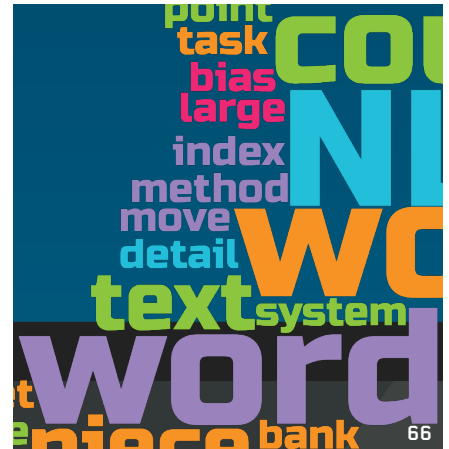


46

46 **Meltdown: Reading Kernel Memory from User Space**
Lessons learned from Meltdown's exploitation of the weaknesses in today's processors.
By Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, Mike Hamburg, and Raoul Strackx

57 **The 'Invisible' Materiality of Information Technology**
It's difficult to see the ecological impact of IT when its benefits are so blindingly bright.
By Alan Borning, Batya Friedman, and Nick Logler

Review Articles



66

66 **Contextual Word Representations: Putting Words into Computers**
Advances in how programs treat natural language words have a big impact in AI.
By Noah A. Smith



Watch the author discuss this work in the exclusive *Communications* video. <https://cacm.acm.org/videos/contextual-word-representations>

75 **Street Lamps as a Platform**
Strategically augmented street lamps can become the key enabling technology in smart cities.
By Max Mühlhäuser, Christian Meurisch, Michael Stein, Jörg Daubert, Julius Von Willich, Jan Riemann, and Lin Wang

Research Highlights

86 **Technical Perspective**
Algorithm Selection as a Learning Problem
By Avrim Blum

87 **Data-Driven Algorithm Design**
By Rishi Gupta and Tim Roughgarden



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

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

ACM COUNCIL
President
Cherri M. Pancake
Vice-President
Elizabeth Churchill
Secretary/Treasurer
Yannis Ioannidis
Past President
Alexandra L. Wolf
Chair, SGB Board
Jeff Jortner
Co-Chairs, Publications Board
Jack Davidson and Joseph Konstan
Members-at-Large
Gabriele Kotsis; Susan Dumais;
Renée McCauley; Claudia Bauzer Medeiros;
Elizabeth D. Mynatt; Pamela Samuelson;
Theo Schlossnagle; Eugene H. Spafford
SGB Council Representatives
Sarita Adve and Jeanna Neefe Matthews

BOARD CHAIRS
Education Board
Mehran Sahami and Jane Chu Prey
Practitioners Board
Terry Coatta

REGIONAL COUNCIL CHAIRS
ACM Europe Council
Chris Hankin
ACM India Council
Abhram Ranade
ACM China Council
Wenguang Chen

PUBLICATIONS BOARD
Co-Chairs
Jack Davidson and Joseph Konstan
Board Members
Phoebe Ayers; Nicole Forsgren; Chris Hankin;
Mike Heroux; Nenad Medvidovic;
Tulika Mitra; Michael L. Nelson;
Sharon Olviatt; Eugene H. Spafford;
Stephen N. Spencer; Divesh Srivastava;
Robert Walker; Julie R. Williamson

ACM U.S. Technology Policy Office
Adam Eisgrau
Director of Global Policy and Public Affairs
1701 Pennsylvania Ave NW, Suite 200,
Washington, DC 20006 USA
T (202) 580-6555; acmpo@acm.org

Computer Science Teachers Association
Jake Baskin
Executive Director

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

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

STAFF
DIRECTOR OF PUBLICATIONS
Scott E. Delman
cacm-publisher@cacm.acm.org

Executive Editor
Diane Crawford
Managing Editor
Thomas E. Lambert
Senior Editor
Andrew Rosenbloom
Senior Editor/News
Lawrence M. Fisher
Web Editor
David Roman
Editorial Assistant
Danbi Yu

Art Director
Andrij Borys
Associate Art Director
Margaret Gray
Assistant Art Director
Mia Angelica Balaquiot
Production Manager
Bernadette Shade
Intellectual Property Rights Coordinator
Barbara Ryan
Advertising Sales Account Manager
Ilia Rodriguez

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

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

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

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

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

ACM ADVERTISING DEPARTMENT
1601 Broadway, 10th Floor
New York, NY 10019-7434 USA
T (212) 626-0686
F (212) 869-0481

Advertising Sales Account Manager
Ilia Rodriguez
ilia.rodriguez@hq.acm.org

Media Kit acmm mediasales@acm.org

Association for Computing Machinery (ACM)
1601 Broadway, 10th Floor
New York, NY 10019-7434 USA
T (212) 869-7440; F (212) 869-0481

EDITORIAL BOARD
EDITOR-IN-CHIEF
Andrew A. Chien
aie@cacm.acm.org
Deputy to the Editor-in-Chief
Morgan Denlow
cacm.deputy.to.eic@gmail.com
SENIOR EDITOR
Moshe Y. Vardi

NEWS
Co-Chairs
Marc Snir and Alain Chesnais
Board Members
Tom Conte; Monica Divitini; Mei Kobayashi;
Rajeev Rastogi; François Sillion

VIEWPOINTS
Co-Chairs
Tim Finin; Susanne E. Hambrusch;
John Leslie King
Board Members
Terry Benzel; Michael L. Best; Judith Bishop;
Lorrie Cranor; Boi Falting; James Grimmelmann;
Mark Guzdial; Haym B. Hirsch;
Richard Ladner; Carl Landwehr; Beng Chin Ooi;
Francesca Rossi; Len Shustek; Loren Terveen;
Marshall Van Alstyne; Jeannette Wing;
Susan J. Winter

PRACTICE
Co-Chairs
Stephen Bourne and Theo Schlossnagle
Board Members
Eric Allman; Samy Bahra; Peter Bailis;
Betsy Beyer; Terry Coatta; Stuart Feldman;
Nicole Forsgren; Camille Fournier;
Jessie Frazelle; Benjamin Fried; Tom Killalea;
Tom Limoncelli; Kate Matsudaira;
Marshall Kirk McKusick; Erik Meijer;
George Neville-Neil; Jim Waldo;
Meredith Whittaker

CONTRIBUTED ARTICLES
Co-Chairs
James Larus and Gail Murphy
Board Members
Robert Austin; Kim Bruce; Alan Bundy;
Peter Buneman; Jeff Chase;
Yannis Ioannidis; Gal A. Kaminka;
Ben C. Lee; Igor Markov;
Lionel M. Ni; Doina Precup;
Shankar Sastry; m.c. schraefel; Ron Shamir;
Hannes Werthner; Reinhard Wilhelm

RESEARCH HIGHLIGHTS
Co-Chairs
Azer Bestavros, Shriram Krishnamurthi,
and Orna Kupferman
Board Members
Martin Abadi; Amr El Abbadi;
Animashree Anandkumar; Sanjeev Arora;
Michael Backes; Maria-Florina Balcan;
David Brooks; Stuart K. Card; Jon Crowcroft;
Alexei Efros; Bryan Ford; Alon Halevy;
Gernot Heiser; Takeo Igarashi;
Srinivasan Keshav; Sven Koenig;
Ran Libeskind-Hadas; Karen Liu; Greg Morrisett;
Tim Roughgarden; Guy Steele, Jr.;
Robert Williamson; Margaret H. Wright;
Nicolai Zeldovich; Andreas Zeller

SPECIAL SECTIONS
Co-Chairs
Sriram Rajamani, Jakob Rehof,
and Haibo Chen
Board Members
Tao Xie; Kenjiro Taura; David Padua

ACM Copyright Notice
Copyright © 2020 by Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to publish from permissions@hq.acm.org or fax (212) 869-0481.

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

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

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

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

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

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

Printed in the USA.



Association for Computing Machinery





Andrew A. Chien

DOI:10.1145/3397339

What Do DDT and Computing Have in Common?

WRITING ON THE 50th Earth Day brings to mind the origins of U.S. environmental movement.

DDT is, of course, Bis(4-chlorophenyl)-1,1,1-trichloroethane, perhaps the most effective insecticide ever invented. DDT was used widely with remarkable effectiveness in the 1940s and 1950s to combat malaria, typhus, and the other insect-borne human diseases. Its efficacy was unsurpassed in insect control for crop and livestock production, and even villages and homes. In short, it was a wonder chemical.⁷

Following Rachel Carson's *Silent Spring*² highlighting DDT's persistent negative environmental effects, including precipitous declines in wildlife and multiple human health problems, Carson's advocacy led to the banning of DDT and the creation of the U.S. Environmental Protection Agency (1970).

DDT and computing both have amazing benefits, but now that I've got your attention, my point is they also both have significant negatives. And, their good doesn't offset their bad. So, in addition to reaping computing's bounty (education, information access, entertainment, commerce, efficiency, and more), we should "own" and work to reduce the negative impacts of computing.

What are computing's negative environmental impacts?

- ▶ Carbon emissions of >500 million metric tons of CO₂/year
- ▶ E-waste of >50 million metric tons/year

Both are growing fast and need attention!³ Even AI is an exemplar of computing's dual nature—driving remarkable advances whilst driving increased computing carbon emissions.^{1,4}

Is this critical? Is it urgent? Climate

change is regularly called an existential crisis for humanity. Further, as Greta Thunberg, who led worldwide protests in 2019 that dwarfed all previous environmental rallies, points out—all must do their part to solve our climate challenge.

"I'm just a technologist, what can I do?" Here are some ideas.

Carbon emissions from computing are growing: the hyperscale cloud is the headline driver—with power consumption increasing 6.5-fold from 2010 to 2018, and even faster since 2018.⁵ But cloud computing can become truly carbon neutral, 7x24 hourly matching! The opportunity for this transformation is the rise of low-cost renewable generation in power grids that produces excess supply (Terawatt-hours) of carbon-free power worldwide. Despite the challenge of the volatility of these renewable excesses, research has proven models that "flex" computing in space and time, and thereby produce zero-carbon compute can be scientifically and economically feasible.⁸

For emissions, we should *adopt a goal of zero carbon emission computing for operations*. To achieve this, we must design applications and workloads and manage resources to time- and space-shift computing to align with the availability of renewables. This is possible for the cloud, due in no small part to efficient hyperscale datacenters, global scale, and spectacular management and optimization systems! Lest you still consider this idealistic, there is real movement in this direction.⁶

Computing e-waste is growing rapidly with the explosion of IoT, universal Internet access, and 5G networks. We must extend the lifetime of computing hardware—servers and clients—and increase the circularity of the computing economy far beyond the <20% of e-waste

that avoids landfills today.

For e-waste, we should *adopt a goal to double the lifetime of computing hardware*, increasing server lifetime far beyond six years for cloud and enterprise computing, and for smartphones as high as eight years. In vertical, global ecosystems controlled by a few players, there are increasing opportunities. In a new NSF OAC project we are working with universities to create an extended ecosystem for servers that combines zero-carbon operation with extended lifetime.

What can software professionals do? Software can be designed and tuned for efficiency and memory size, enabling client devices to remain viable for over eight years. Software upgrades should have as a design goal to avoid driving client hardware obsolescence. For the cloud, software professionals should reengineer applications to be time flexible, adapting the availability of excess renewable energy.

What can hardware professionals do? Designs can be repairable and upgradeable, and complemented by network services for longer life. Products designed for de-manufacture can reduce environmental pollution and enable circular economy. Create and drive circular economy practice—for all computing hardware.

Changing how computing does business is important. If we undertake these goals, our technologies will be both more flexible and more broadly useful. And perhaps we will attract and inspire more young computing professionals!

Happy 50th Earth Day! I'm looking forward to zero-carbon and zero e-waste computing at the 60th in 2030!

Be safe and be well!

Andrew A. Chien, EDITOR-IN-CHIEF

For references, see page 6.

Safety Proposal Points in Same Direction

IN HER FEBRUARY 2020 column (“Are You Sure Your Software Will Not Kill Anyone?”), Nancy Leveson says the solution to software safety is not “building a software architecture and generating the requirements later.” Reading this, we were surprised that anyone would propose such an approach, or that Leveson would find it necessary to argue against it. We were even more surprised to see that Leveson attributes the proposal to a National Academies report that we edited.¹

Perhaps Leveson had a different report in mind. Our report actually substantiates the very arguments she makes in her column, identifying the same misconceptions that she notes, along with others—and provides citations for the earlier origins of these supposedly new arguments.

Contrary to Leveson’s article, however, our report does not take the design as given and claim that “analysis tools can be developed to analyze the safety of complex systems.” Rather than treating safety as a quality to be established by an ex post facto analysis, our report calls for the design to be shaped by the safety requirements. Our report’s key point is that safety needs a compelling and reasoned argument and that, if the design is constructed with this argument in mind, the credibility of the argument can be increased and the cost of producing it reduced.

Daniel Jackson, Cambridge, MA, USA
Lynette Millett, Washington, D.C., USA
Martyn Thomas, London, U.K.

Reference

1. D. Jackson, M. Thomas, and L.I. Millett, Eds. *Committee on Certifiably Dependable Software Systems. Software for dependable systems: Sufficient evidence?* National Academies Press, 2007.

Author response

I have reread the NRC report and I was mistaken. I must have confused it with something else I read around the time (probably related to Agile). I am very sorry for my mistake. I should have checked before my column appeared.

Nancy Leveson, Cambridge, MA, USA

Sustainable Charge

In his February column, Vinton G. Cerf called for more durability from manufacturers and cited Tesla as an example. I would like to suggest that Tesla and other makers of electric cars include with every purchase a garage battery of the same capacity as that in the automobile. The home battery would have suitable programmable switching to enable charging at steady sustainable rate, for example, from renewable resources or overnight, without the wholesale need to strengthen the domestic power grid. This would allow instant vehicle recharging when the driver returns home.

W.B. Langdon, London, U.K.

Response from the Editor-in-Chief

The temporal misalignment of renewable electricity generation with load (use) is indeed a growing challenge. In fact, in grids with solar-dominated renewables, such as California, “time of use” rates already favor consumption from morning to mid-afternoon—when an electric car might well be at work, rather than at home! More charging stations at business offices would be a much cheaper solution than redundant batteries for electric cars.

Andrew A. Chien, Chicago, IL, USA

Bleating in Computing Machinery

I have been a member of the Association for Computing Machinery for over 60 years. Over these three-score years, I have spent a great deal of time adjusting to the machinery in vogue. We first worked hands-on and later used time sharing and a score of other fads. Now we are somewhere in the clouds.

The time has come to call a general meeting of the team. We are not an association of computing machinery; we are an association of individuals concerned about computing. We may have different interests and experience. We work in varied environ-

ments. We research many interesting areas of computation. For almost 70 years, ACM has held us in good stead. However, technology has now overwhelmed us. It is and for several years has been moving too fast for all of us.

With the Coronavirus Revolution we now have to work in an environment that is often unreliable, lonely, and depressing.

In a very bleating voice, I cry out for change. We should be called ACP—the Association of Computing Professionals. Let us put it to a vote; ‘STAY WITH ACM or MOVE TO ACP.’

Donald F. Costello, Lincoln, NE, USA

Response from the Editor-in-Chief

It would be difficult to make the argument that ACM isn’t well beyond “computing machinery,” and I like ACP—any other good suggestions? This sounds like just the thing that should be put to ACM Council, and then to a vote by the membership! We have just missed this year’s cycle, so plenty of lead time to develop this for next year.

Andrew A. Chien, Chicago, IL, USA

From the Editor-in-Chief, continued from page 5.

References

1. AI for Climate Change, 2020; <https://www.climatechange.ai/>
2. Carson, R. *Silent Spring*. Houghton-Mifflin, 1962. ISBN-0618249060.
3. Chien, A.A. Owning computing’s environmental impact. *Commun. ACM* 62, 3 (Mar. 2019).
4. Hao, K. Training a single AI model can emit as much carbon as five cars in their lifetimes: Deep learning has a terrible carbon footprint. *Technology Review* (June 6, 2019).
5. Masanet, E., Shehabi, A., Lei, N., Smith, S. and Koomey, J. Recalibrating global data center energy-use estimates. *Science* 367, 6481 (2020) 984–986.
6. Radovanovic, A. Our data centers now work harder when the sun shines and wind blows. Google Data Centers and Infrastructure blog (Apr. 22, 2020); <https://bit.ly/2SeEKbE>
7. Rogan, W. J. and Chen, A. Health risks and benefits of bis (4-chlorophenyl)-1, 1, 1-trichloroethane (DDT). *The Lancet* 366, 9487 (2005), 763–773.
8. Yang, F. and Chien, A. Large-scale and extreme-scale computing with stranded green power: Opportunities and costs. *IEEE Trans. Parallel and Distributed Systems* 29, 5 (May 1, 2018), 1103–1116.



Vinton G. Cerf

DOI:10.1145/3397262

Implications of the COVID-19 Pandemic

IT IS LATE April 2020 as I write this column and it will appear in early June, at which time I expect we may only just be emerging from the “shelter at home” restrictions imposed by the governors of most of the States in the Union. I contracted the disease in mid-March and spent about three weeks recovering. While my symptoms were mild, the virus left me low on energy and it took time to get back to normal levels. It could have been much worse, and it is for many people, some of whom do not survive. I had occasion to think and speculate about what our profession has to contribute in the response to this worldwide pandemic.


Computing has a lot to offer, not least is what I will call “computational-x” for many values of “x.” Biology, astronomy, physics, linguistics, chemistry, and cosmology are all changed and in some ways enhanced by the prefix. With sufficient computing power and the use of new tools such as machine learning (ML) for deep, multilayer neural networks, we are able to analyze data effectively in ways not feasible earlier. I read one report that ML helped to identify a small number of molecules that might interfere with a virus’ ability to “dock” with a cell and inject its RNA or DNA to take over the cell’s genetic machinery and reproduce the virus. The search space had billions of possible small molecules.^a There are many reports from sources around the world that are making use of supercomputing capacity, neural networks, specialized processing (such as Graphical Processing Units and Google’s Tensor Flow processors)

and text analysis using ML to uncover possible responses to the novel SARS-COV-2 virus.

As a member of the U.S. Department of Energy’s Advanced Scientific Computing Advisory Committee, I was privileged to listen to many reports of discoveries related to the pandemic that illustrated for me that we are only just beginning to take advantage of new algorithms and large-scale data analytics to shed light on difficult and complex problems, not the least of which are the mysteries and complexities of bio-eco-system interactions. The SARS-COV-2 virus appears to have originated in bats (as have several other corona viruses) but reached humans through other intermediary hosts. Since this virus is RNA-based, it has a higher likelihood of mutation simply because the RNA replication process is less stable than the counterpart for DNA. Mutation can frustrate the intent of a vaccine that might be based in part on particular aspects of the virus, such as the docking spicule used by the virus to bind to a cell receptor in

preparation for injecting its RNA payload into a victim cell.

Turning to our global response to the pandemic introduced by COVID-19, it has become apparent that so-called social distancing has been an important element of response that is intended to limit the spread of the airborne disease. Wearing masks to keep from spreading the virus, washing hands frequently, avoiding touching the face, eyes, nose and mouth, staying six feet or more apart, banning group gatherings that would trigger super-spreading of the disease, shutting down restaurants, sports events, movie theaters, personal service businesses (such as hair salons, counter services at banks) and other activities that would contribute to viral spread has been helpful medically but disastrous from an economic perspective. The Internet and World Wide Web, videoconferencing, collaboration tools, email, and social media are now primary avenues for business, social interaction, and entertainment. It is astonishing the system has been able to absorb so much new demand but indicative of the capacity investments made in part to support streaming video.

I wonder what aspects of our daily working lives will be permanently altered, post-COVID-19. Will we no longer shake hands? Will the travel, hotel, and restaurant businesses be permanently altered? Will working and schooling from home become more common and even preferred? There is no doubt in my mind that our profession and the products it creates will have a prominent role in shaping our post-COVID-19 society. That’s an awesome responsibility! 

The Internet and Web, videoconferencing, collaboration tools, email, and social media are now primary avenues for business, social interaction, and entertainment.

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

Copyright held by author.

a <https://www.genengnews.com/insights/trends-for-2020/artificial-intelligence-is-helping-biotech-get-real/>

The *Communications* Web site, <http://cacm.acm.org>, features more than a dozen bloggers in the BLOG@CACM community. In each issue of *Communications*, we'll publish selected posts or excerpts.



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

DOI:10.1145/3392370

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

Detecting/Preventing Infections, and Moving Instruction Online

Terrence DeFranco suggests the Internet of Things could be keeping us safer, and Jeremy Roschelle airs issues related to online instruction.



Terrence DeFranco
The Internet of Things and the Future of Virus Detection and Prevention

<https://bit.ly/39Dcs0b>

March 17, 2020

As of March 17th, 2020, more than 188,297 people have been infected with COVID-19. How can technology aid in curtailing the spread of infectious diseases that have the potential to create panic and infirm thousands of people? The Internet of Things (IoT), a network of interconnected systems and advances in data analytics, artificial intelligence, and connectivity, can help by providing an early warning system to curb the spread of infectious diseases.

China's efforts to control the coronavirus have meant many residents stayed at home and factories just shut down. That had an unintended effect: less air pollution. Cleaner air can improve public health, maybe even save lives. Indoor air pollution has been ranked among the top five environmental risks to public health, and the advent of COVID-19 puts a spotlight on

the need to ensure we can remove volatile indoor contaminants.

Leveraging IoT in our indoor environments could help prevent highly infectious diseases from spreading rapidly in today's global world. Typically, efforts to improve the environment tend to focus on the outdoors. According to the U.S. Environmental Protection Agency (EPA), indoor air in homes and buildings may be more polluted than outdoor air—a serious issue since people spend, on average, 90% of their time indoors. With the advent of COVID-19, we are mandated to remain indoors, further highlighting the need to ensure that our indoor air quality is good.

LEED (Leadership in Energy and Environmental Design) is an internationally recognized green building certification system aimed at improving performance across all the metrics that matter most: energy savings, water efficiency, CO₂ emissions reduction, improved indoor environmental quality, and stewardship of resources. LEED places emphasis on indoor environmental quality because poor air quality negatively impacts occupant health and safety.

According to the EPA, poor indoor air quality affects 33% to 50% of commercial buildings in the U.S., sometimes causing “sick building” syndrome, which causes a wide variety of symptoms. Sick building syndrome could be caused by inadequate ventilation (the introduction and distribution of clean air); biological contaminants such as molds, bacteria, and viruses; or chemical contaminants like volatile organic compounds or formaldehyde.

To create a healthy and comfortable indoor environment, business leaders should implement an indoor air quality management program that both controls contaminants and ensures adequate ventilation. The foundation of a good indoor air quality program is an Internet of Things (IoT) platform that monitors the air continuously to detect the presence of common pollutants and helps maintain the appropriate volume of fresh air. It also provides actionable data you can use to address existing issues and document your improvement over time.

In terms of looking future forward, IoT may be leveraged to more quickly detect infection or deliver care more efficiently. These scenarios may include a network of advanced indoor sensor technology such as virus-detection sensors to detect and remediate the presence of toxins. Additionally, home telemedicine technology using wearable technology sensors can monitor health conditions of affected patients and keep them closely linked via this medium with their health care provider rather than physical visits, which would have the effect of reducing the burden to the healthcare

system and reduce human interaction. Affected patients can be closely monitored via telemedicine to reduce the burden on healthcare facilities.



Jeremy Roschelle
Powerful Online Learning is a Distributed System

<https://bit.ly/2UzlwPI>
 March 25, 2020

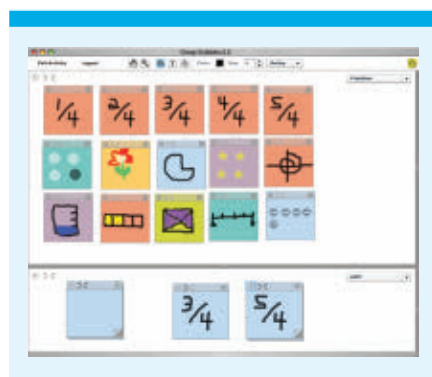
In the midst of a pandemic, universities are now rapidly shifting to online and remote learning. I will suggest a metaphor for powerful online learning. This metaphor should resonate with our backgrounds as computer scientists and it also fits core principles of the learning sciences. As this is my inaugural blog in this venue, let introduce myself: I'm a learning scientist who works on improving STEM and CS learning with technology. I also have a CS degree from MIT. I like to connect the computer science and learning sciences parts of my brain.

Let's start with what good online learning is not: it is not moving lectures online and continuing business as usual. Stated in computational terms, it is not about a central computational process (the instructor) doling out the same unit of work to hundreds or thousands of processors (the students) and then providing an authoritative rating of each processor's individual work.

Let's consider this alternative metaphor: designing powerful online learning as designing an effective distributed processing system. Stated in computational terms, it is about coordinating the active, engaged local work of separate processors (the students) toward a common goal (a community with greater shared understanding of the subject matter). It's about organizing the connectivity in the system so processors (students) give each other help and feedback, and thus converge toward better learning.

Why *active, engaged work*? Because the direct cause of learning is not instruction, but rather the active, engaged effort a student commits to making sense of a challenging concept in their zone of proximal development (their area of growth).

Why *coordinating*? Because learning sciences has collected a large body of evidence that when students have to elaborate their knowledge for another student



(or coordinate on a shared knowledge product), they learn more. Students also learn more when they provide feedback to each other with explanations, not just answers or "do it like this." Collaborative learning is very powerful when roles are structured so that students are required to actively elaborate, coordinate, and give feedback. Together, active learning and collaborative learning expand the "zone" for learning.

In a project called GroupScribbles, we took the distributed systems metaphor literally. We created a "blackboard architecture" for a classroom. In GroupScribbles, individual processors (students) take jobs (intellectual work) from a shared space, and post partial results (learning) back to the shared space. The taking and posting of work is mediated by virtual sticky notes that students move between their own private space and a public space.

It's much simpler than it sounds. In the GroupScribbles system pictured above, an elementary school teacher created initial sticky notes that asked for representation of a particular fraction. Students selected a fraction to work on by taking a blank note from the shared space to their private space (the atomic "take" operation in a blackboard architecture). They posted back a drawing of the fraction (the put operation in a blackboard architecture).

Then small groups were asked to take a collection of stickies for one fraction to a separate room. In the room, they elaborated an explanation for each representation, and then chose a single best explanation to share with the whole class. Finally, the teacher led a discussion of the smaller set of student work that was shared back. This is a powerful learning activity because students are working hard to elabo-

rate, coordinate, and get feedback on what they understand across multiple modes: as individuals, in small groups, and in larger discussions.

Notice who is doing the work that drives learning in this distributed system: the students. Notice the instructor's role: to find a creative way to make every student think hard about a different facet of the same problem, to use small groups to encourage knowledge coordination and feedback among students, and then to regulate the flow of information back to the central blackboard, where a teacher can add their unique value in further discussing some of the selected work of the students.

This metaphor can be fruitfully extended:

Think of students as a heterogeneous set of differently-abled processors. How can an instructor create a coordinated system so that every processor works as hard as they can? How can the cognitive diversity of students become an asset that drives learning to become deeper?

An instructor might consider how students can give peer feedback; how students can self-select easier or harder challenges; how students can promote issues or questions from small groups to bigger groups; how shared understanding of a concept can be made stronger by comparing and contrasting different student elaborations of the same concept. In an elegantly architected distributed system, the connectivity and activity of individual processors (students) overcomes the limited bandwidth of the central processor to give attention to each individual unit.

The **Distributed Systems Metaphor for Online Learning** suggests that an instructor should fully engage and connect the students (the processing nodes) to maximize their active effort to elaborate, coordinate, and give each other constructive feedback, with a collective goal in mind (that is, learning more deeply by harnessing cognitive diversity).

Terrence DeFranco is CEO of IOTA Communications, the first IoT network employing FCC licensed spectrum, based in Allentown, PA, USA. **Jeremy Roschelle** is Executive Director of Learning Sciences Research at Digital Promise and a Fellow of the International Society of the Learning Sciences, Bloomington, IN, USA.

© 2020 ACM 0001-0782/20/6 \$15.00

An Animating Spirit

ACM A.M. Turing Award recipients, Ed Catmull and Pat Hanrahan, overcame industry indifference to found Pixar and put their computer graphics expertise to work.

WHEN ED CATMULL earned his Ph.D. from the University of Utah in 1974, with a thesis on three-dimensional (3D) computer graphics, he applied for jobs in academia. He did not get a single one. His academic interests eventually led him to co-found Pixar Animation Studios, create the breakthrough 1995 animated film *Toy Story*, and receive the Association for Computing Machinery's 2019 A.M. Turing Award, along with his colleague Pat Hanrahan.

At the time, nobody thought computer graphics were worth much of anything. Catmull, 74, remembers trying to convince Hollywood studio executives that the technology he was working on would be important to the future of movies. "We were so irrelevant that we couldn't even get in the door to explain anything whatsoever," he says.

Hanrahan, 66, now a professor at Stanford University, agrees. "At that time, computers were about number crunching, sorting, databases, and all sorts of cool topics," Hanrahan says. Graphics were viewed as too playful for such a utilitarian machine. "They didn't see this as something you would interact with like we do today."

This is only the second time the Turing Award has been awarded for computer graphics. The first, in 1988, went to Ivan Sutherland, who taught the first computer course Catmull took at Utah. At Utah, working in a program funded by the federal Advanced Projects Research Agency, Catmull tackled the problem of how to create images on a curved surface. Computer images were made out of a series of flat polygons, and he had to figure out how to bend them, how to generate a better type of curved surface, and how to know which parts of the 3D image would be visible. The solution to that last problem was

"We were so irrelevant that we couldn't even get in the door to explain anything whatsoever," Catmull recalls.

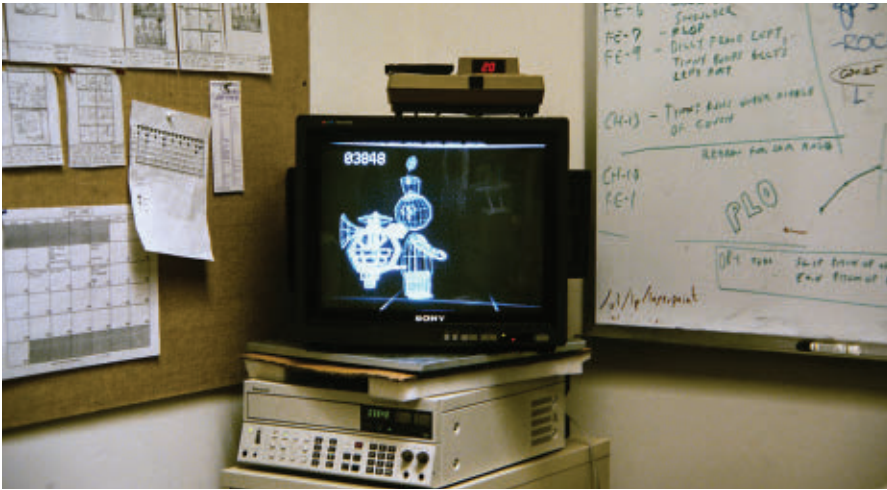
z-buffer, which calculates the depth, or z-axis, coordinates of the image.

After finishing his doctorate, Catmull initially took a job with a small company in Massachusetts that was working on computer-aided design, but within a few months was recruited by the New York Institute of Technology (NYIT) to start a computer graphics laboratory. In 1979, Hollywood finally noticed, and *Star Wars* creator George Lucas hired him to work at LucasFilm developing computer graphics. The graphics division became Pixar in 1986 when Apple founder Steve Jobs bought it; he later sold it to Disney.

Hanrahan originally studied nuclear physics, and after a stint working with Catmull in the graphics lab at NYIT, earned a Ph.D. in biophysics from the University of Wisconsin in 1985. He was trying to map the nervous system of a nematode, a parasitic worm, which led him into the field of data visualization and, ultimately, to computer graphics.

He joined Pixar shortly after Jobs bought it. The studio was trying to create an industry standard for handling graphics, and Hanrahan was the architect of the system, leading a collaboration of 19 different companies. His software to generate 3D images and determine their shading and texture eventually became





Tin Toy, the first 3D-computer-animated short to win an Academy Award for Best Animated Short Film (in 1989), was also the first film to use RenderMan.

RenderMan, a popular image rendering program that won the two a science and engineering Academy Award in 1993. RenderMan was used to create the silver, liquid T-1000 Terminator from the future in *Terminator 2*, and the 3D waltz scene in *Beauty and the Beast*, which led to Disney commissioning the first *Toy Story* movie. The two have been awarded additional Oscars for their achievements with computer animation.

Hanrahan also developed a language for describing the shading of the 3D images to show how they would look as lighting changed, and made it accessible to the artists creating the scenes. Engineers at LucasFilm and Pixar also tackled other challenges, such as reducing the image blur produced by motion. While the state of the art had been to create images with 40,000 polygons, they wanted their systems to handle 80 million, so bits of animation would appear realistic enough to be used in live-action films.

Hanrahan says there were hundreds of innovations and thousands of people involved in improving animation. Whereas SIGGRAPH, the annual Special Interest Group on Computer Graphics and Interactive Techniques conference, was a niche area when first launched in 1974, by the mid-1990s Hanrahan says it was probably the largest ACM conference. The goal set by pioneers in the field of actually making a computer animated movie, and not merely developing technology, fired people's imaginations. "It just became this catalytic intellectual endeavor, like this moonshot kind of thing, and all

these people started improving the technology," Hanrahan says.

It was a goal he did not expect would be achieved quickly, perhaps not even in his lifetime. Hanrahan believed a movie would be a more complex version of a Turing test for artificial intelligence. Instead of creating a chatbot that an interrogator could not distinguish from a human, he imagined creating an artificial world that people could not tell from the real world. "I thought we'd have to solve the AI problem in order to actually make a full-length movie," he says. "I was wrong."

Catmull was more optimistic. He was already imagining a computer animated movie back in 1974. "I figured at the time it would take 10 years, because I knew we didn't have nearly enough processing power and we hadn't solved nearly enough problems," he says. "I was wrong, incidentally. It took 20 years, not 10."

In the early 1980s, the researchers calculated how much of the then-available computing power it would take to make a full-length computer-generated film. The answer: 100 Cray supercomputers, which at the time cost about \$10 million apiece. Catmull recalls framing his approach at that time as, "Let's focus on algorithms and making the pictures look good and various techniques, and at some point the processing power will catch up with what we need, which happened around 1990."

Before giving up on hardware, LucasFilm built the Pixar image computer to composite two-dimensional images together at a rate far faster than anything then available. That technology would

up spurring the development of volumetric medical images from MRI and CT scans.

The graphics algorithms also created a feedback loop with chip designers at companies such as Nvidia. The engineers would learn about new algorithms from SIGGRAPH conferences and incorporate them into their chips, allowing them to make high-powered graphical processing units (GPUs) for the gaming industry. They provided graphics researchers with more powerful equipment they could use to improve their algorithms, which led to further improvements in the chips, until GPUs became so powerful they spilled into other areas. "Deep Learning and Neural Networks all of a sudden had enough power, so this field then exploded into a whole new field," Catmull says.

These days Catmull is semi-retired, though he does some consulting work and is revising his 2014 book on creativity. He says the secret of a successful career is choosing a problem that is neither so easy as to bore you nor so impossible it cannot be achieved. "Taking stuff where you kind of get what it is but it's really hard, that's where you want to be," he says.

Hanrahan continues to teach at Stanford, and his favorite course teaches freshmen how to build an operating system from the ground up. His advice: "Just find that thing that really interests you and do the hard work to become skilled in it."

Both men say they will likely donate their shares of the \$1-million Turing Award prize money to charity. Both also say winning a Turing Award was more exciting than winning an Oscar, because it comes from their own community.

As for which *Toy Story* they prefer, Hanrahan says he likes the second one best. For Catmull, the question is like asking which is his favorite child. "I can't really pick one," he says, "because each had a special meaning." □

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

© 2020 ACM 0001-0782/20/6 \$15.00



Watch the the Turing recipients discuss their work in the exclusive *Communications* video. <https://cacm.acm.org/videos/2019-acm-turing-award>

Leveraging Unlabeled Data

Deep learning looks for better pretexts.

DESPITE THE RAPID advances it has made it over the past decade, deep learning presents many industrial users with problems when they try to implement the technology, issues that the Internet giants have worked around through brute force.

“The challenge that today’s systems face is the amount of data they need for training,” says Tim Ensor, head of artificial intelligence (AI) at U.K.-based technology company Cambridge Consultants. “On top of that, it needs to be structured data.”

Most of the commercial applications and algorithm benchmarks used to test deep neural networks (DNNs) consume copious quantities of labeled data; for example, images or pieces of text that have already been tagged in some way by a human to indicate what the sample represents.

The Internet giants, who have collected the most data for use in training deep learning systems, have often resorted to crowdsourcing measures such as asking people to prove they are human during logins by identifying objects in a collection of images, or simply buying manual labor through services such as Amazon’s Mechanical Turk. However, this is not an approach that works outside a few select domains, such as image recognition.

Holger Hoos, professor of machine learning at Leiden University in the Netherlands, says, “Often we don’t know what the data is about. We have a lot of data that isn’t labeled, and it can be very expensive to label. There is a long way to go before we can make good use of a lot of the data that we have.”

To attack a wider range of applications beyond image classification and speech recognition and push deep learning into medicine, industrial control, and sensor analysis, users want to be able to use what Facebook’s chief AI

“The problem I see now is that supervising with high-level concepts like ‘door’ or ‘airplane’ before the computer even knows what an object is simply invites disaster.”

scientist Yann LeCun has tagged the “dark matter of AI”: unlabeled data.

In parallel with those working in academia, technology companies such as Cambridge Consultants have investigated a number of approaches to the problem. Ensor sees the use of synthetic data as fruitful, using as one example a system built by his company to design bridges and control robot arms that is trained using simulations of the real world, based on calculations made by the modeling software to identify strong and weak structures as the DNN makes design choices.

Although simulation can create useful data for a back-end DNN to learn from with little input from manually labeled data, some researchers in the field believe these systems should do much better at handling unlabeled inputs without the help of synthetic data. Their hope is that by focusing on patterns in the core data, deep learning can approach problems more intuitively. Some see the current reliance on labeled data as even being counterproductive to the development of effective AI.

Alexei Efros, a professor in the Electrical Engineering and Computer Sciences department of the University of California, Berkeley, cites a problem with the current approach to handling images: they generally contain far more information than is implied by the relatively simple tags applied by humans for training purposes. “The problem I see now is that supervising with high-level concepts like ‘door’ or ‘airplane’ before the computer even knows what an object is simply invites disaster.”

What Efros wants is for the AI systems to capture the information that humans remember after they have seen a picture. “Do we have a photographic memory? No, we don’t. But we have a cool embedding that captures a lot of non-linguistic information,” Efros says. “We need to get away from semantics and force the computer to represent more of what is actually in the image.”

Leon Gatys and colleagues at the Technical University of Tübingen, Germany, in 2017 showed an example of the problem with the way in which deep learning models are trained today. A DNN will just as readily identify random patches of fur-like texture as a dog as a picture of the animal itself. “The networks are lazy,” says Efros. “They do the minimum work required to get the reward or minimize the loss. Typically, local image statistics, the texture, are easier for the network to compute than long-range structure. We need to make our computers work harder to understand what they see.”

Natural language processing (NLP) is an area where the performance of neural networks has been improved by forcing them to put much greater emphasis on the structure of the data they process and create embeddings similar to those used by humans to capture the information they learn.

For some time, NLP has relied on a clustering, a form of machine learning where the algorithm finds structure for itself, to try to determine which words have similar meanings. The result is that rather than analyzing text as disconnected words, a neural network provided with additional information by the clustering receives a head start. The most common technique is to give each unique word in the dictionary a value, in the form of a vector. Translation software takes advantage of this, thanks to the tendency of words in different languages tending to map to similar positions in the vector space.

The problem with the simpler clustering methods is that words often have multiple meanings. ‘Bank’, for example, can act as a verb or a noun which, in turn, can refer to the bank of a river or a financial bank. Rather than use simple clustering, a new generation of systems that first appeared in 2018 employ DNNs to take much more contextual information into account.

Examples include the Bidirectional Encoder Representations from Transformers (BERT), published by Google, and Carnegie Mellon University’s XLNet. Each uses subtly different algorithms to analyze the texts used for training. BERT, for example, randomly removes words from the input text and forces the model to predict which is the best candidate. XLNet’s developers argue the data corruption implied by BERT’s word masking degrades performance. They instead opted for a system that uses the relative positions of words in the training texts to determine how they relate to each other.

Though they are sophisticated DNNs in their own right, systems like BERT and XLNet are not standalone NLP engines, they simply provide richer information for use by downstream deep-learning systems that has paid off according to benchmarks such as GLUE that were designed to measure the capabilities of NLP algorithms. Late last year, the developers of that suite of tests revealed at the NeurIPS conference they had created a battery of more stringent tests, which they tagged SuperGLUE, to handle AI that take advantage of BERT-like pretraining.

Researchers working on image, video, and audio recognition see

DNNs that perform pretext tasks that follow in the footsteps of those used in NLP as being important to finding inherent structure in the data their own systems analyze. They believe the pretext tasks should force machine-learning models to do a better job of deconstructing data.

Several years ago, inspired by the clustering used in NLP, researchers working with Efron had a DNN pipeline learn the spatial relationships between patches cut out of images. For example, given pictures of animals from which random patches were removed, the DNN might begin to learn how the ears are placed relative to the eyes, nose, and mouth by determining which arrangement of patches most closely matches the completed images.

Even when trying to force it to recognize the layout of objects within an image, the Berkeley team ran across the DNN’s propensity to cheat. Rather than focus on high-level features, they found the network homed in on color aberrations from the lens used to take the photographs that were undetectable to the naked eye. This forced them to process the images to remove some of the color information. “It stumped us for a long time before we figured out what was going on,” Efron says.

Work on the kind of patch embedding performed by Efron’s team has led to networks that are able to add color to monochrome images effectively and to fill in blank spaces in images using material the DNN has inferred from similar pictures. Similar work on pre-text training at Facebook AI Research in France used a combination of clustering and training on rotated images to improve their image classifier’s ability to work on very large collections of images. They demonstrated their DeeperCluster engine using a massive archive of unlabeled photographs on the Flickr website. After processing using the pre-text task, a second phase trains using more conventional techniques with the help of labeled data so the engine can categorize the images in ways humans understand.

A lingering issue for systems in image recognition and non-NLP tasks is how to develop pre-text tasks that are truly effective at forcing the DNNs to

understand the short- and long-range structure within the data. Efron says pretext task construction remains more an art than a science at the moment.

Hoos says approaches based on synthetic data and pretraining can help better understand, for example, the relationships between objects in images. But he questions whether deep learning will hold all the answers. In image analysis, he says, it is hard to find a technology remotely competitive with deep learning. “But I would not bet everything on this approach.”

Further work, Hoos believes, may show that alternatives to deep learning will be more fruitful in areas such as automation and robot control, situations where understanding the connections between objects in the field of view is critical. “If deep learning is used on its own in the way it exists today, you can’t do cutting-edge robotics. Also, we often want explainability in the models, which is often difficult with deep learning. We need to understand what the systems are doing and how they made a decision. We should keep pushing on semi-supervised and deep learning, but at the same time, we should not neglect other methods.” **■**

Further Reading

Gatys, L., Ecker, A.S., Bethge, M.
Texture and Art with Deep Neural Networks
Current Opinions in Neurobiology, Volume 46, pages 178-186 (2017)

Doersch, C., Gupta, A., Efron, A.A.
Unsupervised Visual Representation Learning by Context Prediction
Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV 2015), pages 1422-1430, <https://arxiv.org/abs/1505.05192>

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V.
XLNet: Generalized Autoregressive Pretraining for Language Understanding
Advances in Neural Information Processing Systems 32 (NIPS 2019), <https://arxiv.org/abs/1906.08237>

Caron, M., Bojanowski, P., Maria, J., Joplin, A.
Unsupervised Pre-Training of Image Features on Non-Curated Data
Proceedings of the 2019 IEEE International Conference on Computer Vision (ICCV '19), Pages 1422-1430

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

Seeing Through Walls

Artificial intelligence makes sense of radio signals to understand what someone in another room is doing.

MACHINE VISION COUPLED with artificial intelligence (AI) has made great strides toward letting computers understand images. Thanks to deep learning, which processes information in a way analogous to the human brain, machine vision is doing everything from keeping self-driving cars on the right track to improving cancer diagnosis by examining biopsy slides or x-ray images. Now some researchers are going beyond what the human eye or a camera lens can see, using machine learning to watch what people are doing on the other side of a wall.

The technique relies on low-power radio frequency (RF) signals, which reflect off living tissue and metal but pass easily through wooden or plaster interior walls. AI can decipher those signals, not only to detect the presence of people, but also to see how they are moving, and even to predict the activity they are engaged in, from talking on a phone to brushing their teeth. With RF signals, “they can see in the dark. They can see through walls or furniture,” says Tianhong Li, a Ph.D. student in the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology (MIT). He and fellow graduate student Lijie Fan helped develop a system to measure movement and, from that, to identify specific actions. “Our goal is to understand what people are doing,” Li says.

Such an understanding could come in handy for, say, monitoring elderly residents of assisted living facilities to see if they are having difficulty performing the tasks of daily living, or to detect if they have fallen. It could also be used to create “smart environments,” in which automated devices turn on lights or heat in a home or an office. A police force might use such a system to monitor the activity of a suspected terrorist or an armed robber.



RF-Pose uses wireless signals to monitor people's movements on the other side of a wall.

In living situations, one advantage of monitoring activity through walls with RF signals is that they are unable to resolve faces or see what a person is wearing, for instance, so they could afford more of a sense of privacy than studding the home with cameras, says Dina Katabi, the MIT professor leading the research. Another is that it does not require people to wear monitoring devices they might forget or be uncomfortable with; they just move through their homes as they normally would.

The MIT system uses a radio transmitter operating at between 5.4 GHz and 7.2GHz, at power levels 1,000 times lower than a Wi-Fi signal, so it should not cause interference. The RF transmissions bounce strongly off people because of all the water content of our bodies, but they also bounce off other objects in the environment to varying degrees, depending on the composition of the object. “You get this mass of reflections, signals bouncing off everything,” Katabi says.

So the first step is to teach the computer to identify which signals are coming from people. The team does this by recording a scene in both visible light and RF signals, and using the visual image to label the humans in the

training data for a convolutional neural network (CNN), a type of deep learning algorithm that assigns weights to different aspects of an image. The signals also contain spatial information, because it takes a longer time for a signal to travel a longer distance. The CNN can capture that information and use it to separate two or more people in the same vicinity, although it can lead to errors if the people are very close or hugging, Katabi says.

Down to the Bones

The researchers also want to identify the actions people are taking based on how they move. To do that, the system takes the intermediate step of rendering the signals from the people into skeletons, simplified versions of human bodies that reduce them to essentially stick figures consisting of a head and limbs. Based on the relative positions of the lines in the stick figures, the computer can identify various actions—sitting down, bending to pick something up, waving, drinking, talking on the phone, and so on.

The team built on previous work done using AI to derive actions from video images. None of the existing action detection datasets, however, in-

cluded RF information, so the team had to create its own. They found 30 volunteers, put them in 10 different environments—offices, hallways, and classrooms, for example—and asked them to randomly perform actions from a list of 35 possibilities. That produced 25 hours of data to train and test their computer model.

To provide more data, they derived skeletons from video that did not have any associated RF signals. Li says those skeletons were slightly different from those derived from the RF signals, so they had to make some adjustments. This approach, however, allowed the researchers to incorporate other information, such as a dataset called PKU-MMD, which contains nearly 20,000 actions from 51 categories performed by 66 people.

Movement in Space

Katabi is not the only researcher using RF to see through walls. Neal Patwari, a professor of computer science and engineering at Washington University in St. Louis, MO, has been working on RF sensing for years, although his work back in 2009 did not involve machine learning. His basic system uses both a radio transmitter and receiver. Bodies in the vicinity alter the strength of the signal being picked up by the receiver. A person directly crossing a line between the two blocks the signal strongly, someone nearby blocks it somewhat less, and when a person is far away, a signal scattering off their body changes the received signal a bit.

Patwari uses machine learning to teach the computer what particular blend of signals goes with a specific action. For instance, a particular set of measurements from the kitchen might mean someone was cooking. “You could have an algorithm that learned what the features are that correspond to cooking. And then I could know, for somebody living home alone, that they’re cooking at this time on this day and make sure that they’re not veering from their regular schedule too much,” he says. Such a system could be useful to monitor a person in the early stages of dementia, or who was at known risk for depression.

One issue, though, is that the measurements could change with changes

in the environment, Patwari says. For instance, if a person had just done the grocery shopping, opening a refrigerator full of food might generate one set of RF signals. A few days later, when there was more empty space in the refrigerator, the signals could look different. The training data goes stale after a while.

Patwari currently is developing a system that uses RF signals to monitor a patient’s breathing through the walls of a psychiatric hospital, for instance. Many psychiatric hospitals have a policy to go into a patient’s room every 15 minutes at night to make sure they are sleeping normally and not trying to harm themselves. Such checks, of course, can wake the patients and interfere with them getting the sleep they need. An RF system can detect the movement of a patient’s chest and see that they are sleeping normally. Because it sees through walls, all the equipment and wires are outside the room, so the patient cannot damage them or use them to harm him/herself.


The Doppler Effect

Respiration can be detected by using RF signals as a sort of Doppler radar, seeing how the signal changes as it reflects off something moving back and forth, says Kevin Chetty, a professor in the department of security and crime science at University College London. His focus is on using machine learning to make sense of what he calls passive Wi-Fi, taking advantage of RF signals already in the environment including, potentially, the signals used by 5G cellphones. Any sort of movement in an RF field produces Doppler signals. “There’s a complex amalgamation of forward and back, left and right, different angles to different Doppler components,” Chetty says. “You get micro-Doppler signatures associated with different types of motion.”

Relying on RF signals already in the environment means there is no need for the type of calibration that a system like Patwari’s requires. On the other hand, it has to make do with whatever signals exist, instead of relying on a predictable signal like Katabi’s.

To train his action recognition system, Chetty has volunteers wear suits studded with LEDs, a system similar to that used in creating virtual reality

scenarios. As the person moves, Chetty and his team collect both visual data recorded by cameras, and RF signals. The visual data provides ground truth for training the system to recognize actions based on the micro-Doppler signatures. It can be a complex set of measurements to decipher. For instance, the Doppler signal for the same action can look different depending on the angle from which it’s viewed.

Some RF sensing technology is already in use. Former students of Patwari’s, for instance, formed a company called Xandem that sells sensors that use RF to detect human presence and motion. Systems using machine learning to identify actions, for use in healthcare or smart environments, still require further development. “We’re not there yet, but we’re working on it,” Chetty says. 

Further Reading

Li, T., Fan, L., Zhao, M., Liu, Y., Katabi, D. **Making the Invisible Visible: Action Recognition Through Walls and Occlusions**, 2019, arxiv.org/abs/1909.09300

Al-Husseiny, A., Patwari, N. **Unsupervised Learning of Signal Strength Models for Device-Free Localization**, 2019 **IEEE 20th International Symposium on “A World of Wireless, Mobile and Multimedia Networks”** DOI: 10.1109/WoWMoM.2019.8792970

Li W, Piechocki R, Woodbridge K, Chetty K **WiFi Sensing, Physical Activity, Modified Cross Ambiguity Function, Stand-Alone WiFi Device**, 2019, **IEEE Global Communications Conference**, <https://discovery.ucl.ac.uk/id/eprint/10084427/>

Wang, X, Wang, X., Mao, S. **RF Sensing in the Internet of Things: A General Deep Learning Framework**, 2018, **IEEE Communications Magazine**, DOI 10.1109/MCOM.2018.1701277

Dina Katabi, MIT China Summit <https://www.youtube.com/watch?v=9nVZwLqG6aI>

Researchers Use Wi-Fi to See Through Walls

<https://www.youtube.com/watch?v=fGZzNzYIH0>

AI Senses People Through Walls https://www.youtube.com/watch?time_continue=59&v=HgDdaMy8KNE&feature=emb_logo

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

Hiring from the Autism Spectrum

Companies increasingly are looking to hire people who are on the autism spectrum to fill IT roles.

YEARS AGO, MICHAEL Fieldhouse had a dinner party and friends attended with their young son Andrew, who is autistic, non-verbal, and low-functioning. At one point, Fieldhouse noticed Andrew, who was five- or six-years-old at the time, outside dropping pebbles into an urn in a Japanese garden.

“I was curious about that and I started timing him,” recalls Fieldhouse. “I noted there were perfect intervals between every stone. He did that for at least an hour.”

Fieldhouse had an epiphany that night that “changed my view on talent. We look at so many people’s deficits and not their strengths; what they can’t do versus what they can do.”

That inspired Fieldhouse to begin conducting research around autism-at-work programs. Over the course of about a year, “I interviewed lots of people at companies that had done this, and all of their programs failed.” He found three reasons for those failures, he says: not enough training for managers and coworkers; not enough effort put into changing the work culture; and a lack of focus on sustainable employment.

Those three elements became the backbone of a program Fieldhouse helped build in 2013 at Hewlett-Packard; he and the program moved in 2017 to DXC, which was the result of the spin-off of the Enterprise Service segment of what in 2015 became Hewlett-Packard Enterprise, and its merger with Computer Sciences Corporation. Today, he is social impact practice leader in DXC’s Dandelion Program, which has approximately 120 employees in Australia and New Zealand, and a 92% retention rate.

“I went in with a business case and [upper management] liked the idea,” he recalls. “I pushed it as a talent gain and didn’t sell it as a disability



Analyst Corey Weiss, who was diagnosed with autism as a young boy, working at Mindspark.

program or some kind of inclusion initiative, but as capability uplift.” As a tech company, there were vacancies in a lot of areas, and Fieldhouse focused on areas of need where autistic workers could fill a void, such as analytics, cybersecurity, software testing, and automation work.

With a technology staffing shortage that shows no signs of letting up any time soon, companies like DXC are turning to an often untapped talent pool: individuals with autism, or who are on the autism spectrum.

It’s a win-win for both companies and people on the autism spectrum. These individuals are good candidates for technology jobs because some have “hyper-focused attention to detail and pattern recognition,” says Marcia Scheiner, founder and president of Integrate Autism Employment Advisors, whose tag line is “people with autism are hot hires for AI jobs.”

When companies are looking for people who can do software testing and repetitive tasks, and find bugs and er-

rors, Scheiner will point out that given the way the autistic brain works, they don’t see similarities, “but differences jump out at them right away,” she says. “So people on the autism spectrum will see errors or breaks in a pattern; that’s something they’re really good at.”

Dave Kearon, director of adult services at Autism Speaks, an advocacy group that also helps companies start a neurodiversity program that recognizes and values neurological differences, agrees that people on the autism spectrum “are particularly well-suited to the IT industry.” Many tend to pay attention to detailed requirements and precision in their work, he says, and they also have great visual acuity, and like to perform repetitive tasks.

It’s hard to know how many companies have neurodiversity work programs. Kearon says Autism Speaks has placed individuals at Michael’s arts and crafts stores, and Stanley Black & Decker, in retail, warehousing, stockroom, and manufacturing roles. Scheiner says Autism Employment Advisors

currently are working with four or five companies, including Prudential Insurance, Barclay's, and Shopify.

Besides DXC, several other large enterprises offer neurodiversity programs. The Autism@Work Employer roundtable is comprised of businesses that have been running autism-focused initiatives for at least a year. The group shares best practices in areas like training and onboarding. Participating companies include Cintas, DXC, EY, Fidelity, Ford, Freddie Mac, IBM, JPMorgan Chase, Microsoft, Rising Tide Car Wash, SAP, Spectrum, Travelers, Ultra Testing, and Willis Towers Watson.

The need for these programs is palpable. There are between 70,000 and 110,000 people on the autism spectrum in the U.S., according to Drexel University, mostly young adults and teens who are aging out of school every year at 21, when most states cut their mandated funding.

"There's a growing number of those youths leaving school and entering the job market, and we're nowhere even close to providing job opportunities to meet that need," Kearon says. Even the "best" programs have hired about 200 individuals on the spectrum, he says, while most typically employ around 100 or less.

"We need tens of thousands of programs like that," Kearon says. Right now, companies with neurodiversity programs tend to be clustered in the tech sector and the financial services industry, he says. "We need companies in all industries to start these types of neurodiversity inclusion programs for people with autism spectrum disorder."

Many of these individuals will graduate from two- or four-year colleges, and because of their "extraordinary abilities," they won't need to go through a job placement service, he notes. Once they are employed, however, they will need certain supports and a culture that is accepting of some of their differences.

"It's more than just hiring and placing people," Kearon says. The big need here is to support people and help them develop careers over time. "It's not just the initial placement; we need to change corporate cultures and make sure they're supportive of people who think differently."

There is also a need for small com-

There is a "massive talent pool" of these individuals, many of whom understand patterns and have a proclivity for complex work.

panies to offer neurodiversity programs. One such company, New York City-based Daivergent, was started in December 2017 by Byran Dai, whose brother Brandon, 20, is on the spectrum. As a data scientist, Dai saw a need for workers in artificial intelligence and machine learning.

Echoing Kearon, Dai says his brother will have to leave his alternative educational setting when he turns 21 and loses his funding. "There is no pipeline for someone like him in the tech and data space." More typically, people on the spectrum are underleveraged, he says, and wind up working in custodial or food services roles.

Daivergence has a software platform that seeks to become the bridge between people on the autism spectrum and companies that need to fulfill a data-related task, anything from building AI data sets to identifying talent for analytics, QA, or software testing, Dai says.

There is a "massive talent pool" of these individuals, many of whom understand patterns and have a proclivity for complex work, Dai says. Daivergence's goal is to build a recruiting pipeline to help companies fulfill their neurodiversity hiring needs, he says.

What Employers Should Know

Scheiner, who has an autistic son, says it's important to know that people on the spectrum are a talented, skilled, and untapped resource. However, "it is incumbent upon employers to make sure if they're going to hire diverse individuals, to invest time and resources for their neurodiverse employees to be successful," she says. "This entails understanding what it means to be autistic, and training their managers to be effective communicators."

This population is well-suited for a variety of jobs, says Kearon, "so keep an open mind and a broader perspective on how someone who is neurodiverse can fit in." As people become more familiar with the neurodiverse "movement," he says, they will learn quickly that these individuals "can fit in as your QA tester or engineer or your work within marketing or within your operations team."

Fieldhouse also emphasizes that employers should keep an open mind. "Talent can come from anywhere. We've got a person who was homeless and unemployed, and is now is one of top people working in identity fraud at DXC."

Autism in the Workplace

In 2013, SAP became one of the first large companies to offer an Autism at Work program, with the goal of fostering a more inclusive workplace. Today the company employs over 175 employees with autism in 14 countries across a wide variety of job types, and the program has a 90% retention rate, according to SAP spokesperson Sue Sutton.

The impetus for a neurodiversity program at JPMorgan Chase came about in 2015 on the heels of a conversation between an HR person and a senior executive about the difficulty of finding IT talent, says Anthony Pacilio, vice president and global head of the company's Autism at Work program. Pacilio had seen people on the spectrum succeed in IT jobs at different companies he had worked for, so JPMorgan Chase tapped a local talent sourcing firm to find candidates and started a pilot with four individuals in QA roles, which was the immediate need at the time, Pacilio says. Today, the company's Autism at Work program has grown significantly: it employs 170 people in eight countries and over 40 different job roles.

The four people in the pilot were 48% more productive than other employees, Pacilio says. "That little piece was the springboard for the pilot to become a program. We thought, 'heck, did we just get lucky with this one job role?' So we wanted to prove out the business case and make sure the QA portion was an accurate representation of the talent we were bringing in."

From there, the program brought in individuals to do application support, and they were 90% to 140% more pro-

ductive than other employees, with zero errors within first the six months, far surpassing expectations, Pacilio says.

Like the others, he says people on the autism spectrum are good candidates for tech jobs because of their focus, pattern recognition, and their “innate ability to be logical in seeing what’s on the screen.” Their decision-making and problem-solving skills are “second to none,” he says. “It brings a whole new perspective to the workplace, and I don’t think we’ve seen this type of work ethic in this space before.”

One of the comments Pacilio hears from employees about hiring people with autism is that they never thought they would work next to someone so different from them, and that this has forced them to think differently. Other comments are that it’s refreshing to get a different point of view, and that people tend to hire people just like them, he adds. “Bringing in people who see things differently and solve issues differently opens up everyone’s mind.”

Of course, the program has had its challenges. Initially, they had trouble finding the right managers to oversee people on the autism spectrum, and ensuring everyone engaged with the program has been trained to handle any situations that crop up.

“When things are new, people are nervous, but we’ve done a good job building up training and education,” Pacilio says. The way to quell nervousness is to include people who are neurodiverse and their caregivers in the discussion, so “neurotypicals” learn from them, he adds.

Jesse Collins, 29, is a software engineer at JPMorgan Chase who learned about the Autism at Work program from his wife, who also works at the company. “It struck me as really cool to be at a place where you could be authentically you and take away all the stress I have in terms of trying to fit in with neurotypicals,” Collins says.

Collins’ college degree is in psychology; he previously was a social worker, but said the job became difficult because when a patient would cry, he would freeze. “I couldn’t work in that environment,” he says. “Computers don’t cry ... so working with computers makes sense to me.”

His skills came in handy when Collins was brought into the company as a QA tester for eight months. “I had to

teach myself how to have a conversation, and just because I wasn’t interested in a topic not to transition to something else,” he says.

Although he finds group settings uncomfortable, Collins says social work and psychology taught him how to socialize. “My brain is very strategic, but I need to know what’s going on in a group ... so you learn how to communicate, and I’m able to memorize things very quickly.”

In fact, the tech vendor JPMorgan Chase was using told Pacilio that Collins went to a bookstore to track down a book so he could learn Java the weekend before his assessment. Collins says he also “watched a lot of YouTube videos.”

“He came in that Monday and knocked it out of the park,” Pacilio says.

Collins says the company has a welcoming environment, and it has allowed him to “bring my own skillsets and uniqueness and thrive and really succeed.”

During that all-important weekend, Collins was mindful that while having a “hyper focus” is a benefit, “I also understand my wife may not want to go to bed hearing TED Talks and YouTube videos, so I have balance. That’s part of life, too.” **C**

Further Reading

Autism@Work Roundtable
<http://bit.ly/39SMqaj>

DXC Dandelion Program
<https://digitalcommons.ilr.cornell.edu/dandelionprogram/>

Oesch, T.
Autism at Work: Hiring and Training Employees on the Spectrum, August 19, 2019, Society for Human Resource Management, <http://bit.ly/39VErty>

Kaufman, J.
Mindset Matters: Autism, Design Thinking And Building A Pathway To Employment,” April 12, 2019 *Forbes*, <http://bit.ly/39VErty>

Bruyere, S.M.
All About Skills: Tapping the Power of Neurodiversity, 2017, U.S. Department of Labor blog, <http://bit.ly/2QGyYzr>

Henry, Z.
Changing Employers’ Perceptions, One Autistic Worker at a Time, *Inc.com*, <http://bit.ly/2T4R7sd>

Esther Shein is a freelance technology and business writer based in the Boston area.

© 2020 ACM 0001-0782/20/6 \$15.00

ACM Member News

TEACHING THE COMMUNITY ABOUT COMPUTER SCIENCE



“In the late 1970s, Radio Shack put out its own personal computer, and it just

fascinated me,” recalls Michael Littman, a professor of computer science at Brown University in Providence, RI.

After that early experience, Littman was hooked on computer science. He went on to earn his undergraduate and master’s degrees simultaneously, both in computer science, from Yale University.

After graduation, Littman took a job as a researcher at Bellcore (Bell Communications Research) in Morristown, NJ, for several years. He then briefly attended Carnegie Mellon University, before moving to Rhode Island to finish his Ph.D. at Brown.

After completing his doctorate, Littman joined the faculty of Duke University in North Carolina, where he spent several years before leaving to work for AT&T in Florham Park, NJ. After several years there, he returned to academia as a professor at Rutgers University in New Brunswick, NJ, where he remained for about a decade. In 2012, he joined the faculty at Brown University as a professor, where he has remained ever since.

Littman’s research focuses on examining algorithms for decision-making under uncertainty. His primary interests include reinforcement learning, artificial intelligence (AI), and machine learning.

“Currently, I am excited by the prospect of finding ways of taking what is going on in computing in general, and AI and machine learning as well, to the broader community,” Littman says. “I’m exploring writing and making videos as a way to get the message out to determine how cool research ideas can end up in the heads of many more people.”

—John Delaney

Inside Risks

How to Curtail Oversensing in the Home

Limiting sensitive information leakage via smart-home sensor data.

FUTURE HOMES WILL employ potentially hundreds of Internet of Things (IoT) devices whose sensors may inadvertently leak sensitive information. A previous *Communications* Inside Risks column (“The Future of the Internet of Things,” Feb. 2017) discusses how the expected scale of the IoT introduces threats that require considerations and mitigations.² Future homes are an IoT hotspot that will be particularly at risk. Sensitive information such as passwords, identification, and financial transactions are abundant in the home—as are sensor systems such as digital assistants, smartphones, and interactive home appliances that may unintentionally capture this sensitive information. IoT device manufacturers should employ sensor permissioning systems to limit applications access to only sensor data required for operation, reducing the risk that malicious applications may gain sensitive information. For example, a simple notepad application should not have microphone access.

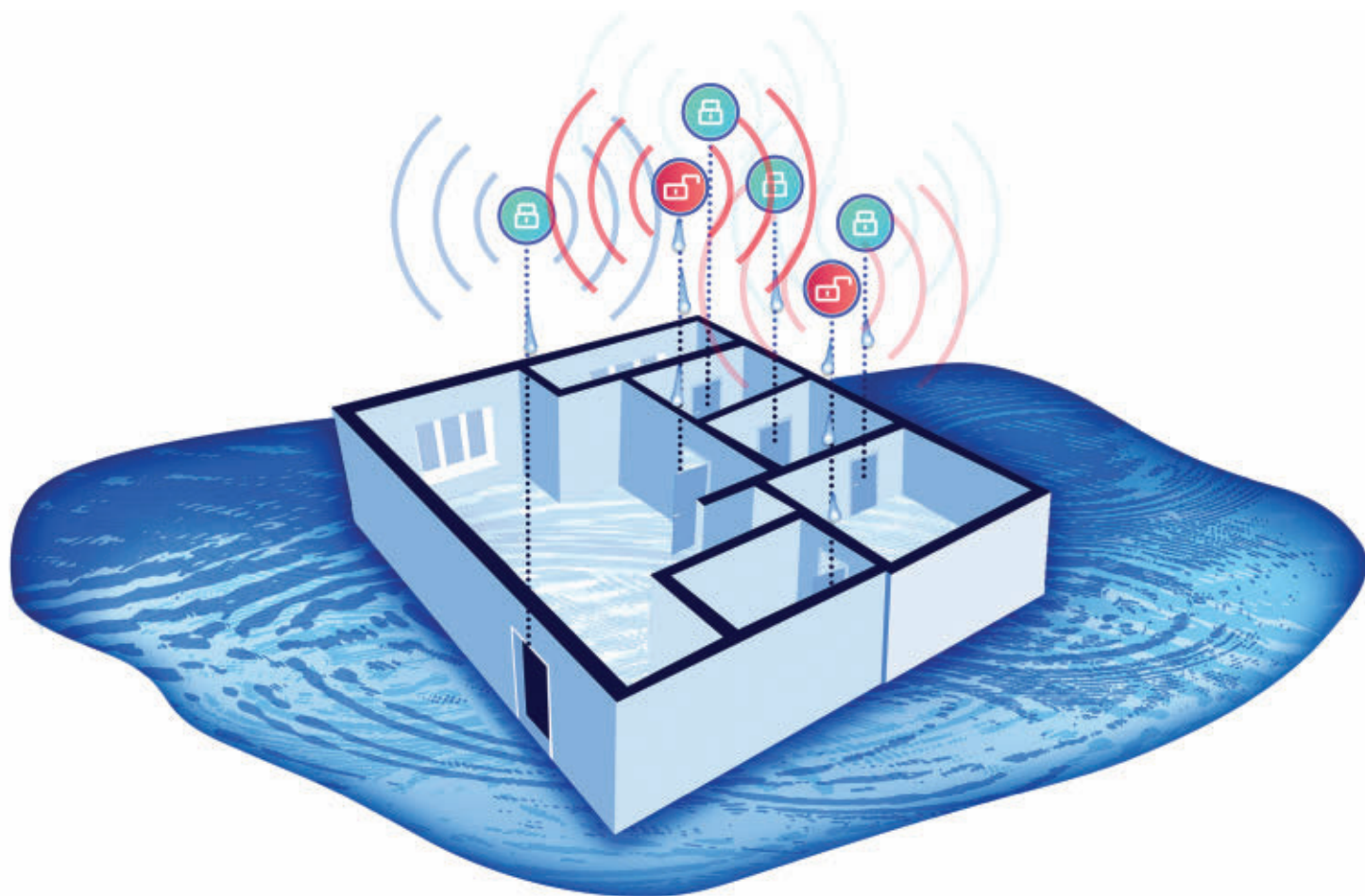
However, even if this least-privilege approach was enacted across all IoT systems (a difficult task), sensor systems still gather much more information than intended or required by an application—for example, how motion sensors can capture nearby sounds, including words and keystrokes. We call this *oversensing*: where authorized access to sensor data provides an application with superfluous and potentially sensitive information. Manufacturers and system designers must employ the principle of least privilege at a more fine-grained level and with awareness of how often different sensors overlap in the sensitive information they leak. We project that directing technical efforts toward a more holistic conception of sensor data in system design and permissioning will reduce risks of oversensing.

Risks of Oversensing

Oversensing unintentionally leaks potentially sensitive information, such as login information, user location, and identification information, through

sensor data. Smart-device manufacturers prevent malicious applications from trivially obtaining such information through sensor permissioning systems, attempting to limit applications to only necessary sensors. However, oversensing subverts permissioning systems; one sensor’s data may allow an adversary to access sensitive information that should require a different permission. For example, motion sensor access may provide information on nearby speech. Intelligent attackers may then covertly decipher the sensitive information using tools such as machine learning.

We see primarily two mechanisms of oversensing: first, individual sensors provide information outside of their normal permission context or more information than necessary for the application goal, and second, synthesis of multiple sensor’s data to reveal new information that is beyond the scope of existing permissions. A common source of individual oversensing involves cases where a sensor converts physical stimuli not explicitly specified in its design. For example, speech



should not affect accelerometer output; however, deficiencies in signal processing such as aliasing and ineffective low-pass filters mix speech content with measured acceleration. Thus, an application with permission to use motion sensor data may also receive some speech information. Additionally, we regard a sensor providing more information than required by an application to also be oversensing. For example, consider an application that requests unfettered camera access to count the number of human occupants in a room. The application needs only the number of occupants, but receives additional unnecessary, potentially sensitive information from the video. Furthermore, attackers may synthesize multiple co-located sensors to reveal emergent information that would be outside any of the original sensor permissions. For example, access to multiple co-located accelerometers, geophones, microphones, or gyroscopes may enable an adversary to triangulate events in that room, such as keystrokes.

Vulnerabilities

Applications will soon have access to networks of sensors within smart-homes, with research showing that many of those common sensor systems already leak information via oversensing. Vulnerabilities may reside in hardware or software design. Sensor hardware designed to sense one stimulus may allow other physical stimuli to affect measurement output. Even minute alterations in output may result in privacy complications, as adversaries may use advancements in machine learning to extract private information without significant engineering. Permissioning system design may allow applications access to more sensed information than required. The research community demonstrates several examples of how a malicious application may masquerade as a benign application can receive sensor access, then use oversensed information from that sensor to access information from an elevated privilege. For example, an application could masquerade as a game to gain a motion sensor privilege, and then use

that motion sensor privilege for eavesdropping on nearby speech, which should be a separate privilege.

Illustrative Oversensing Risks

It is no surprise that adversaries with sensor access obtain information exceeding permissions or specifications, since current research in activity recognition and synthetic sensors do this by design. Selected sensors and examples of how access to each can lead to unintended consequences include the following schemes:

Motion Sensors. Adversaries with motion sensor permissions can infer sensitive information such as nearby speech, keystrokes, and location.¹ Most motion sensors are designed to capture signals with significantly lower frequency than speech, indicating that speech would not be sensed. However, typical digital filters may not eliminate all traces of higher-frequency information as needed for privacy and security. Many filters are designed to ensure the desired signal is distinct, meaning sensitive information may remain in the filtered signal in an altered state. The very presence of this

sensitive information, even altered, may be all that is necessary for an attack to render the defense ineffective.

Microphones. Recent research shows how adversaries may employ acoustic data from microphones, a common sensor permission, to create a copy of a physical key and infiltrate a home.⁴ This work enables a nearby attacker to utilize a smartphone microphone to capture the emitted sound when a victim inserts a key. This is possible because the time interval between audible clicks is correlated to the key's cut depths. This is a prime example of oversensing, as users and manufacturers would not expect that an inherent acoustic noise would enable a nearby attacker to infer one's key secrets, and eventually create a duplicate key to gain access to their house.

Power Consumption. While access to device power consumption data may seem innocuous, power consumption may leak sensitive information such as location.³ Given prior knowledge of the area, the power draw of the smartphone can be used as a proxy for the location of the phone with some prior knowledge of the area. Essentially, wireless connections such as 4G, WiFi, and so forth, require a variable power draw with signal strength. Then with a bit of machine learning, the changes within the rate of consumption of the power draw can be used to discern the relative location of the smartphone and its owner. Users are unlikely to anticipate the mismatch between the permission granted (power consumption) and what an attacker may achieve (location inference).

Yet, these examples of individual sensor oversensing are likely only the beginning of risks, as future adversaries may synthesize sensor data to achieve further capability. "Synthetic sensors"—a combination of different sensors that synergize to enable new activity recognition capabilities—have shown to be effective in non-security and privacy settings. These typically use common sensors that are included in a variety of smart-devices. In the context of security, this suggests adversaries may be able to use multiple permissions to obtain access to these same sensors, and then create synthetic sensors of their own. Moreover, adversary capability to exploit oversensing will grow as machine learning improves. Sensor-related re-

Software mitigations may not easily fix an oversensing problem that is essentially baked in on manufacture or accidentally by design.

search such as activity recognition and synthetic sensing continue to improve largely due to advancements in machine learning. This benign research may enable more sophisticated attacks through the creation of adversarial synthetic sensors. Additionally, usability improvements to machine learning toolkits allow non-expert to easily train complex models. From this we conclude that these attacks will only grow in capability.

Each of the aforementioned oversampling examples is currently difficult to detect. Adversaries have two options: to process data onboard the device or send it back for offline processing. With offline processing there is little a defender may do to detect privacy violations, however even with onboard processing the black box nature of machine learning makes discerning each application's purpose for sensor data difficult. Thus, adversarial applications may easily masquerade as benign.

Confronting Oversensing

Fixing oversensing is difficult, as one must minimally affect benign applications while balancing the effectiveness of the mitigation with ease of deployment. Higher-capability sensors are desirable to benign application designers; therefore, mitigations are burdened with maximal preservation of the original sensing characteristics. Deployment of any mitigation is further complicated by the ubiquity of devices already in existence with many different operating systems, hardware suppliers, purposes, and protocols. Software mitigations may

not easily fix an oversensing problem that is essentially baked in on manufacture or accidentally by design. Yet, hardware solutions can be tricky to design and cannot be easily updated after deployment. Despite the difficulty, there are steps taken from established fields—such as operating system design and cryptography—that researchers, operating system designers, and manufacturers can employ to begin to confront oversensing.

Principle of Least Privilege. Manufacturers and operating system designers should further integrate the operating systems concept of the principle of least privilege (POLP) into sensor system design. Often applications receive more information from sensor data than needed for full functionality. For example, an application that receives a raw camera video stream to detect presence or count people in a room is overprivileged for the task. The application's only required information is the number of people in a room, yet it will have access to additional information such as facial data, documents, or anything else the camera captures.

To employ POLP for sensors, permissioning systems must err on the side of fine-grain permissions and be far more strict in giving full sensor access, even for seemingly innocuous sensors. Taking the preceding example, the optimal solution is if a "number of people on camera" permission were available. However, providing such individualized permissions for the near limitless number of applications may be difficult. Thus, a balance between fine-grain and coarse-grain will be needed for practical deployment, but should err on the side of fine-grain permissions. Systems should minimize the number of applications that need permissions to raw data. Common machine learning features should be supported as a permission.

Development of application specific hardware may enable a realistic, fine-grain sensor permissions that could serve as the basis for practical POLP in sensor systems. A sensor processing chip could provide a large variety of common sensing-based calculations while simultaneously easing the computational burden on the processor. More infrequent permissions could be supported by operating systems as these could be calculated on demand in software.

Design Patterns. Hardware and software design patterns for oversensing resiliency should be heavily researched and adopted industry-wide. Similar to cryptography, implementing oversensing mitigations can be technically challenging, since small implementation details can render mitigations effectively useless. The difficulty lies in how an attacker goal may be achievable using only minute traces of unintended data remaining in a signal. Furthermore, determining if this is the case can be difficult, as tools to measure oversensing do not currently exist.

This new context demands that well established fixes proposed purely for performance must be carefully applied to not impact privacy. For example, consider the removal of speech information from motion sensor data. A commonly suggested method to remove the speech from the motion sensor data is to use low-pass filtering, which has existed since before the transistor. However, digital filters will encounter the problem of aliasing, which will leave traces of the original speech signal—even *digital anti-aliasing filters included on some motion sensor chips*. These filters are designed for performance, not privacy. As such they are designed to attenuate the signal so that legitimate motion is clear, but they are *not* designed to completely remove traces of all other signals. Methods such as machine learning may be able to discern remaining traces, rendering the defense ineffective.

Lessons from cryptography may inspire successful mitigation design patterns. Adding randomness to sensor output may render minute traces of sensitive information unidentifiable. Another idea may be to use non-linear data transforms on sensor data before providing it to applications, such as the aforementioned processed data for specific privileges. Dedicated sensor processing hardware may alleviate the difficulty of efficiently and verifiably implementing these design patterns—similar to dedicated crypto-hardware.

Thoughts for the Future

Anticipating Standards and Regulatory Oversight. Economic and market incentives, consumer outrage, technical difficulty for mitigation implementation, coupled with political forces, make it

likely that some level of regulatory oversight or at least industry standards will happen; how should we as a community respond to this? If history is any indication, we ignore the effect of market incentives and consumer outrage in creating regulatory frameworks at our peril. The General Data Protection Regulation (GDPR) and similar centralized privacy legislation require protection of user data, with companies having to drastically change storage guidelines and collection policy to be in compliance. In the U.S., YouTube, TikTok, and others ran afoul of the Children's Online Privacy Protection Act (COPPA) by collecting data on minors; they were fined, and had to institute software changes. IoT devices are new enough to have escaped significant regulatory scrutiny in this oversensing realm, but a parade of works like those cited in this column (and subsequent coverage in the media) have increased user awareness and market incentives for changes and standardization. As a word of caution, regulation and oversight for IoT privacy is inevitable; its coming will be made less disruptive and more beneficial should researchers, manufacturers, and developers aid the process ahead of the attacks.

Additionally, there are technical benefits for industry-wide standards or oversight due to the difficulty in oversensing mitigation implementations and the vast heterogeneity and rapid development of the IoT landscape. As previously stated, technical solutions to oversensing are challenging in a similar manner to cryptography implementations. A single, minute flaw can lead to wholly ineffective defenses in both cases, and an incredibly diverse set of hardware and applications must be able to use the systems correctly. We should learn from the more established field of cryptography, and similarly discourage in-house solutions to this deceptively technically difficult problem.

Similarly, regulation can aid users in understanding privacy concerns and reducing safety and security risks posed by the diverse set of applications strewn through the IoT landscape. There is a need for formalized oversensing risks, sensor permissions, protocols, and related language that is universal for all IoT (including smartphones). Each

Calendar of Events

At press time, scheduled conferences were significantly impacted by COVID-19, often requiring cancellation or postponement of the event. The following conferences, however, have opted to hold virtual events.

Jun. 8–11 - virtual
MMSys '20: 11th ACM Multimedia Systems Conference,
Sponsored: ACM/SIG,
Contact: Ali C. Begen,
Email: acbegen@gmail.com

Jun. 10–12 - virtual
SACMAT '20: The 25th ACM Symposium on Access Control Models and Technologies,
Sponsored: ACM/SIG,
Contact: Jorge Lobo,
Email: jorgelobo@voladorlibre.net

Jun. 14–19 - virtual
PLDI '20: ACM SIGPLAN Conference on Programming Language Design and Implementation,
Sponsored: ACM/SIG,
Contact: Alastair Donaldson,
Email: alastair.donaldson@imperial.ac.uk

Jun. 14–19 - virtual
SIGMOD/PODS '20: International Conference on Management of Data,
Sponsored: ACM/SIG,
Contact: David Maier,
Email: maier@cs.pdx.edu

Jun. 15–19 - virtual
ITiCSE '20: Innovation and Technology in Computer Science Education,
Sponsored: ACM/SIG,
Contact: Michail Giannakos,
Email: mgiannakos@hotmail.com

Jun. 15–17 - virtual
SIGSIM-PADS '20: SIGSIM Principles of Advanced Discrete Simulation,
Sponsored: ACM/SIG,
Contact: Jason Liu,
Email: liux@cis.fiu.edu

Distinguished Speakers Program

A great speaker can make the difference between a good event and a WOW event!

Students and faculty can take advantage of ACM's Distinguished Speakers Program to invite renowned thought leaders in academia, industry and government to deliver compelling and insightful talks on the most important topics in computing and IT today. ACM covers the cost of transportation for the speaker to travel to your event.

speakers.acm.org



Association for Computing Machinery

Sensor systems may leak sensitive information unwittingly by oversensing.

standard permission should be associated with an agency-assigned level of risk, with known attacks and other information made available in a singular public database. This level of risk should be made clear to users in a universal manner proven to be effective by research and case studies across manufacturers and devices. Standardization and regulation are needed as each cognitive difference between systems or methods of displaying information can distract and mislead users.

Developing Tools. There is a need for applications, programs, hardware, and other tools to assist manufacturers, application designers, and users against oversensing. Sensor manufacturers need methods to detect, measure, and otherwise test oversensing to develop mitigations against oversensing threats. For example, a hardware tool that systematically applies a variety of physical stimuli to a sensor and measures the response enables oversensing detection and measurement. Application designers need transparent APIs that give minimal privileges for any given task. Coupling a variety of commonly needed processed sensor data to the APIs may persuade designers to use such APIs instead of creating their own solution. User-focused tools are also needed. For example, tools that automatically detect oversensing risks and make such risks transparent may help users avoid overprivileged applications. High-level optional “safe-sensing” operating modes may give an easily understandable choice between power consumption and privacy to users. The development of these tools, even if imperfect, is needed to enable better mitigation design, more accurate design patterns, and provide higher-quality information for regulatory practices.

Conclusion

Sensor systems may leak sensitive information unwittingly by oversensing. Future homes are particularly at risk due to the abundance of private information coupled with increasing numbers of IoT sensing systems. Sensor and permissioning systems must better implement the principle of least privilege using far more fine-grained permissions. IoT applications should rarely receive raw sensor data, and instead should receive processed information specific to their actual need. The design of privacy-aware hardware accelerators that calculate this fine-grain sensor data may reduce privacy risk while simultaneously easing this computational burden of fine-grain permissions. Both hardware and software should follow standard design patterns to avoid easy-to-make mistakes, similarly to how in-house cryptography implementations are discouraged. With cooperation between academia, manufacturers, and regulatory agencies, the looming threat of oversensing may be mitigated before the problem grows uncontrollably. **□**

References

1. Ba, Z. et al. Learning-based practical smartphone eavesdropping with built-in accelerometer. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium* (Feb. 23–26, 2020, San Diego, CA, USA); DOI: 10.14722/ndss.2020.24076
2. Lindqvist, U. and Neumann, P.G. The future of the Internet of Things. *Commun. ACM* 60, 2 (Feb. 2017), 26–30; DOI: 10.1145/3029589.
3. Michalevsky, Y. et al. PowerSpy: Location tracking using mobile device power analysis. In *Proceedings of the 24th USENIX Security Conference*. (Aug. 12–14, 2015, Washington, D.C., USA).
4. Ramesh, S., Ramprasad, H., and Han, J. Listen to your key: Towards acoustics-based physical key inference. In *Proceedings of HotMobile '20*, (Mar. 3–4, 2020, Austin, TX, USA), ACM; DOI: 10.1145/3376897.3377853

Connor Bolton (mcbolto@umich.edu) is a Ph.D. candidate in computer science at the University of Michigan, Ann Arbor, MI, USA.

Kevin Fu (kevinfu@umich.edu) is an associate professor of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor, MI, USA.

Josiah Hester (josiah@northwestern.edu) is an assistant professor of Electrical and Computer Engineering and Computer Science at Northwestern University, Evanston, IL, USA.

Jun Han (junhan@comp.nus.edu.sg) is an assistant professor of Computer Science at the National University of Singapore.

This research was supported in part by a gift from Analog Devices Inc., a grant from the United States National Science Foundation (CNS-1915847), and a grant from Singapore Ministry of Education Academic Research Fund Tier 1 (R-252-000-A26-133).

Copyright held by authors.



Kode Vicious Kode Vicious Plays in Traffic

With increasing complexity comes increasing risk.

Dear KV,

I hear many cars today are built as distributed systems containing hundreds of CPUs that control the smallest bits of the car. These components, with millions of lines of code in them, seem to be very complicated—more so than a typical operating system. This does not sound like a terribly great idea, given that today we struggle to understand multicore behavior of systems in the presence of optimizing compilers, let alone the challenges posed by distributed systems that have no access to atomic operations. I hear they are even planning to use Ethernet moving forward. I am scared a car might malfunction or get exploited and run me over. What can we do to remedy this situation?

Sincerely,

**A Frightened Citizen
Running from Cars**

Dear Frightened,

The only thing we have to fear is fear itself—and poorly written code that has kinetic side effects and off-by-one errors. In other words, we have much to fear. There is a very simple answer to all this car silliness, and it is, of course, bicycles. Nice, mechanical, muscle-driven machines with nary a processor anywhere near them.

Unfortunately, it is unlikely bicycles will replace automobiles anytime soon, and as you point out, automobiles are becoming increasingly



automated. As people who work in software, we know this is a terrible idea because we see how much terrible code gets written and then foisted upon the world. At one point, KV might have suggested that more stringent requirements, such as those used in the aerospace industry, might have been one way to ameliorate the dangers of software in the four-wheeled killing machines all around us, but then Boeing 737s started fall-

ing out of the air and that idea went out the window as well.

There is no single answer to the question of how to apply software to systems that can, literally, kill us, but there are models to follow that may help ameliorate the risk. The risks involved in these systems come from three major areas: marketing, accounting, and management. It is not that it is impossible to engineer such systems safely, but the history of automated



Association for
Computing Machinery

2018 JOURNAL IMPACT
FACTOR: 6.131

ACM Computing Surveys (CSUR)

ACM Computing Surveys (CSUR) publishes comprehensive, readable tutorials and survey papers that give guided tours through the literature and explain topics to those who seek to learn the basics of areas outside their specialties. These carefully planned and presented introductions are also an excellent way for professionals to develop perspectives on, and identify trends in, complex technologies.



For further information
and to submit your
manuscript,
visit csur.acm.org

There is a wealth of literature on safety-critical systems, much of which points in the same direction: toward simplicity.

systems shows us that it is difficult to do so cheaply and quickly. The old adage, “Fast, cheap, or correct—choose two,” really should be “Choose correct, and forget about fast or cheap.” But the third leg of the stool here, management, never goes for that.

There is a wealth of literature on safety-critical systems, much of which points in the same direction: toward simplicity. With increasing complexity comes increasing risk, in part because humans—and I am told management is made up of humans—are quite bad at understanding both complexity and risk. Understanding the safety parameters of a system means understanding the system as a whole, and a simpler system is easier to understand than a complex one.

The first design principle of any safety-critical system must be simplicity. Systems such as Ethernet, which you reference in your letter, are known to be complex, with many hidden failure modes, and so it is a poor choice for use in a safety-critical system. But I hear accounting screaming about the cost of extra wiring in the harness of the car’s control system. “Think how much money we can save if all the signals go over a single pair of wires instead of a harness with 10!” In response, we must say, “Think of what happens when the network traffic showing your kids their favorite TV programs interferes with the signal from the brake pedal.”

Which brings up the next design principle: separation of concerns. A safety-critical system must never be mixed with a system that is not safety critical, for this both increases com-

plexity and lowers the level of safety provided by the system overall. The brakes and steering are far more important than the entertainment system, at least if you think that stopping in time for a light is more important than singing along to “Life on Mars.” This type of design failure has already shown up in a variety of systems that are safety critical, including automobiles.

A third, but by no means final, design principle can be stated as, “Don’t talk to strangers.” Many of the latest features in systems—such as automobile systems—are meant to make them an extension of the Internet. I don’t know if you have seen the Internet lately, but it is not a safe space. Why anyone should be distracted by email while driving is beyond KV’s understanding, but this is something marketing clearly wants to push, so, against all sanity, it is definitely happening. There have already been spectacular takeovers of cars in the field by white-hatted attackers, so it ought to be obvious that this is an important design principle, and anyone who tells me this problem can be solved with a firewall will be dragged behind an SUV and dumped off a cliff. Adding a firewall adds complexity, violating the first tenet mentioned previously.

The irony of this list is that it is not new. These principles have existed for at least 40 years in one form or another, but now they have more weight, or perhaps kinetic energy.

KV

Related articles on queue.acm.org

DNS Complexity

Paul Vixie

<https://queue.acm.org/detail.cfm?id=1242499>

Tom’s Top Ten Things Executives Should Know About Software

Thomas A. Limoncelli

<https://queue.acm.org/detail.cfm?id=3325792>

Programming Languages

Kode Vicious

<https://queue.acm.org/detail.cfm?id=1035604>

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

Copyright held by author.

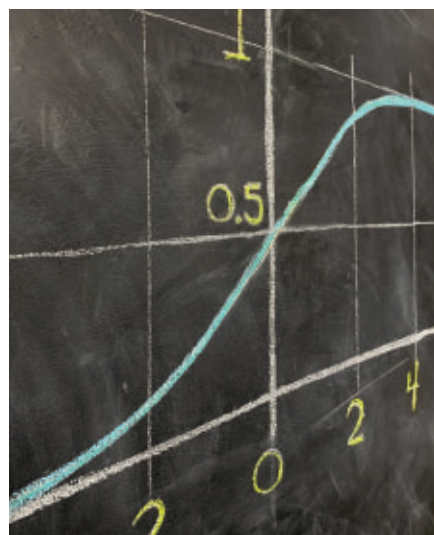
The Profession of IT Technology Adoption

The S-shaped curve of technology adoption is a welcome recurrence in an otherwise chaotic adoption world.

TECHNOLOGY ADOPTION IS accelerating. The telegraph was first adopted by the Great Western Railway for signaling between London Paddington station and West Drayton in July 1839, but took nearly 80 years to peak (1920).^a The landline telephone took 60 years to reach 80% adoption, electric power 33 years, color television 15 years, and social media 12 years.^b The time to adoption is rapidly decreasing with advances in technology.

When we develop new technology, we would dearly like to predict its future adoption. For most technologies, total adoptions follow an *S* curve that features exponential growth in number of adopters to an inflection point, and then exponential flattening to market saturation. Is there any way to predict the *S* curve, given initial data on sales?

Technology adoption means that people in a community commit to a new technology in their everyday practices. A companion term diffusion means that ideas and information about a new technology spread through a community, giving everyone the opportunity to adopt. Adoption and diffusion are not the same. Here we are interested in adoption as it is manifest in sales of technology. Adoption models attempt to estimate two quantities that affect business decisions whether to produce technology. One is the total addressable market *N*, the number of



people who will ultimately adopt. The other is t^* , the time of the inflection point of the *S* curve.

It would seem that to develop a model of the *S* curve we would need a model of the underlying process by which technology is produced and sold. Three process models are common:

► *Pipeline*: an idea flows through the stages of invention, prototyping, development, marketing, and sales, finally being incorporated into the marketplace as a product people buy.

► *Funnel*: similar to pipeline but the pipeline begins with multiple ideas and each stage winnows the number passed to the next stage until finally one product emerges into the marketplace. This model aims to compensate for the high failure rate of ideas. If failure rate is 96% (a common estimate), the funnel-pipeline must be seeded with 25 ideas so that there will be one survivor to the final stage.

► *Diffusion-Adoption*: ideas are treated as innovation proposals that spread through a social community, giving each person the opportunity to adopt it or not.

Unfortunately, there are important innovations that are not explained by some or all of these models. For example, spontaneous innovations do not follow the pipeline or funnel models, and many diffusions do not result in adoption. Moreover, the models are unreliable when used as ways to organize projects—they explain what happened in the past but offer little guidance on what to do in the immediate future. Many organizations manage their internal processes according to one of these models. People in these organizations frequently experience a “Fog of Uncertainty” when something unanticipated comes up in one of the stages and it is not obvious what to do.

Insight from the Diffusion Model

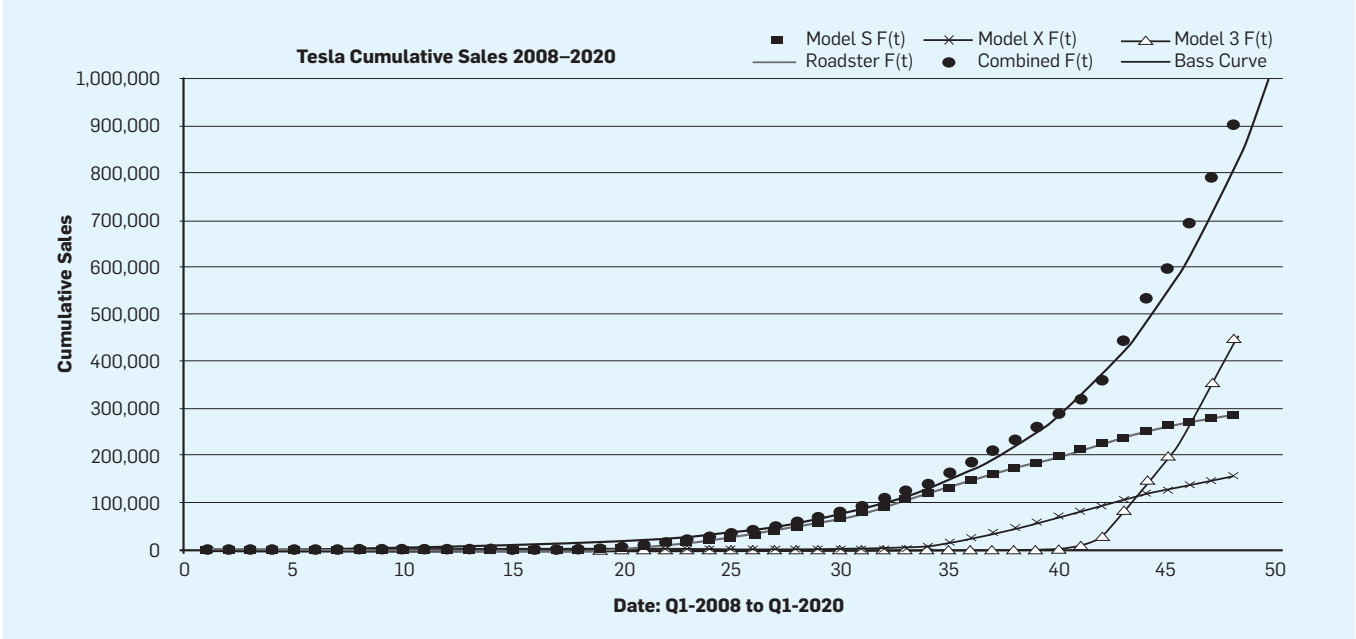
The defects in pipeline models brought much attention to Everett Rogers’ diffusion model, first introduced in 1962.⁵ Rogers’ diffusion model explicitly recognizes social interactions in technology adoption. Neither the pipeline nor funnel models gives a role to social preferences in the adopting communities. The diffusion model postulates that individuals in the community receive information through their social networks that enables them at some point to decide to adopt the technology.

Rogers observed that each individual in a community takes a different time to adopt. He found that the statistics of time-to-adoption follow

a See https://en.wikipedia.org/wiki/Telegraphy#Electrical_telegraph

b See <https://ourworldindata.org/technology-adoption>

Figure 1. Cumulative quarterly sales of Tesla products from 2008 to 2020 and the Bass S-shaped curve fit for combined cumulative sales.



a Bell curve. On a Bell curve 68% of adoptions are within one standard deviation S of the mean adoption time T . Rogers labeled those adopters “the majority.” He also labeled the 12.5% of adopters in the band from $T-2S$ to $T-S$ the “early adopters” and those in the 2.5% band above $T+2S$ the “laggards” who will never adopt. His key insight was to connect these bands of the Bell curve with dispositions of community members toward adopting a proposed innovation. From many interviews, Rogers confirmed that the early adopters are disposed to adopt quickly, whereas the majority are disposed to wait until the technology is widely adopted, trustworthy, and reliable. And the laggards are disposed against any innovation.

An important insight from these distinctions is that the company needs a different appeal for early and majority adopters. In 1991, Geoffrey Moore explained many start-up business failures by noting their initial business plans aimed at early adopters, and by the time they realized they saturated

that market, it was too late to raise new capital to organize for the majority.⁴ He advised business leaders, as a matter of survival, to anticipate the majority as part of their initial planning.

Thomas Edison is a famous historical innovation leader who understood this. Although he is often credited for inventing the light bulb, he did not begin to experiment with light bulbs until he saw a way to generate and distribute electricity. Without cheap in-home electricity, there would be few customers for light bulbs. In 1878, when he finally found a technology to generate and distribute electricity, he set his research staff searching for a light bulb that would last for days or weeks without burning out. The idea of a light bulb was about 60 years old by that time, but the bulbs that existed burnt out within a few minutes of being turned on.

Elon Musk is a modern business leader who understands the need to make offers for majority adopters. Even before the first Tesla electric car rolled off the assembly line, he said

that without charging stations there would be few customers for electric cars. He invested in building a network of charging stations that would be available for the first Tesla drivers.

Novelty and Imitation

Rogers’ early adopters respond to novelty. The newness of a technology and possibilities it offers are very attractive to them. In contrast, majority adopters are imitators. They will not adopt unless they see a sufficient number of others already doing so.³

The process of adopting by imitation is dominated by what complexity theorists call “preferential attachment.” That means that an adopter may have choices between several technologies, or none at all. Imitation adopters will tend to line up with the option that seems to have attracted the biggest following—safety in numbers.

Complexity theory has established that a process dominated by preferential attachment has a power law distribution of times between adoptions. The power law distributions observed in practice have no finite means or standard deviations. That means that traditional statistical methods of predicting the time till next event, or the number of events in a time interval, all fail.² This puts project leaders into a bind: they cannot use standard statistical methods to predict how successful the future adoptions of their technology will be.

Results for Bass model and projections.

| Parameter | Roadster | Model S | Model X | Model 3 | Combined |
|--------------------------------------|----------|---------|---------|---------|--------------------|
| p (innovate) | .0265 | .0087 | .0104 | .0077 | 7×10^{-6} |
| q (imitate) | 0.9470 | 0.1556 | 0.2347 | 0.4042 | 0.1302 |
| t^* (time to inflection, quarters) | 3.67 | 17.56 | 12.70 | 9.62 | 75.43 |
| N (total market) | 1,896 | 311,185 | 234,736 | 808,361 | 43,402,353 |

Fitting Bass Equation to Data

The Bass model is an S -curve of total sales (adoptions) defined by this differential equation, where s and S are functions of time t :

$$s = \frac{dS}{dt} = (p + \frac{q}{N}S)(N - S)$$

Function S is the total sales. Function s is the rate of change of S , that is, the sales rate. Parameter N is the total size of market, that is, the maximum number of adoptions possible. Parameter p is the rate of spontaneous adoptions from non-adopters. Parameter q is the rate of imitation, that is, the rate at which each existing adopter attracts more adoptions.

The right side of the equation can be understood by separating it into two terms. The term $p(N-S)$ is the rate at which non-adopters spontaneously become adopters. The term $qS(1-S/N)$ is the imitation rate generated by current adopters (qS) throttled back by the fraction of market that has not yet adopted. The solution to this equation is

$$S(t) = N \frac{1 - e^{-at}}{1 + re^{-at}}$$

Where $a=p+q$ and $r=q/p$. This equation has starting value $S=0$ (initially no one has adopted) and final value for large time of $S=N$ (everyone has adopted). Its inflection point is $t^*=(\ln r)/a$.

The right side of the Bass equation can be re-expressed as a quadratic equation,

$$s = Np + (q - p)S - \frac{q}{N}S^2$$

This can be interpreted as the quadratic form

$$s = A + BS + CS^2$$

where $A = Np$ and $B = q-p$, and $C = -q/N$. This form is very useful for fitting the model to data because the available data are usually in the form of sales rate versus total sales, that is, s versus S . Using a least-squares fitting algorithm we can find A , B , and C that give the least error fit of the right side of this quadratic form with the data. The total market N is unknown but can be estimated from the data as follows. When the quadratic form for s is 0, there can be no growth of S . The solutions of the corresponding quadratic equation are:

$$S = \frac{-B \pm \sqrt{B^2 - 4AC}}{2C}$$

Thus $s=0$ when S is either of the values

$$S = N \quad \text{and} \quad S = -\frac{Np}{q}$$

Since negative sales are of no interest, the positive root of the quadratic form gives us the estimate of N . Then

$$p = \frac{A}{N} \quad \text{and} \quad q = B + \frac{A}{N}$$

Thus, the model parameters p , q , and N can be estimated from the data.

N . The result is shown as “Combined $S(t)$ ” in the figure here. Similar S -shaped curves are shown for all models. The accompanying table summarizes the values for p , q , t^* , and N . It is interesting to note that p is lowest for the Model 3, suggesting that conservative imitators are dominating sales of the Model 3. In the same period sales of the similarly priced BMW 3/4 series cars declined from 140,000 units to 66,000 units, suggesting that customers are preferentially attaching themselves to Tesla.

The adoption curves for Roadster, Model S, and Model X each take over when the previous one starts to sag. This suggests Tesla is engaging in technology jumping: as sales of one model peak and reach an inflection point, the next generation begins. When consumers jump to Model 3, sale of Model S and X decline.

Conclusion

But Bass’s model is limited. It offers no guidance for project leaders on a day-to-day basis. How do they deal with unexpected contingencies? Factors that cause deviations from the business plan? Project leaders need to learn navigational skills to make it through these trials.¹

The S -curve and its Bass model are a welcome recurrence in an otherwise chaotic adoption world. Because of this, the S -curve can be a very useful tool for businesses seeking to estimate the total size of their market and the time peak sales rate. □

References

- Denning, P. Winning at innovation. *IEEE Computer* (Oct. 2018), 32–39.
- Denning, P. and Lewis, T. Uncertainty. *Commun.* 62, 12 (Dec. 2019), 26–28.
- Lewis, T.G. and Bergin, R. Imitation and novelty in product development. *Cognitive Systems Research* 38 (2016), 23–30.
- Moore, G. *Crossing the Chasm*. Harper Collins, 2014, First edition 1991.
- Rogers, E. *Diffusion of Innovations*. 5th Ed. Free Press, 2003, First edition 1962.

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.

Ted G. Lewis (tedglewis@redshift.com) is an author and consultant with more than 30 books on computing and hi-tech business, a retired professor of computer science, most recently at the Naval Postgraduate School, Monterey, CA, Fortune 500 executive, and the co-founder of the Center for Homeland Defense and Security at the Naval Postgraduate School, Monterey, CA.

Copyright held by authors.

The Bass Model

But all is not lost. Many studies of adoption processes have found that the number of adoptions over time follows an S -curve, irrespective of the innovation models managers espouse or chaos in the adoption process. The S -curve is a stable feature of almost all technology adoption processes. Is it possible to estimate the S -curve from its initial adoption data? If so, managers could estimate the saturation market and time to inflection.

In 1969 Frank Bass, a marketing analyst, came up with a model of the S -curve that had just two parameters that could be measured from initial

adoption data. Bass’s two parameters correspond to Rogers’ early adopters and majority. Bass’s first parameter p is that rate at which early adopters adopt spontaneously. His second parameter q is the rate at which the majority adopts by imitation. Bass defined a differential equation whose solution $S(t)$ is the expected number of sales (adoptions) by time t , given that the process began at time 0. When t is large, $S(t)$ levels off at N , the total market size; see the sidebar.

Let us illustrate with sales figures from Tesla Motors. We used the method of the sidebar to fit a Bass curve to the combined sales of all models since the beginning of Tesla. This yields p , q , and

Viewpoint

Studying Programming in the Neuroage: Just a Crazy Idea?

Programming research has entered the Neuroage.

“THIS IS A crazy idea,” the review read. Closing my laptop lid, I added in my mind “and ... it will never work,” as a lump welled in my throat. What we were proposing to do was simple yet ambitious. Using functional magnetic resonance imaging, we might better understand what goes on in the minds of programmers as they read and understand code. We had performed pilot experiments with a neurobiologist, had promising results, and encouraging words from colleagues and reviewers. Still, the words, “this is a crazy idea,” echoed in our minds. Would it be possible to break open the stale progress in program comprehension research? After all, researchers have been working on understanding programmers since the 1970s. Could neuroimaging really help devising a conclusive and comprehensive theory of program comprehension?

The Waves of Program Comprehension Research

Research in program comprehension has been a cycle of booms and busts. In the early 1970s and 1980s, the first wave of researchers were psychologists, using methods, such as memory recall, to probe how programmers represent and process code in their mind. As a result, various theories and mechanisms were proposed, such as programming plans and bottom-up com-



prehension, but no clear consensus and only few impulses for programming research (including research on programming methodology, language design, or education) emerged.

More than a decade passed without significant research progress and many leaving the field.²⁵ In the mid-2000s, a second wave of researchers emerged, now with “big code” as the methodology of choice. Researchers mined traces of program comprehension as manifested through program-

ming activities in code repositories, such as GitHub, asking statistical questions, such as whether long or short identifier names lead to more defects. While this data has proved valuable, the community was drifting farther and farther away from understanding the inner workings of the programmer’s mind, and so did our ability to devise and validate a conclusive and explanatory theory of program comprehension.

Meanwhile, in the field of psychol-

ogy and cognitive neuroscience, considerable progress has been made in building theories of fundamental cognitive processes, such as language comprehension and logical reasoning. One important enabling technology was brain imaging, including functional magnetic resonance imaging (fMRI), electroencephalography (EEG), and functional near infrared spectroscopy (fNIRS). These methods have revolutionized the understanding of cognitive processes and are routinely used as a measurement tool in various disciplines, including psychology, economics, and social sciences.²⁴

From Idea to Implementation

Excited by the developments in brain imaging and its successful application in other disciplines, the idea grew to revitalize the research efforts of program comprehension. The first step was to build a team. We had to make several pitches to different scientists, before we finally found a neurobiologist who would listen to our idea. With him (the third author) on board, we started by absorbing the relevant neuroscience literature, which required several years of investigation and an in-depth understanding of brain imaging methods and what they can or cannot do.²⁴ Most notably, brain-imaging methods cannot directly observe cognitive processes, but only their neurobiological correlates, which poses limitations on measurement resolution, experiment design, and interpretability of results. For example, fMRI measures the change in the oxygenation level of blood as brain activity occurs based on different magnetic properties of oxygenated and deoxygenated blood. Oxygenation changes take time, adding a lag of a few seconds (that is, the hemodynamic lag¹²). This limits the temporal resolution of fMRI to the order of seconds. As further constraint, one needs a baseline level of activation as comparison to detect changes in the first place. Also financial constraints play a role (one hour in our fMRI scanner costs about 150 €).

Further challenges lie in the experimental design. First, one needs to develop a set of tasks that could be performed by programmers to measure the act (and only the act) of program comprehension. This is in itself chal-

lenging, as there is no canonical set of programming problems. Additionally, the tasks need to respect the limitations of brain imaging methods, which affect code length and the time to accomplish the task. For example, only 20 lines of code can be reasonably shown without allowing scrolling, and they require about 30 to 60 seconds to understand (depending on complexity), so we used 60 seconds as time limit for each snippet. Longer source code that requires scrolling and longer task duration is also possible in today's scanners. After many discussions and several pilot studies, we arrived at a set of general program comprehension tasks covering basic (imperative) program structures and programming problems inspired by introductory programming courses and textbooks.

A second key ingredient is a method to subtract brain activity related to program comprehension from brain activity unrelated to program comprehension, for example, related to reading,

speaking, or motor activity. If we just observe programmers while they work with source code, we see a lot of activated brain areas, but we do not know which are directly related to the act of program comprehension. To determine which part of the brain is specifically activated during program comprehension, we apply a subtraction method,⁵ illustrated in Figure 2: We let the programmer identify syntax errors in code as a baseline task, called control condition, which reveals the difference between “glancing through” the code as compared to deeply understanding its semantics. This subtraction method is conservative in order not to discover spurious activations, such that, activation not related to the act of program comprehension is filtered out as much as possible.

After several years of planning and many pilot tests outside an fMRI scanner, we had arrived at an experimental design that could be executed inside the fMRI machine. We presented our

Figure 1. Experiment design of the first fMRI study²⁶; participants understood a source code snippet (60s), followed by a rest period to let the oxygenation level return to the baseline (30s), followed by identifying syntax errors (30s), and another rest period (30s); this process was repeated for 12 times.

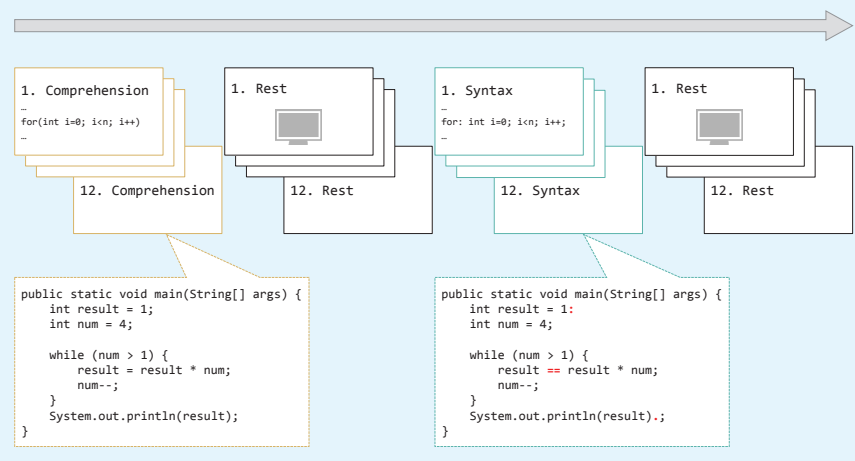
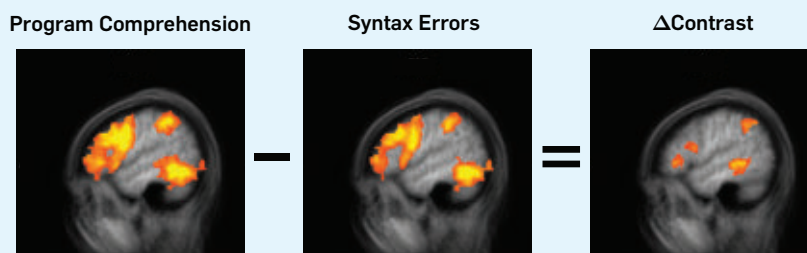


Figure 2. The subtraction method: Program comprehension triggers also unrelated brain activation, so we subtract brain activation during identifying syntax errors to obtain brain activation specific to program comprehension.



plans at the New Ideas and Emerging Results track at ESEC/FSE in 2012, receiving very diverse feedback, ranging from “oh wow, this is so cool” to “that will never work.”

The First Program Comprehension Study of Its Kind

We conducted our first study with 17 students in an fMRI scanner understanding a dozen code snippets (Figure 1 provides an overview of the experiment procedure), all well prepared in terms of difficulty and duration needed to understand by the average programmer.²⁶ The results indicated that a specific network of brain areas in the left hemisphere was used by participants to understand code, including areas related to working memory, divided attention, and reading comprehension. Surprisingly, we did not observe cognitive processes related to mathematical and logical reasoning, which would be consistent with the perspective that programming is a formal, logical, and mathematical process. The most striking result is a clear left-lateral activation during program comprehension. In conjunction with

Could neuroimaging really help devising a conclusive and comprehensive theory of program comprehension?

the activation in semantically associated areas (BAS 21 and 47), this suggests that program comprehension involves semantic processes that are also relevant in language processing. Although neuroscience researchers still argue about the differential role of the two hemispheres in semantic processing in general, there is clear evidence on the dominant role of the left hemisphere in language-related processing. If the role of the left hemisphere in program comprehension can be consolidated in future studies, this would strengthen Dijkstra’s con-

jecture that a proper education in natural languages is imperative for successful programmers.⁴

Further Studies

Although standard in neuroscience, the complexity of the process and machinery led to the question of how reliable the results are—doubts were rising. Following best practice in neuroscience, we replicated our study twice with varying participants, comprehension tasks, and source code snippets. Especially, the small initial sample drove us to collect more data to increase statistical power. However, as more studies are following with more specific questions, leading to smaller differences in conditions, we need to increase the sample size. Unfortunately, without upfront experience about the expected effect size, it is difficult to recommend a sample size. In the replications, we reliably found activation in the same brain areas,^{23,27} which increased our confidence in the validity of our setup and results. This reliability of the setup is possibly even more valuable than the individual results we obtained so far.

The Hasso Plattner Institute (HPI) is Germany’s university excellence center for digital engineering. The Faculty of Digital Engineering, established jointly by HPI and the University of Potsdam, offers a practical and engineering-oriented study program in computer science that is unique throughout Germany.

Annually, HPI’s Research Schools grant
18 Ph.D. and Postdoctoral Scholarships.

With their interdisciplinary and international structure, HPI’s two Research Schools on “Service-oriented Systems Engineering” and “Data Science and Engineering” interconnect HPI’s research groups as well as its branches at the University of Cape Town, Technion, Nanjing University, and UC Irvine.

HPI RESEARCH GROUPS

- Algorithm Engineering, Prof. Dr. Tobias Friedrich
- Business Process Technology, Prof. Dr. Mathias Weske
- Computer Graphics Systems, Prof. Dr. Jürgen Döllner
- Cybersecurity - Enterprise Security, Prof. Dr. Christian Dörr
- Cybersecurity - Identity Management, Prof. Dr. Anja Lehmann
- Data Analytics and Computational Statistics, Prof. Dr. Bernhard Renard
- Data Engineering Systems, Prof. Dr. Tilmann Rabl
- Digital Health, Prof. Dr. Erwin Böttinger, Prof. Dr. Bert Arnrich, Prof. Dr. Christoph Lippert
- Enterprise Platform and Integration Concepts, Prof. Dr. h.c. Hasso Plattner
- Human Computer Interaction, Prof. Dr. Patrick Baudisch
- Information Systems, Prof. Dr. Felix Naumann
- Internet Technologies and Systems, Prof. Dr. Christoph Meinel
- Operating Systems and Middleware, Prof. Dr. Andreas Polze
- Software Architecture, Prof. Dr. Robert Hirschfeld
- System Analysis and Modeling, Prof. Dr. Holger Giese

Applications must be submitted by August 15 of the respective year. For more information on HPI’s Research Schools please visit: www.hpi.de/research-school



Advertise with ACM!

For a media kit
with specifications
and pricing, contact
acmm mediasales@acm.org

Further evidence on the validity of our results came also from other research teams, who replicated and extended our experimental design. Inspired by our study, Floyd and others changed the contrast condition to review source code (instead of finding syntax errors) and added a third condition to review natural-language prose.⁸ They could also replicate a subset of the brain areas that we found. Lee and others replicated our results with EEG. In contrast to fMRI, EEG directly measures the electrical responses of neurons with high temporal resolution (that is, in the order of milliseconds), but with a limited spatial resolution. They found brain areas that we also found, with the additional insight that programming expertise modulates the respective activation strength.¹⁷

Despite all these results, it is important to note that typically, no brain area is specifically associated with one cognitive process, but with several. For example, BAs 21, 44, and 47 are also associated with spoken language, but since we did not require participants to give a spoken response, it is not relevant for our setting. Open databases, such as BrainMap or Neurosynth, let us identify which tasks and cognitive processes are associated with a selected brain region (Genon and others give a summary of how we can map cognitive processes to brain areas¹⁰).

The Third Wave of Program Comprehension Research

Other research teams also adopted a neurocognitive perspective on program comprehension: Kosti and others used our experiment design to record EEG data and found a correlation of EEG activation with subjective rating of difficulty.¹⁵ Studies using fNIRS found activation in frontal parts of the brain depending on the kind of task (memorizing variable names vs. mental arithmetic)¹⁴ and task difficulty.²⁰ Duraes and others used fMRI to observe the neural activation during the location of code defects, identifying a specific activity pattern when a bug was spotted and confirmed.⁶ Studies using EEG revealed a relationship between mental load and expertise^{3,28} such that with lower expertise, the same tasks require higher mental load. This is because experts

have developed an efficient cognitive representation, so they can manage more information without requiring a larger working memory capacity. This can be observed in many areas other than program comprehension, for example, with expert chess players or superior memorizers. They do not have an exceptionally large working memory capacity, yet they can recall large amount of information by using specific encoding strategies, that is, certain configurations that imply the positions of chess figures² or visual objects linked to spatial landmarks.¹⁸ Compared to non-experts, this is reflected in a less pronounced or different activation pattern that develops concomitant with their growing expertise.¹⁹ The mental load is also reflected in the response of the default mode network, which is typically activated during self-referential processing and deactivated during a cognitively demanding task to avoid interference. The higher the cognitive load, the stronger the deactivation.⁹ We observed this effect also in our recent study, such that more complex tasks are related with a stronger deactivation of the default mode network.²²

A Multimodal Approach

Despite promising results, the community has realized quickly that a single imaging or measurement method is not sufficient to understand the full complexity of the cognitive processes involved in program comprehension and, more importantly, to connect the cognitive processes to the programmer's behavior. Only a multimodal approach is able to achieve this. As an

Neuroimaging offers a unique opportunity to understand, build, and test theories of program comprehension like never before.

example, a recent study²³ suggests that the integration of eye tracking and fMRI is able to draw this connection by connecting eye movement patterns and brain activation to the programmer's strategy of program comprehension (the eye movement pattern observed suggests that programmers follow program identifiers during top-down comprehension). Fakhoury and others combined fNIRS and eye tracking showing that, when participants are examining unsuitable identifier names, cognitive load increases.⁷ A multimodal setting is able to better identify expert programmers, as Lee and others showed by extending their EEG study with eye tracking.¹⁶ Huang and others have investigated programmers' brains while manipulating data structures, such as binary trees, in contrast to mental rotations with fMRI and fNIRS. The study not only provided insights into programmers' cognitive processes, but also showed that fMRI is more sensitive for identifying cognitive processes than fNIRS (in terms of statistical power and spatial resolution).¹¹ Nevertheless, fNIRS may be an interesting alternative in certain settings, as it is cheaper, less restrictive, and easier to combine with other modalities.

fMRI and EEG can also be combined to complement their strengths and weaknesses.¹³ For example, Bledowski and others mapped the stages of retrieving information from working memory to the fast neuronal response of different brain areas, shedding light on how the retrieval process unfolds temporally and spatially.¹

fMRI not only lets us dive deeper into cognitive processes, but it also allows us to decode brain activity to predict the tasks (referred to as reverse inference).²⁴ Floyd and others trained a classifier based on the data of their replication study to predict the kind of task participants were completing (79% accuracy). Future studies might reveal whether specific aspects of a task (for example, words vs. numbers being manipulated in source code) are represented in specific brain areas or in different activation strengths.

Entering the Neuroage

Programming research has entered the Neuroage. Neuroimaging offers

a unique opportunity to understand, build, and test theories of program comprehension like never before. It empowers researchers to obtain insights into the cognitive processes involved and their connection to the programmer's behavior. This way, the research community will be able to understand why, how, and to what extent program comprehension takes place, not only whether. For example, rather than stating that programmers tend to process loops faster than recursive structures, we would like to quantify how and to what extent the use of loops or recursion influences task completion time and which cognitive processes are responsible for this difference (for example, keeping track of intermediate results in recursive computations easily exceeds the programmer's working memory), and how this is modulated by expertise.

Although research has started off just a few years ago, the small but growing community already provided interesting insights into the cognitive processes of program comprehension. By adopting neuroimaging methods, researchers may overcome the stale progress of program comprehension by better understanding accompanying cognitive subprocesses. This progress can pave the way toward a theory of program comprehension. Maybe program comprehension is not a unique cognitive process, but just a special form of problem solving? Maybe cognitive modeling (for example, using cognitive architectures like ACT-R, which let researchers describe a cognitive process as a series of discrete operations²¹) can help to describe the fundamental cognitive processes underlying program comprehension? We cannot know, yet, but having demonstrated the validity and usefulness of neuroimaging methods for programming research, the real fun (and hard work) begins. With these methods, we might be able to understand why programmers make mistakes, why some code is more difficult than other, what programmer expertise looks like, and how we can best train and prepare programmers. Ultimately, these are steps forward to reach a more conclusive and comprehensive theory of program comprehension.

We believe this research direction

Maybe program comprehension is not a unique cognitive process, but just a special form of problem solving?

can also feed back to neuroscience research, such that we may discover new nuances of cognitive processes. Specifically, our setup requires participants to mentally execute code, so program comprehension is more than reading comprehension, and it might be different from other mental simulations, say thinking through a cooking recipe. We have reached an exciting new point in studying the human programmer, and we cannot wait to see where the research community will take it. **C**

References

- Bledowski, C. et al. Mental chronometry of working memory retrieval: A combined functional magnetic resonance imaging and event-related potentials approach. *Journal of Neuroscience* 26, 3 (2006), 821–829.
- Chase, W. and Simon, H. Perception in Chess. *Cognitive Psychology* 4, 1 (1973), 55–81.
- Crk, I., Kluthe, T., and Stefik, A. Understanding programming expertise: An empirical study of phasic brain wave changes. *ACM Transactions on Computer–Human Interaction* 23, 1 (2015), 1–29.
- Dijkstra, E. *Selected Writings on Computing: A Personal Perspective*. Springer, 1982.
- Donders, F. On the speed of mental processes. *Acta Psychologica* 30, 1 (1969), 412–431.
- Duraes, J. et al. WAP: Understanding the brain at software debugging. In *Proceedings of the International Symposium on Software Reliability Engineering (ISSRE)*, IEEE, 2016, 87–92.
- Fakhoury, S. et al. The effect of poor source code lexicon and readability on developers' cognitive load. In *Proceedings of the International Conference on Program Comprehension (ICPC)*, ACM, 2018, 286–296.
- Floyd, B., Santander, T., and Weimer, W. Decoding the representation of code in the brain: An fMRI study of code review and expertise. In *Proceedings of the International Conference on Software Engineering (ICSE)*, pages 175–186. IEEE, 2017.
- Gazzaniga, M., Ivry, R., and Mangun, G. *Cognitive Neuroscience: The Biology of the Mind*. Norton & Company, 2013.
- Genon, S. et al. How to characterize the function of a brain region. *Trends in Cognitive Sciences* 22, 4 (2018), 350–364.
- Huang, Y. et al. Distilling neural representations of data structure manipulation using fMRI and fNIRS. In *Proceedings of the International Conference on Software Engineering (ICSE)*. IEEE, 2019.
- Huettel, S., Song, A., and McCarthy, G. *Functional Magnetic Resonance Imaging, volume 3*. Sinauer Associates, 2014.
- Huster, R. et al. Methods for simultaneous EEG-fMRI: An introductory review. *Journal of Neuroscience* 32,18 (2012), 6053–6060.
- Ikutani, Y. and Uwano, H. Brain Activity Measurement during program comprehension with NIRS. In *Proceedings of the International Conference Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, IEEE, 2014, 1–6.
- Kosti, K. et al. Towards an affordable brain computer interface for the assessment of programmers' mental workload. *J. Human–Computer Studies* 115, (2018), 52–66.
- Lee, S. et al. Mining biometric data to predict programmer expertise and task difficulty. *Cluster Computing* 21, 1 (2018), 1097–1107.
- Lee, S. et al. Comparing Programming Language Comprehension between Novice and Expert Programmers Using EEG Analysis. In *Proceedings of the International Conference on Bioinformatics and Bioengineering (BIBE)*, IEEE, 2016, 350–355.
- Mallow, J. et al. Superior memorizers employ different neural networks for encoding and recall. *Frontiers in Systems Neuroscience* 9, 128 (2015).
- Moore, C. Neural mechanisms of expert skills in visual working memory. *Journal of Neuroscience* 26, 43 (2006), 11187–11196.
- Nakagawa, T. et al. Quantifying programmers' mental workload during program comprehension based on cerebral blood flow measurement: A controlled experiment. In *Proceedings of the International Conference on Software Engineering (ICSE)*, ACM, (2014), 448–451.
- Newell, A. *Unified Theories of Cognition*. Harvard University Press, 1994.
- Peitek, N. et al. A look into programmers' heads. *IEEE Transactions on Software Engineering* 46, (2020), 442–462.
- Peitek, N. et al. Simultaneous measurement of program comprehension with fMRI and eye tracking: A case study. In *Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM)*, ACM, 2018, 1–24.
- Poldrack, R. *New Mind Readers: What Neuroimaging Can and Cannot Reveal about our Thoughts*. Princeton University Press, 2018.
- Siegmund, J. Program comprehension: Past, present, and future. In *Proceedings of the International Conference Analysis, Evolution, and Reengineering (SANER)*, IEEE, 2016, 13–20.
- Siegmund, J. et al. Understanding source code with functional magnetic resonance imaging. In *Proceedings of the International Conference on Software Engineering (ICSE)*, ACM, 2014, 378–389.
- Siegmund, J. et al. Measuring neural efficiency of program comprehension. In *Proceedings of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE)*, ACM, (2017), 140–150.
- Yeh, M. et al. Detecting and comparing brain activity in short program comprehension using EEG. In *Proceedings of Frontiers in Education Conference*, IEEE, 2017, 1–5.

Janet Siegmund (siegmunj@fim.uni-passau.de) is a professor of computer science at Chemnitz University of Technology, Chemnitz, Germany.

Norman Peitek (norman.peitek@lin-magdeburg.de) is a Ph.D. student at the Leibniz Institute for Neurobiology, Magdeburg, Germany.

André Brechmann (brechmann@lin-magdeburg.de) is head of the Combinatorial NeuroImaging Core Facility at the Leibniz Institute for Neurobiology, Magdeburg, Germany.

Chris Parnin (cjparnin@ncsu.edu) is a professor of computer science at North Carolina State University, Raleigh, USA.

Sven Apel (sven.apel@acm.org) is a professor of computer science and chair of software engineering at Saarland Informatics Campus and Saarland University, Saarbrücken, Germany.

This work has been funded by the German Research Foundation (SI 2045/2, BR 2267/7, AP 206/6).

Copyright held by authors.

Viewpoint

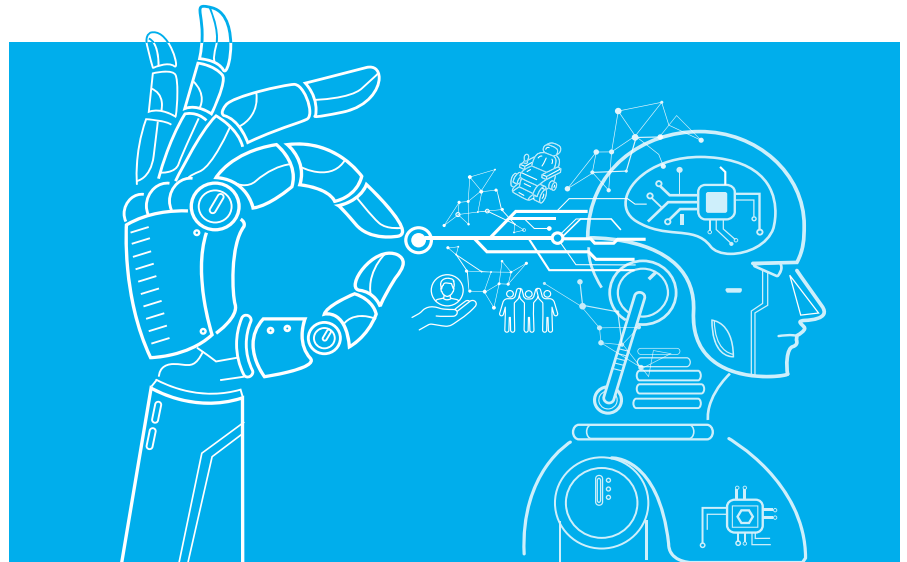
AI and Accessibility

A discussion of ethical considerations.

ACCORDING TO THE World Health Organization, more than one billion people worldwide have disabilities. The field of disability studies defines disability through a social lens; people are disabled to the extent that society creates accessibility barriers. AI technologies offer the possibility of removing many accessibility barriers; for example, computer vision might help people who are blind better sense the visual world, speech recognition and translation technologies might offer real-time captioning for people who are hard of hearing, and new robotic systems might augment the capabilities of people with limited mobility. Considering the needs of users with disabilities can help technologists identify high-impact challenges whose solutions can advance the state of AI for all users; however, ethical challenges such as inclusivity, bias, privacy, error, expectation setting, simulated data, and social acceptability must be considered.

Inclusivity

The inclusivity of AI systems refers to whether they are effective for diverse user populations. Issues regarding a lack of gender and racial diversity in training data are increasingly discussed; however, inclusivity issues with respect to disability are not yet a topic of discourse, though such issues are pervasive.^{4,16} For example, speech recognition systems, which have become popularized by virtual assistants, do not work well for people with speech disabilities such as dysarthria, deaf accent, and other conditions since train-



ing data does not typically include samples from such populations.⁶ Advances in computer vision have led several groups to propose using such algorithms to identify objects for people who are blind, but today's vision-to-language algorithms have been trained on datasets comprised of images taken by sighted users, limiting their efficacy when applied to images captured by blind users, which tend to have far lower quality.² These inclusivity issues threaten to lock people with disabilities out of interacting with the next generation of computing technologies. Proposed methodologies for increasing awareness of the provenance and limitations of datasets³ are a starting point to increasing dataset transparency. Efforts to directly source data from underrepresented user groups, such as the VizWiz dataset,⁵ which contains thousands of images and related questions captured by people who are blind, are a step in the right direction.

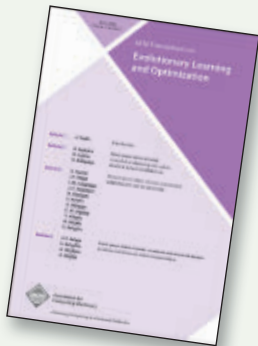
Bias

Just as AI technologies can amplify existing gender and racial biases in our society, they can also exacerbate disability-based discrimination. Recent research indicates that AI researchers can infer disability status from online data traces; for example, our research showed we could infer whether people are blind by analyzing their Twitter profiles and activity,¹⁰ and researchers have also demonstrated the ability to assess whether people have Parkinson's disease based on their mouse movements while on a search engine homepage.¹⁸ The potential for peoples' disability status to be implicitly revealed through their computing actions means that algorithms could treat users differentially based on inferred disability status, such as the possibility for health insurers to deny coverage or employment advertisements to be targeted to avoid display to people with disabilities.⁷ Developing ethical and legal frameworks regarding



ACM Transactions on Evolutionary Learning and Optimization (TELO)

ACM Transactions on Evolutionary Learning and Optimization (TELO) publishes high-quality, original papers in all areas of evolutionary computation and related areas such as population-based methods, Bayesian optimization, or swarm intelligence. We welcome papers that make solid contributions to theory, method and applications. Relevant domains include continuous, combinatorial or multi-objective optimization.



For further information and to submit your manuscript, visit telo.acm.org

the application of AI to inferring disability status is a crucial issue.

Privacy

People with rare disabilities may experience greater privacy risks if they contribute data to AI systems or participate in research studies evaluating AI technologies. Past incidents of re-identification of individuals from anonymized datasets, such as the 2006 AOL search data leak, indicate the difficulty of truly anonymizing data. For some small disability populations, privacy-preserving techniques such as k -anonymity may not be effective, creating increased re-identification risk. These increased privacy risks to people with disabilities are amplified by the aforementioned *bias* issues that people with disabilities may face if their disability status is exposed, and creates an incentive for people with disabilities to withhold their data from research studies, an issue that can further amplify the *inclusivity* problem of AI systems. Hence, reflection on how current research practices may impact risks of deductive disclosure (for example, Abbot¹), as well as development of stronger technical and legal privacy frameworks are critical to creating accessible AI technologies.

Error

Many people with disabilities need to trust and rely on the output of an AI system without the ability to verify the output (for example, a person who is blind relying on the output of a computer vision system). Our recent study found that people who were blind were overtrusting of an AI image captioning system, even when the output made little sense;⁹ we also found that incorrect output from a computer vision system negatively impacted the understanding of blind users to an extent that could not be corrected even with human assistance.¹³ Precision/recall trade-offs for AI systems may need to be recalibrated for vulnerable user populations—for example, the level of confidence a computer vision system must have to offer a label of an image for retrieval by sighted users of an image search engine may be quite different than the level of confidence the same system would need to offer a label of a scene to a blind pedestrian, where errors may have safety consequences. Er-

ror metrics output by AI systems must also be intelligible to end users. Translating mathematical confidence values to user-actionable information is an open challenge for AI that is particularly critical to users with disabilities.

Expectation Setting

The language used to describe AI technologies in the scientific literature, the popular press, and advertising materials often creates unreasonable expectations of such systems' current capabilities to lay users.⁸ For example, an article announcing that machine translation systems have reached "human parity" for very specific tasks on very specific data sets may lead the general public to make incorrect assumptions about the current state of the art of such AI in the open world. Terminology, demos, articles, and marketing materials that lead the public to misunderstand the capabilities of AI systems are particularly ethically problematic in the case of sensitive user populations, such as people with disabilities, whose quality of life may be fundamentally changed by such advances. In the domain of healthcare, bodies like the FDA regulate the kinds of statements that can be made about the efficacies of various treatments, yet no such regulations exist around promises made regarding the capabilities of apps and algorithms that may impact users' health, well-being, and ability to perform activities of daily living.

Simulated Data

As the need to create inclusive data sets is increasingly recognized, some technologists have used simulation to generate synthetic data; for instance, one approach has been to digitally modify or generate data to create variation not originally present in a dataset.¹⁵ Simulating disability is generally not recommended,¹⁴ since studies have found the data generated by users simulating disabilities (for example, a sighted person wearing a blindfold) is not the same as that from people who are truly disabled (that is, a person who has been blind for years will be much more skilled than a person who has just put on a blindfold). Further, disability simulation has been found to create negative impressions of the capabilities of people with disabilities.¹¹ However, lack of data is a tension that makes simulation appealing to

The deployment of AI systems in the open world creates important questions surrounding the social acceptability of new technologies.

many technologists. For example, AAC (augmentative and alternative communication) systems that are used by many people with severe speech and motor disabilities are often extremely slow to operate; intelligent language prediction can greatly increase communication bandwidth for AAC users. However, status quo prediction models are often trained on publicly available corpora such as news articles, whose linguistic structure may not represent the typical speech patterns of the target end users. To address this problem, researchers created a simulated AAC speech corpus by asking workers on Amazon's Mechanical Turk service to imagine what they might say if they were disabled;¹⁷ the resulting corpus is skewed toward stereotyped phrases such as "Can I have some water please?" or "Who will drive me to the doctor's office tomorrow?", despite the fact that actual AAC users wish to talk about interests and hobbies beyond their healthcare needs.⁶ Creating guidelines and best practices around the use of simulation for developing and testing AI systems for people with disabilities is important for ensuring that systems are not trained on non-representative data. Solutions addressing the aforementioned *privacy* challenges, which may prevent some people with disabilities from contributing data to AI efforts, may be particularly important for enabling inclusive data set creation that obviates the need for data simulation.

Social Acceptability

The deployment of AI systems in the open world creates important ques-

tions surrounding the social acceptability of new technologies. Of particular concern are questions regarding technologies' impact on indirect stakeholders, including issues of privacy and fairness. As an example of how privacy concerns influence social acceptability, consider the case of wearable technologies that may be capturing visual and audio data for algorithmic processing, such as Google Glass, which faced a public backlash due to privacy concerns. However, recent research indicates that people are more tolerant of such systems if the technology is being used by a person with disabilities.¹² Technologists, ethicists, and lawmakers must grapple with whether and how rules regarding the use of AI systems might change depending on an end user's disability status, and, if such differential uses of technology are deemed acceptable, how such distinctions could be facilitated in practice.

Conclusion

AI technologies offer great promise for people with disabilities by removing access barriers and enhancing users' capabilities. However, ethical considerations must carefully guide the development, deployment, and discussion around such technologies. These considerations regarding inclusivity, bias, privacy, error, expectation setting, simulated data, and social acceptability apply to all users, but are particularly nuanced and salient when considering the large potential benefits and large potential risks of AI systems for people with disabilities. While legal regulation may address some of these considerations in the future, it is unlikely to keep pace with the changing technological landscape. Educating our next generation of innovators is of paramount importance; emerging ethics curricula for computer science students should include content such as sociological concepts from the field of disability studies and inclusive design approaches from HCI. It is also vital to ensure that people with disabilities have a voice in technological innovation by making our educational institutions and workplaces more inclusive so as to increase the representation of people with disabilities among computer science professionals. As technologists, it is our responsibility to pro-

actively address these issues in order to ensure people with disabilities are not left behind by the AI revolution. □

References

1. Abbot, J. Local standards for anonymization practices in health, wellness, accessibility, and aging research at CHI. In *Proceedings of CHI 2019*; <https://dl.acm.org/citation.cfm?id=3300692>
2. Brady, E.P. Visual challenges in the everyday lives of blind people. In *Proceedings of CHI 2013* (2013); <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/chi2013-vizwiz.pdf>
3. Gebru, T. et al. Datasheets for datasets. arXiv, v3, (July 2018); <https://arxiv.org/abs/1803.09010>
4. Guo, A. et al. Toward fairness in AI for people with disabilities: A research roadmap. In *Proceedings of the ASSETS 2019 Workshop on AI Fairness for People with Disabilities*; <https://arxiv.org/pdf/1907.02227.pdf>
5. Gurari, D. et al. VizWiz grand challenge: Answering visual questions from blind people. *CVPR 2018*; <https://arxiv.org/abs/1802.08218>
6. Kane, et al. At times avuncular and cantankerous, with the reflexes of a mongoose: Understanding self-expression through augmentative and alternative communication devices. In *Proceedings of CSCW 2017* (2017); <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/10/aacselfexpression-1.pdf>
7. Lazar, J. The potential role of U.S. consumer protection laws in improving digital accessibility for people with disabilities. *U. Pa. J.L. & Soc. Change* 22, (2019), 185; <https://scholarship.law.upenn.edu/jlasc/vol22/iss3/3/>
8. Lipton, Z.C. and Steinhardt, J. Troubling trends in machine learning scholarship. *ICML 2018: Machine Learning Debates*. (2018); <https://arxiv.org/pdf/1807.03341.pdf>
9. MacLeod, H. et al. Understanding blind people's experiences with computer-generated captions of social media images. In *Proceedings of CHI 2017*; (2017); <http://aka.ms/captiontrust>
10. Morris, M.R., et al. With most of it being pictures now, I rarely use it: Understanding Twitter's evolving accessibility to blind users. In *Proceedings of CHI 2016*; <https://dl.acm.org/citation.cfm?id=2858116>
11. Nario-Redmond, M.R., Gospodinov, D., and Cobb, A. Crip for a day: The unintended negative consequences of disability simulations. *Rehabilitation Psychology*, 62, 3 (Aug. 2017), 324–333; <http://psycnet.apa.org/doiLanding?doi=10.1037%2Frep0000127>
12. Proffitt, H. et al. The AT effect: How disability affects the perceived social acceptability of head-mounted display use. In *Proceedings of CHI 2016*. (2016); <https://dl.acm.org/citation.cfm?id=2858130>
13. Salisbury, E., Kamar, and Morris, M.R. Toward scalable social alt text: Conversational crowdsourcing as a tool for refining vision-to-language technology for the blind. In *Proceedings of HCOMP 2017*. (2017); <http://aka.ms/hcompalttext>
14. Sears, A. and Hanson, V. Representing users in accessibility research. In *Proceedings of CHI 2011*; <https://dl.acm.org/citation.cfm?doi=1978942.1979268>
15. Simonite, T. Some startups use data to train AI. *Wired* (Apr. 25, 2018); <https://www.wired.com/story/some-startups-use-fake-data-to-train-ai/>
16. Trewin, S. AI fairness for people with disabilities: Point of view. arXiv, 2018; preprint arXiv:1811.10670.
17. Vertanen, K. and Kristensson, P.O. The imagination of crowds: Conversational AAC language modeling using crowdsourcing and large data source. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (2011), 700–711; <http://www.actweb.org/anthology/D11-1065>
18. White, R.W. et al. Detecting neurodegenerative disorders from Web search signals. *npj Digital Medicine* 1, 8 (Apr. 2018); <https://www.nature.com/articles/s41746-018-0016-6>

Meredith Ringel Morris (merrie@microsoft.com) is a Senior Principal Researcher at Microsoft Research, where she leads the Ability team in conducting research that combines advances in HCI and AI to address the technology needs of people with disabilities. She is also an affiliate Professor in The Paul G. Allen School of Computer Science & Engineering and in The Information School at the University of Washington, Seattle, WA, USA.

Copyright held by author.

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

Chipping away at Moore's Law.

BY JESSIE FRAZELLE

Commit to Memory

CPUs ARE MADE up of billions of teeny, tiny transistors. Transistors are electrical gates that can be switched on and off individually. Since each transistor can be in two distinct states (on or off), it can store two different numbers: zero and one. With billions of transistors, a chip can store billions of zeros and ones, and almost as many ordinary numbers and characters. The smaller the transistor, the less power is required for a chip to function.

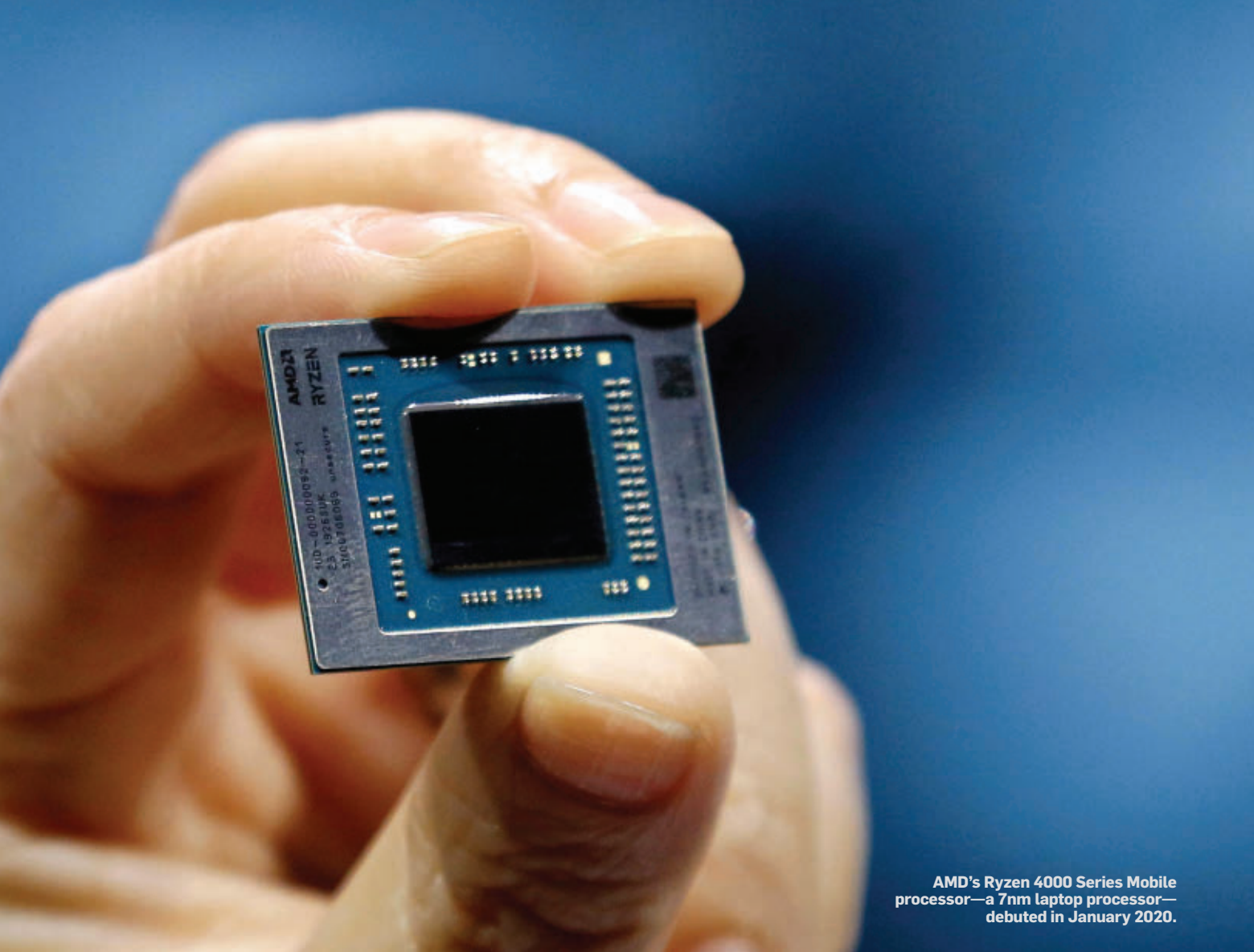
You might have heard of AMD's 7nm chips or Intel's 10nm chips. A nanometer (nm) is a useful metric for judging how powerful a CPU is since it denotes the measurement of transistor size (or linewidth).

Smaller transistors can do more calculations without overheating, which makes them more power

efficient. It also allows for smaller die sizes, which reduce costs and can increase density, allowing more cores per chip. The silicon wafers that chips are made of vary in purity, and none are perfect, which means every chip has a chance of having imperfections that differ in effect. Manufacturers can limit the effect of imperfections by using *chipllets*.

Traditionally, chip manufacturers connect two monolithic CPUs together in a multichip module (MCM). An MCM is a package with pins where multiple ICs (integrated circuits, or chips), semiconductor dies, and/or other components are integrated. This is usually done on a unifying substrate, so when the MCM is in use it can be treated as if it were one large chip. An MCM is sometimes referred to as a *hybrid IC*.

Chipllets are the individual ICs that make up an MCM. They provide a way to minimize the challenges of building



AMD's Ryzen 4000 Series Mobile processor—a 7nm laptop processor—debuted in January 2020.

with cutting-edge transistor technology. Spawning out of DARPA, the idea of chiplets has existed since the 1980s.²

The smallest transistors are also the most expensive and difficult to design and manufacture. In processors made up of chiplets, the most cutting-edge technology can be reserved for the pieces of a design where the investment will pay off the most. Then other chiplets manufactured using more reliable and economical techniques can be combined with the latest-technology chiplets into the same package. Intel does this with its Foveros Project, which combines, by way of stacking, 10nm chiplets for power-efficient activities and 14nm chiplets that serve higher power functions.¹³ The section on 7nm chips later in this article will provide more examples of this, although you can undoubtedly anticipate that 7nm chiplets are combined with larger transistor chiplets into one package.

Smaller pieces of silicon are also inherently less prone to manufacturing defects and have a reduced risk of getting destroyed by dust particles during semiconductor fabrication. Using chiplets over monolithic die architecture guarantees no single core on a package will act as a single point of failure for the MCM since there are multiple individual cores. Manufacturers can also use a process known as *binning* to select the best cores to be paired with each other.

Both AMD and Intel are breaking down monolithic processors into chiplets that are connected on a multichip module. AMD was the first in the mass-market CPU industry to move to chiplets when it announced its third-generation Ryzen CPU.⁷ This improves performance and reduces costs since splitting a large monolithic IC into smaller chiplets allows for more transistors split across multiple chiplets, more ICs per silicon wafer, and better

yields since smaller dies have less risk of defects or manufacturing errors.

In his 1965 paper, Gordon Moore wrote, “It may prove to be more economical to build large systems out of smaller functions, which are separately packaged and interconnected.”¹⁰ This is very true for chiplets. Lisa Su of AMD gave a talk at the International Electron Devices Meeting in 2017¹⁴ on how it split up its 32-core EPYC server-class chip onto four 8-core chiplets.¹⁹ Doing so resulted in a 10% overhead for the added I/O to the four chiplets. Since it could fit 9% more of the smaller dies onto a wafer than larger dies, however, it was able to reduce the overall cost because of the improvements in die yield.

Chiplet Interconnects

There are a few different ways manufacturers are interconnecting chiplets to form MCMs, most of which are proprietary to each vendor. Think of

die-to-die interconnect (or chiplet interconnect) as the interface connecting the various chiplets. A chiplet is not just a piece of hardware; it is also a controller and a physical layer of software (PHY). The die-to-die interconnect is the controller and the PHY for a chiplet. When designing die-to-die interconnects, power is the main constraint. Low power is table stakes because any additional power is considered overhead.


AMD calls its die-to-die interconnect Infinity Fabric, or IF (https://en.wikichip.org/wiki/amd/infinity_fabric). IF is not just a die-to-die interconnect but a processor-to-processor interconnect. It is made up of two parts: Infinity SDF (Scalable Data Fabric) and SCF (Scalable Control Fabric). SDF is the primary way data flows throughout the system. This can connect memory controllers, PCIe devices, USB hubs, and other peripherals. This interface is PHY level. The SCF, on the other hand, handles communication for controls such as power management, thermal controls, security, and tests.

Because most die-to-die interconnects are proprietary, the benefits of building with chiplets as if they are Lego blocks has not entirely come to fruition. Adding functionality such as an FPGA (field-programmable gate array) or an accelerator from a third party requires adopting the proprietary die-to-die interconnect.


As a response and solution to this problem, Intel open sourced its Advanced Interface Bus (AIB), which is a die-to-die interconnect standard that enables system design in a modular way with libraries containing chiplet IP (intellectual property) blocks.⁸ (The specification is on GitHub.¹) It is PHY level like AMD's SCF, but it will take a lot more than just Intel working on this problem for it to be solved for the wider industry. This effort is ongoing as a part of the Open Compute Project's subgroup ODSA (Open Domain-Specific Architecture).¹¹

7nm Chips

Moore's Law observed that the number of transistors on a chip doubles every year, while the costs are halved. While this theory has held for a long time, it has been slowing down lately. In the late 1990s and early 2000s, transistors



When designing die-to-die interconnects, power is the main constraint. Low power is table stakes because any additional power is considered overhead.



shrunk in size by half every two years, leading to massive improvements on a regular schedule. Making transistors smaller and smaller has gotten more complicated, however, and there has not been a transistor shrink from Intel since 2014 when it promised 10nm,^{3,18} 7nm is possible but has not yet been done by Intel. The slowdown of transistor size in semiconductor devices led to the development of multichip modules and other innovations, since semiconductor designers are always looking for new ways to provide increased compute capabilities.

Taiwan Semiconductor Manufacturing Company (TSMC) began production of 256-Mbit SRAM (static random-access memory) chips using a 7nm process in 2017.¹⁶ Later in 2018, Samsung and TSMC began mass production of 7nm devices.¹⁵ The Apple A12 Bionic, a 7nm chip that went mainstream, was released at Apple's September 2018 event.⁶ Technically, Huawei had announced its own 7nm processor, the Kirin 980, on August 31, 2018,¹² before the A12 Bionic, but the A12 was first to market and was released to consumers before the Kirin 980. Both the Kirin 980 and Apple A12 Bionic are manufactured by TSMC.

AMD released its Rome family of processors for servers and data centers, which are based on TSMC's 7nm node and feature up to 64 cores and 128 threads. AMD also released its Matisse family of consumer desktop processors with 16 cores and 32 threads. Technically, Intel does have one 7nm part obtained with its acquisition of Barefoot Networks: the Tofino 2 chip, an Ethernet-switch ASIC, based on TSMC's 7nm node.

Shrinking transistor size is not just about performance, though; it has huge implications for mobile, data-center, and laptop chips. Compared with 14nm, 7nm can achieve 25% more performance under the same power, or the same performance for half the power. This means longer battery life for laptops and phones, and more power-efficient data centers, with the same performance. For smaller devices, since twice as much performance can effectively fit into the limited power target, this translates into much more powerful chips. Apple did this with its A12 Bionic chip.

You might have noticed a trend in that all these 7nm chips rely on TSMC for manufacturing. Companies using these chips are focusing the 7nm technology in small special-purpose chips in order to increase the yield and effectiveness since it is in high demand. For example, AMD's second-generation EPYC is built as nine die packages with eight 7nm CCD (complex core die) chiplets, each with up to eight cores, surrounding a 14nm I/O die.

The question is, if 7nm parts are in such high demand, why is TSMC the only foundry manufacturing them?

Semiconductor Foundries

TSMC is the world's largest semiconductor foundry. Most of its production is in Taiwan, with only one plant in Shanghai, China. (It is interesting to note where each foundry has facilities since at the time of this writing the CoVID-19 coronavirus and U.S./China tariffs are relevant, and some might be curious about their effect on the semiconductor industry.) Aside from TSMC, other common foundries include Globalfoundries and United Microelectronics Corporation (UMC). Globalfoundries, which has plants in Singapore, Germany, and the U.S., had set out to manufacture 7nm chips and stopped because of the expense.⁹ After Globalfoundries publicly stated it would not manufacture 7nm chips, UMC followed and said it would not manufacture anything under 14nm because of the expense.¹⁷ UMC has plants in Taiwan, Singapore, and the two Chinese cities of Suzhou and Xiamen.


What makes 7nm manufacturing so capital intensive? Foundries need to increase their capital expenditures to deal with the technical difficulties of decreasing the size of transistors, among which lithography remains the biggest hurdle. Lithography, also known as photolithography, is the process used in microfabrication to pattern transistor circuits onto silicon. Shining light on certain regions of silicon can create a specific pattern. Equipment for lithography is very expensive, preventing foundries from investing further in research and development to shrink transistor size.

Lithography technology has been improving with immersion lithog-

raphy and multipatterning but with significant increases in the number of masks and processes. When additional exposures are needed to create complicated patterns on silicon, the costs of masks increase immensely. According to eBeam Initiative's survey,⁵ the average number of masks per mask set has reached 76 for 7nm–10nm process node, and the number reaches more than 100 for manufacturers. Because of the increase in mask cost, 7nm manufacturing processes have been outside the economical scope for most small and medium-sized design houses.

While phones, servers, graphics, and data centers all benefit from enhanced computing performance and power efficiency, the cost to manufacture bleeding-edge chips is increasing significantly. Therefore, not all foundries can handle the economics of 7nm chip manufacturing. The companies that have built products with 7nm chips include AMD, Apple, Samsung, Huawei, NVIDIA, and Barefoot Networks. What all these companies have in common is the motivation to be on the bleeding edge of technology in order to remain or become a market leader. They all benefit from economies of scale for their large production needs, so they can share the costs of masks, design, and manufacturing. Foundries without such high demand from a mass market cannot afford to invest in advanced technology since the risk might outweigh the reward.

Future

What does all this mean for the future? It is likely semiconductor designers will continue to innovate without needing to shrink transistor size, as happened with multichip modules. Hopefully, the increase in demand for 7nm chips will give other foundries the economic incentive to enter the market, so the industry is not tied to TSMC alone. This is easier said than done, of course. I also hope by the end of reading this, you have learned that modern-day CPUs are actually just a few chiplets connected together into one package. The ability to construct packages of chiplets gives a Lego-like building power to people for designing their own MCMs and innovating faster. 

Related articles on queue.acm.org

Getting Gigascale Chips

Shekhar Borkar

<https://queue.acm.org/detail.cfm?id=957757>

CPU DB: Recording Microprocessor History

Andrew Danowitz, et al.

<https://queue.acm.org/detail.cfm?id=2181798>

Reconfigurable Future

Mark Horowitz

<https://queue.acm.org/detail.cfm?id=1388771>

References

- Advanced Interface Bus (AIB) die-to-die hardware open source; <https://github.com/chipsalliance/aib-phy-hardware>.
- Coldewey, D. DARPA project aims to make modular computers out of chiplets. *Techcrunch*; <https://tcrn.ch/2VSpe8j>
- Cuttress, I. Intel's 10nm Cannon Lake and Core i3-8121U deep dive review. *Anandtech*, 2019; <http://bit.ly/2PMx9A6>
- eBeam Initiative. Mask makers' survey, 2019; https://www.ebeam.org/docs/2019-mask-maker-survey_en.pdf.
- eBeam Initiative. Survey: 2019 eBeam Initiative mask makers' survey results. *Semiconductor Engineering*, 2020; <http://bit.ly/320Ad44>.
- Frumusanu, A. The iPhone XS and XS Max review: Unveiling the silicon secrets. *Anandtech*, 2018; <http://bit.ly/2wp5LBm>.
- Hruska, J. Chiplets are both solution to and symptom of a larger problem. *Extremetech*, 2019; <http://bit.ly/2uMmjCG>
- Leibson, S. Intel releases royalty-free high-performance AIB interconnect standard to spur industry's chiplet adoption and grow the ecosystem. Intel Programmable Logic, 2019; <https://intel.ly/32L6gSp>.
- McGregor, J. Globalfoundries' change in strategy pays off. *Forbes*, 2019; <http://bit.ly/2PLoLRf>.
- Moore, G. Cramming more components onto integrated circuits. *Electronics* 38, 8 (1965); <https://intel.ly/2VtGq2>
- Open Compute Project. Open Domain-Specific Architecture subgroup, 2020; <https://www.opencompute.org/wiki/Server/ODSA>.
- Savov, V. Huawei promises its 7nm Kirin 980 processor will destroy the Snapdragon 845. *The Verge*, 2018; <http://bit.ly/32KM00G>.
- Savov, V. Intel unveils Foveros 3D chip stacking and new 10nm 'chiplets'. *The Verge*, 2018; <http://bit.ly/2vstSz0>
- Schor, D. IEDM 2017: AMD's grand vision for the future of HPC. *WikiChip Fuse*, 2017; <https://fuse.wikichip.org/news/523/iedm-2017-amds-grand-vision-for-the-future-of-hpc/>.
- Tallis, B., Shilov, A. Samsung starts mass production of chips using its 7nm EUV process tech. *Anandtech*, 2018; <http://bit.ly/2IcPXUO>.
- TSMC. 7nm technology; <https://www.tsmc.com/english/dedicatedFoundry/technology/7nm.htm>.
- Wang, L. UMC not to rejoin race to develop 7nm technology. *Taipei Times*, 2018; <http://www.taipetimes.com/News/biz/archives/2018/09/04/2003699736>.
- Williams, C. Intel TOCK BLOCK: 10nm Cannonlake delayed to 2017, bonus 14nm Kaby Lake to '16. *The Register*; 2015; https://www.theregister.co.uk/2015/07/16/intel_10nm_14nm_plans/.
- Yeric, G. Three dimensions in 3DIC, Part 1. *Arm Research*, 2018; <https://community.arm.com/developer/research/b/articles/posts/three-dimensions-in-3dic-part-1>.

Jessie Frazelle is the co-founder and Chief Product Officer of the Oxide Computer Company, Emeryville, CA, USA. Before that, she worked on various parts of Linux including containers as well as the Go programming language.

© 2020 ACM 0001-0782/20/6 \$15.00

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

Leveraging expectations for better communication.

BY THOMAS A. LIMONCELLI

Communicate Using the Numbers 1, 2, 3, and More

THE HUMAN BRAIN reacts differently to lists of different lengths. When you align what you say with what the human brain expects, you communicate more effectively. In this column I'll explain how to leverage the way the brain reacts to various quantities to make your speaking and writing more effective.

People often use lists of various sizes when communicating. I might have 2 reasons for supporting the new company strategy. I might tell you my 3 favorite programming languages. I might make a presentation that describes 4 new features. There is 1 vegetable that I like more than any other.

The length of the list affects how the audience interprets what is being said. Not aligning with what the human brain expects is like swimming upstream. Given the choice, why would anyone do that?

The remainder of this column focuses on 3 things:

- ▶ Part A. Lists of size 2, 3, and 4 or more;
- ▶ Part B. Lists of size 1; and,
- ▶ Part C. Giving your audience a mental roadmap.

Part A. Lists of Size 2, 3, and 4 or More

As I mentioned earlier, people often use lists of various sizes when speaking and writing:

- ▶ I'm going to discuss 2 carmakers: Toyota and GM.
- ▶ My 3 favorite bands are: the Beatles, the Rolling Stones, and Led Zeppelin.
- ▶ I ate 9 types of fruit last week: apples, oranges, bananas, grapes, pineapples, cherries, watermelons, blueberries, and pomegranates.

The length of the list affects how the audience interprets what is being said.

Lists of 2 encourage comparison. When I tell you that I'm going to talk about 2 carmakers, your brain naturally starts comparing them. One is Japanese, the other is not. One is doing financially better than the other, and so on.

Your brain can't help but compare 2 items. If you want to encourage your audience to compare and contrast, present a list of 2 items. If you want to discourage comparison, add a third item as a decoy.

Lists of 3 are interesting for a different reason. Typically, people can hold 3 things in their minds at once. If you are going to talk about many things but it is important that the audience hold 3 specific items in their heads as you discuss them, make sure your list is 3 or fewer.

Suppose I'm going to talk about rock bands for an hour. During that time, I'm going to be switching the discussion from one band to the



next. I need the audience to hold onto the names of these bands or they will get confused. Therefore, I better not talk about more than 3 bands. More than that is too much cognitive load. The audience will get confused, bored, or worse.

What about lists of 4 or more? These are difficult for the human brain to remember. In fact, if I told you that last week I ate 9 types of fruit: apples, oranges, bananas, grapes, pineapples, cherries, watermelons, blueberries, and pomegranates ... it is pretty reasonable to assume you won't remember the entire list.

What you will remember, however, is that the list is very long. If my intention is to convince the audience that I like a lot of variety, listing many fruits will drive that point home. Next week you won't remember specifically what I liked, but you will remember I liked many different types of fruit.

If I want the audience to remember exactly which fruits I like, I better make it a shorter list.

What if I really, really, really want people to remember a list longer than 3? The best you can do is to group the items into 3 categories. People will remember the categories, not the individual items.

Part B. Lists of Size 1

If you really want people to remember something, the best approach is to talk about that 1 thing, and nothing else. Any added information is a distraction. Murphy's law will result in people remembering the distraction, not what you wanted them to remember.

While I mentioned 9 types of fruit earlier, the vegetable I would like to tell you about is a list of size 1: broccoli.

I want you to remember that I like broccoli. Therefore, I'm going to mention only 1 vegetable: broccoli.

As an engineer, I may feel compelled to let you know that I like other

vegetables too. I'm not exclusive to broccoli. I'm not going to mention those other vegetables, however. That would be a distraction.

I want the list to have 1 item on it, and I'm going to repeat that 1-item list a lot.

I like broccoli. That's what I want you to remember. In the past month I've had broccoli 6 different ways: raw, steamed, sautéed, roasted, blanched, and in a garlic sauce. I love broccoli. I really love broccoli.

Next week when I ask you about this you will remember that I like broccoli because I had the self-control not to mention any other vegetables. I'm not going to tell you about other vegetables that I like such as... No, no, no. Not gonna do it.

As an interesting aside: When I was younger, I didn't like broccoli. In March 1990, however, it became public that President George Bush disliked the vegetable,¹ and that inspired me to give it another try. It turns out, I liked it.

Getting people to remember something is like getting children to eat their vegetables. As Josh Lyman reminds us, "It helps if there's nothing else on their plates."² If you want the audience to remember something, don't distract them with things you don't want them to remember.

Part C. Giving Your Audience a Mental Roadmap

The third and final piece of advice when communicating with numbers is to tell the audience the quantity ahead of time. Don't make them count.

In fact, you should reveal the count, then list the items, then discuss each item. Don't make the audience wait for the count or the individual items. Surprise endings are good for mystery novels but not in speaking or writing.

By doing this, you give your audience a mental roadmap of what is to come. As you present the full details, people mentally check off the items and see the progress.

Because you gave them the count, they are prepared to react in 1 of 3 ways: compare, hold in their head, or memorize.

Begin by saying, "I'm going to tell you about 3 rock bands: the Beatles, the Rolling Stones, and Led Zeppelin," then talk about each band in the order

you listed them, and conclude by telling your audience that these were your thoughts about the Beatles, the Rolling Stones, and Led Zeppelin.

This is what your grade-school teachers meant when they taught you, "Tell them what you are going to say, then say it, then tell them what you told them." When my teachers told me this, I thought it was some boring old formula, but I didn't understand the cognitive importance. It took me years to realize that by following this rule, you are creating a mental roadmap for your audience, filling in the details, then reminding them what you want them to remember.

I find this particularly useful during long presentations for 2 reasons: It fortifies the roadmap, and it helps wake up bored people in the audience.

If I've painted the roadmap at the start, as I introduce each item people can check off the previous item in their heads and mentally prepare themselves for the next item. I pause for dramatic effect, and use a slide that looks like a chapter title page. This is called bookending your topics. The pause gives people a moment to catch up with what you were saying. Call it the palate cleanser of presentations.

The pause also has the effect of waking up the bored members of the audience. (Not that you or I have ever given a boring presentation, but this bit of advice might help your colleagues and friends.)

Consider people in the audience who are bored with a speaker's presentation. Once they are bored, they have checked out and aren't listening. A pause will startle them and grab their attention. Maybe they are bored by the speaker's discussion of the Beatles, but they notice the pause and hear the speaker say, "Now that we've learned about the Beatles, let's talk about Led Zeppelin." Now alert, they give the speaker a second chance and start paying attention again.

For these reasons, I often structure my talks and papers as a "Top 5" or "Top 10" list. This gives the audience a mental roadmap, it creates periodic opportunities to wake up bored people, and it helps everyone gauge how much progress I've made through the list. Instead of wondering, "How much longer is this stupid talk going to be?," they are thinking, "OK, this is

point number 3; maybe number 4 will be more interesting." Or "He's on the third point, but he has used 80% of his time; this is going to be a disaster." Or maybe they are thinking, "If this is point number 9, I can leave now and be first in line for the conference dinner."


Conclusion

These tricks really work. That's why I told you I ate 9 types of fruit before I started naming each one. I wanted to prepare your brain to hear a long list. If you weren't prepared, your brain would freak out and worry that I would never stop listing names of fruit.

That's why I told you I had 1 favorite vegetable, broccoli, and never mentioned any other vegetable.

That's why I told you this column has 3 parts. Because now you know it is done.

Postscript

Technically, this article made 6 major points, but because I grouped them and presented them as 3, you were more likely to hold them in your head and follow along. By including this point in a postscript, I've extended the total to 7 points. By going against the rules, I've created a contrast that makes this last point stand out. Only break these rules once per talk, and do it with purpose. 

Related articles on queue.acm.org

Time Protection in Operating Systems and Speaker Legitimacy Detection

Adrian Colyer

<https://queue.acm.org/detail.cfm?id=3344779>

A Conversation with Steve Furber

<https://queue.acm.org/detail.cfm?id=1716385>

The Importance of a Great Finish

Kate Matsudaira

<https://queue.acm.org/detail.cfm?id=3293457>

References

1. Dowd, M. 'I'm President,' so no more broccoli! *New York Times* (Mar. 23, 1990); <https://www.nytimes.com/1990/03/23/us/i-m-president-so-no-more-broccoli.html>.
2. Manchester: Part II. *The West Wing*. Season 3, ep. 3 (aired Oct. 17, 2001).

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

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

Publish Your Work Open Access With ACM!

ACM offers a variety of Open Access publishing options to ensure that your work is disseminated to the widest possible readership of computer scientists around the world.



Please visit ACM's website to learn more about ACM's innovative approach to Open Access at:
<https://www.acm.org/openaccess>



Association for
Computing Machinery

Lessons learned from Meltdown's exploitation of the weaknesses in today's processors.

BY MORITZ LIPP, MICHAEL SCHWARZ, DANIEL GRUSS, THOMAS PRESCHER, WERNER HAAS, JANN HORN, STEFAN MANGARD, PAUL KOCHER, DANIEL GENKIN, YUVAL YAROM, MIKE HAMBURG, AND RAOUL STRACKX

Meltdown: Reading Kernel Memory from User Space

Memory isolation is a cornerstone security feature in the construction of every modern computer system. Allowing the simultaneous execution of multiple mutually distrusting applications at the same time on the same hardware, it is the basis of enabling secure execution of multiple processes on the same machine or in the cloud. The operating system is in charge of enforcing this isolation, as well as isolating its own kernel memory regions from other users.

Given its central role on modern processors, the isolation between the kernel and user processes is backed by the hardware, in the form of a supervisor bit that determines whether code in the current

context can access memory pages of the kernel. The basic idea is that this bit is set only when entering kernel code and it is cleared when switching to user processes. This hardware feature allows operating systems to map the kernel into the address space of every process, thus supporting very efficient transitions from the user process to the kernel (for example, for interrupt handling) while maintaining the security of the kernel memory space.

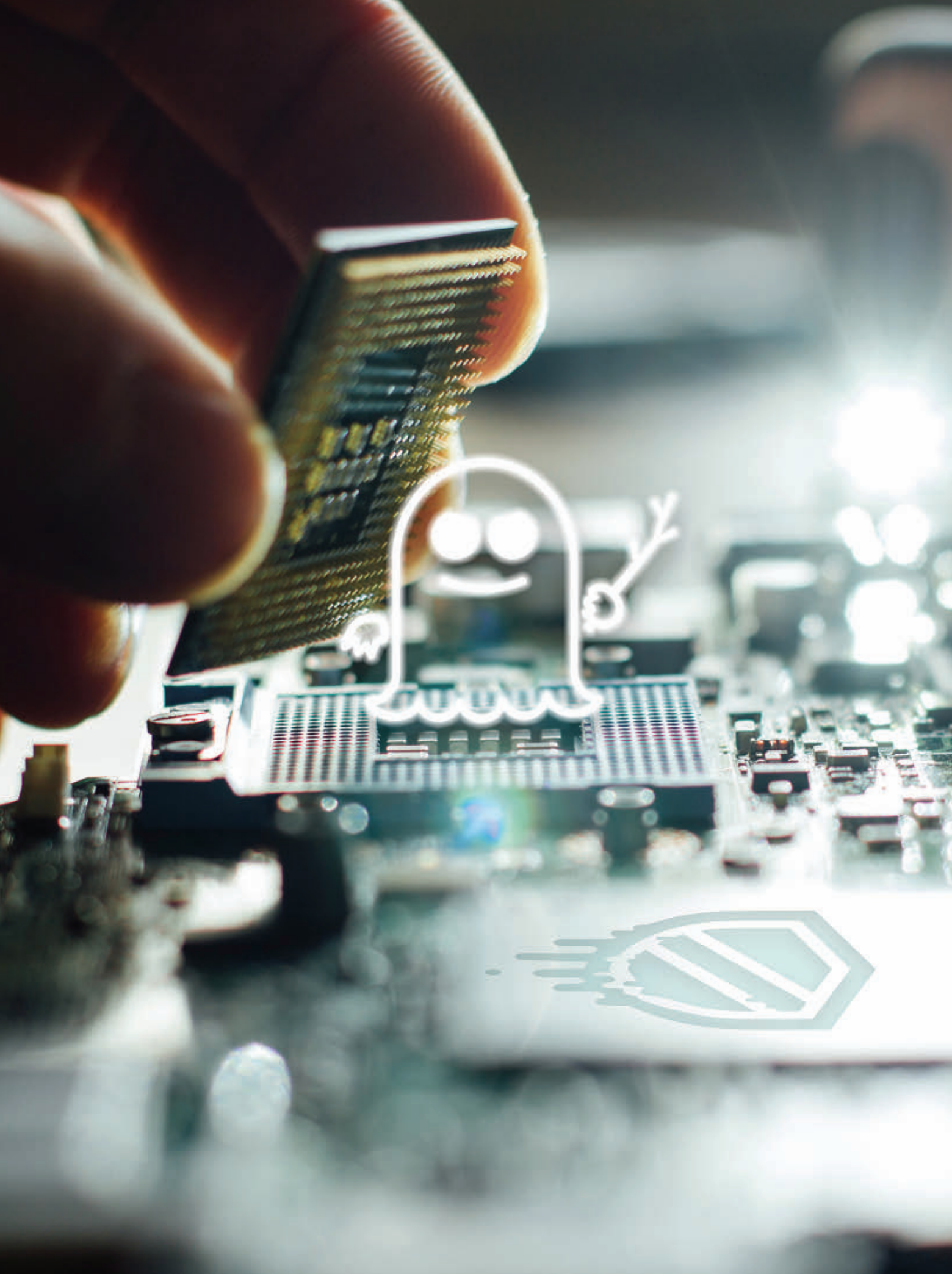
This article presents Meltdown, a novel attack that exploits a vulnerability in the way the processor enforces memory isolation.

Root cause. At a high level, the root cause of Meltdown's simplicity and strength are side effects caused by *out-of-order execution*, which is an important performance feature of modern processors designed to overcome latencies of busy execution units (for example, a memory fetch unit waiting for data arrival from memory). Rather than stalling the execution, modern processors run operations out-of-order, that is, they look ahead and schedule subsequent operations on available execution units of the core.

While this feature is beneficial for performance, from a security perspective, one observation is particularly significant. Some CPUs allow an unprivileged process to load data from a privileged (kernel or physical) address into a temporary register, delaying exception handling to later stages. The

>> key insights

- **Out-of-order execution, a performance feature in most modern processors, is not as harmless as was hitherto believed.**
- **Meltdown breaks memory isolation by exploiting out-of-order execution and a delayed permission check on some Intel, IBM, and ARM CPUs, allowing unprivileged attackers to leak privileged data using microarchitectural side channels.**
- **Since the discovery of Meltdown, many other transient-execution vulnerabilities have followed. As the root cause of all of these lays in the underlying hardware, the design of modern CPUs must be modified to fully mitigate them.**



Summary of processors affected by Meltdown.

| Arch. | Description |
|---------|---|
| x86 | Most Intel and VIA processors are vulnerable. AMD processors are not. |
| Arm | Cortex-A75 and SoCs based on it are vulnerable. Some proprietary Arm-based processors, including some Apple and Samsung cores, are also vulnerable. Arm Cortex-A72, Cortex-A57 and Cortex-A15 are vulnerable to a Variant 3a of Meltdown. Other Arm cores are not known to be vulnerable. |
| Power | All IBM Power architecture processors are vulnerable. |
| z/Arch. | IBM z10, z13, z14, z196, zEC12 are vulnerable. |
| SPARC | V9-based SPARC systems are not vulnerable. Older SPARC processors may be impacted. |
| Itanium | Itanium processors are not vulnerable. |

CPU even allows performing further computations based on this register value, such as using it as an index to an array access. When the CPU finally realizes the error, it reverts the results of this incorrect transient execution, discarding any modifications to the program state (for example, registers). However, we observe that out-of-order memory lookups influence the internal state of the processor, which in turn can be detected by the program. As a result, an attacker can dump the entire kernel memory by reading privileged memory in an out-of-order execution stream, and subsequently transmit-

ting the data via a covert channel, for example, by modulating the state of the cache. As the CPU's internal state is not fully reverted, the receiving end of the covert channel can later recover the transmitted value, for example, by probing the state of the cache.

Threat model. To mount Meltdown, the adversary needs the ability to execute code on a vulnerable machine. Executing code can be achieved through various means, including hosting in cloud services, apps in mobile phones, and JavaScript code in website. Vulnerable machines include personal computers and servers featuring a large

range of processors (see the accompanying table). Furthermore, while countermeasures have been introduced to both operating systems and browsers, these only became available after the disclosure of Meltdown.

Impact. Three properties of Meltdown combine to have a devastating effect on the security of affected systems. First, exploiting a hardware vulnerability means the attack does not depend on specific vulnerabilities in the software. Thus, the attack is generic and, at the time of discovery, affected all existing versions of all major operating systems. Second, because the attack only depends on the hardware, traditional software-based protections, such as cryptography, operating system authorization mechanisms, or antivirus software, are powerless to stop the attack. Last, because the vulnerability is in the hardware, fixing the vulnerability requires replacing the hardware. While software-based countermeasures for Meltdown have been developed, these basically avoid using the vulnerable hardware feature, incurring a significant performance hit.

Evaluation. We evaluated the attack on modern desktop machines and laptops, as well as servers and clouds. Meltdown is effective against all major operating systems (including Linux, Android, OS X and Windows), allowing an unprivileged attacker to dump large parts of the physical memory. As the physical memory is shared among all other tenants running on the same hardware, this may include the physical memory of other processes, the kernel, and in the case of paravirtualization, the memory of the hypervisor or other co-located instances. While the performance heavily depends on the specific machine—for example, processor speed, TLB and cache sizes, and DRAM speed—we can dump arbitrary kernel and physical memory at a speed of 3.2KiB/s to 503KiB/s.

Countermeasures. While not originally intended to be a countermeasure for Meltdown, KAISER,⁶ developed initially to prevent side-channel attacks targeting KASLR, also protects against Meltdown. Our evaluation shows that KAISER prevents Meltdown to a large extent. Consequently, we stress it is of utmost importance to deploy KAISER on all operating systems immediately.

Figure 1. On Unix-like 64-bit systems, a physical address (blue) which is mapped accessible to the user space is also mapped in the kernel space through the direct mapping.

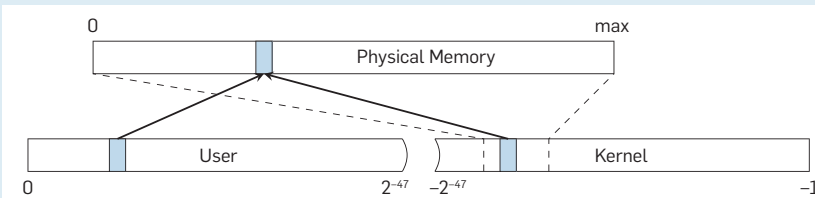
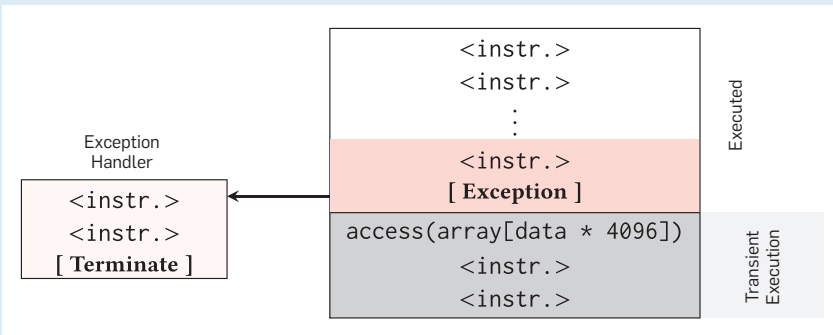


Figure 2. If an executed instruction causes an exception, control flow is diverted to an exception handler. Subsequent instruction may already have been partially executed, but not retired. Architectural effects of this transient execution are discarded.



Fortunately, during the responsible disclosure window, the three major operating systems (Windows, Linux, and OS X) implemented variants of KAISER and recently rolled out these patches.


Spectre attacks and follow-up works. Meltdown was published simultaneously with the Spectre Attack,¹⁷ which exploits a different CPU performance feature, called speculative execution, to leak confidential information. Meltdown is distinct from Spectre in several ways, notably that Spectre requires tailoring to the victim process's software environment but applies more broadly to CPUs and is not mitigated by KAISER. Since the publication of Meltdown and Spectre, several prominent follow-up works exploited out of order and speculative execution mechanisms to leak information across other security domains.^{1,14,16,18,22,24,25,27} See Canella et al.⁴ for a survey.

At the time of writing, Microarchitectural Data Sampling (MDS) is the most recent line of attacks,^{3,21,26} which exploit speculative and out-of-order execution in order to leak information across nearly all possible security domains. Finally, while some of the attacks discussed in this section have been mitigated, additional work is required to mitigate others as well as future yet-to-be discovered CPU vulnerabilities.


Background

Here, we provide background on out-of-order execution, address translation, and cache attacks.

The microarchitecture. The Instruction Set Architecture (ISA) of a processor is the interface it provides to the software it executes. The ISA is typically defined as some state, which mostly consists of the contents of the architectural registers and the memory, and a set of instructions that operate on this state. The implementation of this interface consists of multiple components, collectively called the microarchitecture. The microarchitecture maintains a state that extends the architectural state of the processor as defined by the ISA, adding further information required for the operation of the microarchitectural components. While changes in the microarchitectural state do not affect the logical behavior of the program, they may affect



Our evaluation shows that KAISER prevents Meltdown to a large extent. Consequently, we stress it is of utmost importance to deploy KAISER on all operating systems immediately.



its performance. Thus, the microarchitectural state of the processor depends on prior software execution and affects its future behavior, creating the potential for untraditional communication channels.⁵

Out-of-order execution. Out-of-order execution¹⁰ is an optimization technique that increases the utilization of the execution units of a CPU core. Instead of processing instructions strictly in sequential program order, waiting for slow instructions to complete before executing subsequent instructions, the CPU executes them as soon as all required resources are available. While the execution unit of the current operation is occupied, other execution units can run ahead. Hence, instructions execute in parallel as long as their results follow the architectural definition.

Address spaces. To isolate processes from each other, CPUs support virtual address spaces where virtual addresses are translated to physical addresses. The operating system kernel plays a key role in managing the address translation for processes. Consequently, the memory of the kernel must also be protected from user processes. Traditionally, in segmented architectures,¹⁰ the kernel had its own segments that were not accessible to user processes.

In modern processors, a virtual address space is divided into a set of pages that can be individually mapped to physical memory through a multilevel page translation table. In addition to the virtual to physical mapping, the translation tables also specify protection properties that specify the allowed access to the mapped pages. These properties determine, for example, whether pages are readable, writable, and executable. A pointer to the currently used translation table is held in a dedicated CPU register. During a context switch, the operating system updates this register to point to the translation table of the next process, thereby implementing a per-process virtual address space, allowing each process to only reference data that belongs to its virtual address space. To reduce the cost of consulting the translation tables, the processor caches recent translation results in the Translation Lookaside Buffer (TLB).

While the TLB reduces the cost of address translation, its contents must be flushed when changing address space. Modern systems include the kernel memory within the address space of every process to avoid the cost of flushing the TLB on every switch between the user program and the kernel. Following an idea first introduced in the VAX/VMS system, the page table also includes a protection bit that indicates whether the page belongs to the kernel or to the user program. Kernel pages are only accessible when the processor executes with high privileges, that is, when executing the kernel. Thus, user processes are not able to read or to modify the contents of the kernel memory.

To aid in memory management, many modern operating systems directly map (a part of) the physical memory in the kernel's part of the virtual address space at a fixed location m (see Figure 1). A physical address p can then be assessed through virtual address $p + m$.

Cache attacks. To speed-up memory accesses and address translation, the CPU contains small memory buffers, called caches, that store frequently used data. CPU caches hide slow memory accesses by buffering frequently used data in smaller and faster internal memory. Modern CPUs have multiple levels of caches that are either private per core or shared among them. Address space translation tables are also stored in memory and, thus, also cached in the regular caches.

Cache side-channel attacks exploit the timing differences introduced by the caches. Several cache attack techniques have been proposed and demonstrated in the past, including Prime+Probe¹⁹ and Flush+Reload.²⁸ Flush+Reload attacks work on a single cache line granularity. These attacks exploit the shared, inclusive last-level cache. An attacker frequently flushes a targeted memory location using the `clflush` instruction. By measuring the time it takes to reload the data, the attacker determines whether the memory location was loaded into the cache by another process since the last `clflush`. The Flush+Reload attack has been used for attacks on various computations, for example, cryptographic algorithms,²⁸ Web server function

calls,²⁹ user input,⁹ and kernel addressing information.⁷

A special use case of a side-channel attack is a covert channel. Here, the attacker controls both the part that induces the side effect and the part that measures the side effect. This can be used to leak information from one security domain to another while bypassing any boundaries existing on the architectural level or above. Both Prime+Probe and Flush+Reload have been used in high-performance covert channels.⁸

A Toy Example

We start with a toy example, which illustrates that out-of-order execution can change the microarchitectural state in a way that leaks information.

Triggering out-of-order execution. Figure 2 illustrates a simple code sequence first raising an (unhandled) exception and then accessing an array. The exception can be raised through any mean, such as accessing an invalid memory address, performing a privileged instruction in user code, or even division by zero. An important property of an exception, irrespective of its cause, is that the control flow does not follow program order to the code following the exception. Instead, it jumps to an exception handler in the operating system kernel. Thus, the code in our toy example is expected not to access the array because the exception traps to the kernel and terminates the application before the access is performed. However, we note the access instruction after the exception has no data dependency on the trapping instruction. Hence, due to out-of-order execution, the CPU might execute the access before triggering the exception. When the exception is triggered, instructions executed out of order are not retired and, thus, never have architectural effects. However, instructions executed out-of-order do have side-effects on the microarchitecture. In particular, the contents of the memory accessed after the exception in Figure 2 are fetched into a register and also stored in the cache. When the out-of-order execution is reverted (that is, the register and memory contents are never committed), the cached memory contents survive reversion and remain in the cache for

some time. We can now leverage a microarchitectural side-channel attack, such as Flush+Reload,²⁸ to detect whether a specific memory location is cached, thereby making the affected microarchitectural state visible.

Observing out-of-order execution. The code in Figure 2 accesses a memory address that depends on the value of data. As data is multiplied by 4,096, data accesses to array are scattered over the array with a distance of 4KiB (assuming a 1B data type for array). Thus, there is an injective mapping from the value of data to a memory page, that is, different values for data never result in accesses to the same page. Consequently, if a cache line of a page is cached, we can determine the value of data.

Figure 3 shows the result of Flush+Reload measurements iterating over all of the pages of array, after executing the out-of-order snippet in Figure 2 with `data = 84`. Although the array access should not have happened due to the exception, we can clearly see that the index which would have been accessed is cached. Iterating over all pages (for example, in the exception handler) shows a cache hit for page 84 only. This demonstrates that instructions that are only executed out-of-order but are never retired, change the microarchitectural state of the CPU. Later, we show how we modify this toy example to leak an inaccessible secret.

Building Blocks of the Attack

The toy example noted earlier illustrates that side effects of out-of-order execution can modify the microarchitectural state to leak information. While the code snippet reveals the data value passed to a cache side channel, we want to show how this technique can be leveraged to leak otherwise inaccessible secrets. Here, we want to generalize and discuss the necessary building blocks to exploit out-of-order execution for an attack.

Overview of Meltdown. Assume an adversary that targets a secret value which is kept somewhere in physical memory. The full Meltdown attack leaks this value using two building blocks, as illustrated in Figure 4. The first building block of Meltdown is to make the CPU execute one or more transient instructions, that is, instructions that do not occur during regular

execution. The second building block of Meltdown is to transfer the microarchitectural side effect of the transient instruction sequence to an architectural state to further process the leaked secret. Later, we describe methods to lift a microarchitectural side effect to an architectural state using covert channels.

Executing transient instructions.

The first building block of Meltdown is the execution of transient instructions, which are executed out-of-order and leave measurable side effects. We focus on transient instructions that follow an illegal access to addresses that are mapped within the attacker’s process such as user-inaccessible kernel space addresses. In general, accessing such user-inaccessible addresses triggers an exception, which typically terminates the application. Because we want to measure the microarchitectural state of the processor after the transient execution, we want to avoid terminating the process. We now present several approaches the attacker can use to cope with the exception.

Fork-and-crash. A trivial approach is to fork the attacking application before accessing the invalid memory location that terminates the process and only access the invalid memory location in the child process. The CPU executes the transient instruction sequence in the child process before crashing. The parent process can then recover the secret by probing the microarchitectural state.

Exception handling. Next, it is also possible to install a signal handler that is executed when a certain exception occurs, for example, a segmentation violation. This allows the attacker to issue the instruction sequence and prevent the application from crashing, reducing the overhead as no new process has to be created.

Exception suppression via TSX. An alternative approach to deal with exceptions is to prevent them from being raised in the first place. Intel’s Transactional Synchronization Extensions (TSX) defines the concept of transaction, which is a sequence of instructions that execute atomically, that is, either all of the instructions in a transaction are executed, or none of them is. If an instruction within the transaction fails, already executed in-

structions are reverted, but no exception is raised. By wrapping the code in Figure 5 in such a TSX transaction, the exception is suppressed. Yet, the microarchitectural effects of transient execution are still visible. Because suppressing the exception is significantly faster than trapping into the kernel for handling the exception, and continuing afterwards, this results in a higher channel capacity.

Exception suppression via branch predictor. Finally, speculative execution issues instructions that might not occur in the program order due to a branch misprediction. Thus, by forcing a misprediction that speculatively executes the invalid memory access, we can achieve transient execution of both the invalid memory access and the instructions following it, without triggering

an exception. See Kocher et al.¹⁷ for further details on speculative execution and transient instructions.

Building a covert channel. The second building block of Meltdown is lifting the microarchitectural state, which was changed by the transient instruction sequence, into an architectural state (as shown in Figure 4). The transient instruction sequence can be seen as the transmitting end of a microarchitectural covert channel. The receiving end of the covert channel receives the microarchitectural state change and deduces the secret from the state. Note the receiver is not part of the transient instruction sequence and can be a different thread or process, for example, the parent process in the fork-and-crash approach.

A cache-based covert channel. Previ-

Figure 3. Even if a memory location is only accessed during out-of-order execution, it remains cached. Iterating over the 256 pages of array shows one cache hit, exactly on the page that was accessed during the out-of-order execution.

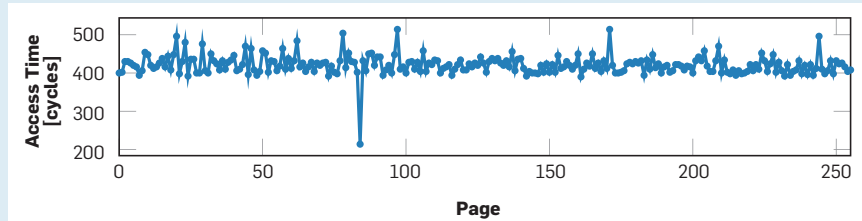


Figure 4. The Meltdown attack uses exception handling or suppression, for example, TSX, to run a series of transient instructions.

These transient instructions obtain a (persistent) secret value and change the microarchitectural state of the processor based on this secret value. This forms the sending part of a microarchitectural covert channel. The receiving side reads the microarchitectural state, lifts it to architectural, and recovers the secret value.

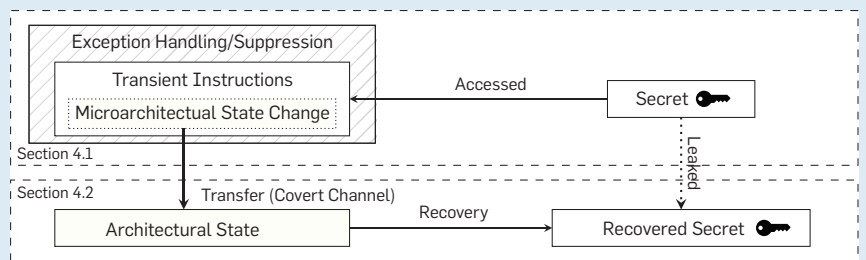


Figure 5. The core of Meltdown.

An inaccessible kernel address is moved to a register, raising an exception. Subsequent instructions are executed out of order before the exception is raised, leaking the data from the kernel address through the indirect memory access.

```


1 mov al, byte [rcx] ; rcx = kernel address
2 shl rax, 0xc
3 mov rbx, qword [rbx + rax] ; rbx = probe array
    
```

ous works^{8,19,28} have demonstrated that the microarchitectural state of the cache can be easily lifted into a architectural state. We, therefore, employ these techniques for our covert channel. Specifically, we use Flush+Reload,²⁸ as it allows building a fast and low-noise covert channel.


After accessing a user-inaccessible secret value, the transient instruction sequence executes the cache covert channel transmitter, performing a memory access using the secret value as part of the address. As explained earlier, this address remains cached for subsequent accesses, and survives the soon-to-be-raised exception. Thus, part of the cache state depends on the secret value and lifting this state to an architectural state leaks the secret value.

Recovering the leaked value. The covert channel receiver can then monitor whether an address has been loaded into the cache by measuring the access time to the address. For example, the sender can transmit a ‘1’-bit by accessing an address which is loaded into the monitored cache, and a ‘0’-bit by not accessing such an address. Using multiple different cache lines, as in our toy example, allows transmitting multiple bits at once. For every one of the 256 different byte values, the sender accesses a different cache line. By performing a Flush+Reload attack on all of the 256 possible cache lines, the receiver can recover a full byte rather than just one bit of secret value. However, since the Flush+Reload attack takes much longer (typically several hundred cycles) than the transient instruction sequence, transmitting only a single bit at once is more efficient. The attacker can choose the bit to transmit by shifting and masking the secret value accordingly.

Using other covert channels. Note the covert channel part is not limited to cache-based microarchitectural channels. Any instruction (or sequence) that influences the microarchitectural state of the CPU in a way that can be observed from a user process can be used to build a covert channel transmitter. For example, to send a ‘1’-bit the sender could issue an instruction (or sequence), which occupies a certain execution port such as the ALU. The receiver measures the latency when executing an instruction (sequence) on



Any instruction (or sequence) that influences the microarchitectural state of the CPU in a way that can be observed from a user process can be used to build a covert channel transmitter.



the same execution port. A high latency implies that the sender sends a ‘1’-bit, whereas a low latency implies that the sender sends a ‘0’-bit. The advantage of the Flush+Reload cache covert channel is the noise resistance and the high transmission rate.⁸ Furthermore, with cache architectures commonly used in current CPUs, different memory access latencies can be observed from any CPU core,²⁸ that is, rescheduling events do not significantly affect the covert channel.

The Meltdown Attack

Here, we present Meltdown, a powerful attack enabling arbitrary kernel memory (typically including the entire physical memory) to be read from an unprivileged user program, comprised of the building blocks presented earlier. First, we discuss the attack setting to emphasize the wide applicability of this attack. Second, we present an attack overview, showing how Meltdown can be mounted on both Windows and Linux on personal computers, on Android on mobile phones as well as in the cloud. Finally, we discuss a concrete implementation of Meltdown allowing to dump memory with 3.2KiB/s to 503KiB/s.

Attack setting. In our attack, we consider personal computers and virtual machines in the cloud. In the attack scenario, the attacker can execute arbitrary unprivileged code on the attacked system, that is, the attacker can run any code with the privileges of a normal user. The attacker targets secret user data, for example, passwords and private keys, or any other valuable information. Finally, we assume a completely bug-free operating system. That is, we assume the attacker does not exploit any software vulnerability to gain kernel privileges or to leak information.

Attack description. Meltdown combines the two building blocks discussed previously. At a high level, Meltdown consists of three steps:

- ▶ Step 1: Reading the secret. The content of an attacker chosen memory location, which is inaccessible to the attacker, is loaded into a register.
- ▶ Step 2: Transmit the secret. A transient instruction accesses a cache line based on the secret content of the register.

► **Step 3: Receive the secret.** The attacker uses Flush+Reload to determine the accessed cache line and hence the secret stored at the chosen memory location.

By repeating these steps for different memory locations, the attacker can dump the kernel memory, including the entire physical memory.

Figure 5 shows a typical implementation of the transient instruction sequence and the sending part of the covert channel, using x86 assembly instructions. Note this part of the attack could also be implemented entirely in higher level languages such as C. In the following, we discuss each step of Meltdown and the corresponding code line in Figure 5.

Step 1: Reading the secret. Recall that modern operating systems map the kernel into the virtual address space of every process. Consequently, a user process can specify addresses that map to the kernel space. As discussed, in parallel with performing the access, the CPU verifies that the process has the required permission for accessing the address, raising an exception if the user tries to reference a kernel address. However, when translating kernel addresses, they do lead to valid physical addresses the CPU can access, and only the imminent exception due to illegal access protects the contents of the kernel space. Meltdown exploits the out-of-order execution feature of modern CPUs, which execute instructions for a small-time window between the illegal memory access and the subsequent exception.

Line 1 of Figure 5 loads a byte value from the target kernel address, pointed to by the RCX register, into the least significant byte of the RAX register represented by AL. The CPU executes this by fetching the MOV instruction, decoding and executing it, and sending it to the reorder buffer for retirement. As part of the execution, a temporary physical register is allocated for the updated value of architectural register RAX. Trying to utilize the pipeline as much as possible, subsequent instructions (lines 2–3) are decoded and sent to the reservation station holding the instructions while they wait to be executed by the corresponding execution units.

Thus, when the kernel address is accessed in Line 1, it is likely that the CPU already issues the subsequent instructions as part of the out-of-order execution, and these instructions wait in the reservation station for the content of the kernel address to arrive. When this content arrives, the instructions can begin their execution. Furthermore, processor interconnects¹³ and cache coherence protocols²³ guarantee that the most recent value of a memory address is read, regardless of the storage location in a multi-core or multi-CPU system.

When the processor finishes executing the instructions, they retire in-order, and their results are committed to the architectural state by updating the register renaming tables, that is, the mapping of architectural to physical registers.¹⁰ During the retirement, any interrupts and exceptions that occurred while executing of the instruction are handled. Thus, when the MOV instruction that loads the kernel address (Line 1) is retired, the exception is registered, and the pipeline is flushed to eliminate all results of subsequent instructions which were executed out of order. However, there is a race condition between raising this exception and our attack Step 2.

Step 2: Transmitting the secret. The instruction sequence from Step 1, which is executed out-of-order, is chosen such that it becomes a transient instruction sequence. If this transient instruction sequence is executed before the MOV instruction is retired, and the transient instruction sequence performs computations based on the secret, it can transmit the secret to the attacker.

As already discussed, we use cache attacks that allow building fast and low-noise covert channels using the CPU's cache. Thus, the transient instruction sequence has to encode the secret as a microarchitectural cache state, similar to our toy example.

We allocate a probe array in memory and ensure that no part of this array is cached. To transmit the secret, the transient instruction sequence performs an indirect memory access to an address which depends on the secret (inaccessible) value. Line 2 of Figure 5 shifts the secret value from Step 1 by 12 bits to the left, effectively multiplying it

by the page size of 4KiB. This ensures that accesses to the array have a large spatial distance from each other, preventing the hardware prefetcher from loading adjacent memory locations into the cache. Here, we read a single byte at once. Hence, our probe array is 256×4096 bytes long.

Line 3 adds the multiplied secret to the base address of the probe array, forming the target address of the covert channel. It then accesses this address, effectively bringing its content to the cache. Consequently, our transient instruction sequence affects the cache state based on the secret value that was read in Step 1.

Finally, since the transient instruction sequence in Step 2 races against raising the exception, reducing the runtime of Step 2 can significantly improve the performance of the attack. For instance, taking care that the address translation for the probe array is cached in the Translation Lookaside Buffer (TLB) increases the attack performance on some systems.

Step 3: Receiving the secret. In Step 3, the attacker recovers the secret value from Step 1 by implementing the receiving end of a microarchitectural covert channel that transfers the cache state (Step 2) back into an architectural state. As discussed, our implementation of Meltdown relies on Flush+Reload for this purpose.

When the transient instruction sequence of Step 2 is executed, exactly one cache line of the probe array is cached. The position of the cached cache line within the probe array depends only on the secret, read in Step 1. To recover the secret, the attacker iterates over all 256 pages of the probe array and measures the access time to the first cache line of each page. The number of the page containing the cached cache line corresponds directly to the secret value.

Dumping physical memory. Repeating all three steps of Meltdown, an attacker can dump the entire memory by iterating over all addresses. However, as the memory access to the kernel address raises an exception that terminates the program, we use one of the methods discussed earlier to handle or suppress the exception. Furthermore, because most major operating systems also map the entire physical memory

into the kernel address space in every user process, Meltdown can also read the entire physical memory of the target machine.

Evaluation

In this section, we evaluate Meltdown and the performance of our proof-of-concept implementation.^a We discuss the information Meltdown can leak, and evaluate the performance of Meltdown, including countermeasures. Our results are consistent across vulnerable laptops, desktop PCs, mobile phones, and cloud systems.

Leakage and environments. We evaluated Meltdown on various operating systems with and without patches. On all unpatched operating systems, Meltdown can successfully leak kernel memory. We detail the Linux, Windows and Android evaluation here.

Linux. We successfully evaluated Meltdown on multiple versions of the Linux kernel, from 2.6.32 to 4.13.0, without the patches introducing the KAISER mechanism. On all of these Linux kernel versions, the kernel is mapped into the address space of user processes, but access is prevented by the permission settings for these addresses. As Meltdown bypasses these permission settings, an attacker can leak the complete kernel memory if the virtual address of the kernel base is known. Since all major operating systems (even 32 bit as far as possible) also map the entire physical memory into the kernel address space, all physical memory can also be read.

Before kernel 4.12, kernel address space layout randomization (KASLR) was not enabled by default.²⁰ Without KASLR, the entire physical memory was directly mapped starting at address 0xffff 8800 0000 0000. On such systems, an attacker can use Meltdown to dump the entire physical memory, simply by reading from virtual addresses starting at 0xffff 8800 0000 0000. When KASLR is enabled, Meltdown can still find the kernel by searching through the address space. An attacker can also de-randomize the direct physical map by iterating through the virtual address space.

On newer systems KASLR is usually active by default. Due to the large size

and the linearity of the mapping the randomization of the direct physical map is usually 7 bits or lower. Hence, the attacker can test addresses in steps of 8GB, resulting in a maximum of 128 memory locations to test. Starting from one discovered location, the attacker can again dump the entire physical memory.

Windows. We successfully evaluated Meltdown on a recent Microsoft Windows 10 operating system, last updated just before patches against Meltdown were rolled out. In line with the results on Linux, Meltdown can leak arbitrary kernel memory on Windows. This is not surprising, since Meltdown does not exploit any software issues, but is caused by a hardware issue.

In contrast to Linux, Windows does not map the physical memory directly in the kernel's virtual address space. Instead, a large fraction of the physical memory is mapped in the paged pools, nonpaged pools, and the system cache. Windows does map the kernel into the address space of every application. Thus, Meltdown can read kernel memory that is mapped in the kernel address space, that is, any part of the kernel which is not swapped out, and any page mapped in the paged and non-paged pool, and in the system cache.

Note that there are physical pages which are mapped in one process but not in the (kernel) address space of another process. These physical pages cannot be attacked using Meltdown. However, most of the physical memory is accessible through Meltdown.

We could read the binary code of the Windows kernel using Meltdown. To verify that the leaked data is indeed kernel memory, we first used the Windows kernel debugger to obtain kernel addresses containing actual data. After leaking the data, we again used the Windows kernel debugger to compare the leaked data with the actual memory content, confirming that Meltdown can successfully leak kernel memory.

Android. We successfully evaluated Meltdown on a Samsung Galaxy S7 mobile phone running LineageOS Android 14.1 with a Linux kernel 3.18.14. The device is equipped with a Samsung Exynos 8 Octa 8890 SoC consisting of an ARM Cortex-A53 CPU with four cores as well as an Exynos M1 "Mongoose" CPU with four cores.² While we

were not able to mount the attack on the Cortex-A53 CPU, we successfully mounted Meltdown on Samsung's custom cores. Using exception suppression via branch misprediction as described previously, we successfully leaked a predefined string using the direct physical map located at the virtual address 0xffff fbf c000 0000.

Containers. We evaluated Meltdown in containers sharing a kernel, including Docker, LXC, and OpenVZ and found that the attack can be mounted without any restrictions. Running Meltdown inside a container allows to leak information not only from the underlying kernel but also from all other containers running on the same physical host. The commonality of most container solutions is that every container uses the same kernel, that is, the kernel is shared among all containers. Thus, every container has a valid mapping of the entire physical memory through the direct-physical map of the shared kernel. Furthermore, Meltdown cannot be blocked in containers, as it uses only memory accesses. Especially with Intel TSX, only unprivileged instructions are executed without even trapping into the kernel. Thus, the confidentiality guarantee of containers sharing a kernel can be entirely broken using Meltdown. This is especially critical for cheaper hosting providers where users are not separated through fully virtualized machines, but only through containers. We verified that our attack works in such a setup, by successfully leaking memory contents from a container of a different user under our control.


Meltdown performance. To evaluate the performance of Meltdown, we leaked known values from kernel memory. This allows us to not only determine how fast an attacker can leak memory, but also the error rate, that is, how many byte errors to expect. The race condition in Meltdown has a significant influence on the performance of the attack, however, the race condition can always be won. If the targeted data resides close to the core, for example, in the L1 data cache, the race condition is won with a high probability. In this scenario, we achieved average reading rates of 552.4KiB/s on average ($\sigma = 10.2$) with an error rate of 0.009% ($\sigma = 0.014$) using exception suppression.

a <https://github.com/IAIK/meltdown/>


sion on the Core i7-8700K over 10 runs over 10 seconds. On the Core i7-6700K we achieved on average 515.5KiB/s ($\sigma = 5.99$) with an error rate of 0.003 % on average ($\sigma = 0.001$) and 466.3KiB/s on average ($\sigma = 16.75$) with an error rate of 11.59 % on average ($\sigma = 0.62$) on the Xeon E5-1630. However, with a slower version with an average reading speed of 137KiB/s, we were able to reduce the error rate to zero. Furthermore, on the Intel Core i7-6700K if the data resides in the L3 data cache but not in the L1, the race condition can still be won often, but the average reading rate decreases to 12.4KiB/s with an error rate as low as 0.02 % using exception suppression. However, if the data is uncached, winning the race condition is more difficult and, thus, we have observed average reading rates of less than 10B/s on most systems. Nevertheless, there are two optimizations to improve the reading rate: First, by simultaneously letting other threads prefetch the memory locations⁷ of and around the target value and access the target memory location (with exception suppression or handling). This increases the probability that the spying thread sees the secret data value in the right moment during the data race. Second, by triggering the hardware prefetcher within our own thread through speculative accesses to memory locations of and around the target value before the actual Meltdown attack. With these optimizations, we can improve the reading rate for uncached data to 3.2KiB/s on average.

For all of the tests, we used Flush+Reload as a covert channel to leak the memory as described, and Intel TSX to suppress the exception. For brevity, we omit the results of evaluating exception suppression using conditional branches. See Kocher et al.¹⁷ for further information.

Limitations on ARM and AMD. We verified that some of the processors listed as not affected (see the table) are not vulnerable. Specifically, we experimented with some AMD and Arm-based processors and were unable to reproduce Meltdown on those. We nevertheless note that for both ARM and AMD, the toy example works reliably, indicating that out-of-order execution generally occurs and instructions past illegal memory accesses are also performed.



Meltdown is a security issue rooted in hardware. Thus, to fully mitigate Meltdown, the hardware of modern CPUs must be modified.



Real-world Meltdown exploit. To demonstrate the applicability of Meltdown, we show a possible real-world exploit that allows an attacker to steal the secret key used to store sensitive data. We look at VeraCrypt,¹² a freeware solution that allows users to protect sensitive data using file or hard disk encryption. We note that VeraCrypt is just an example, and any software that keeps its key material in main memory can be attacked in a similar manner.

Attack scenario. In our scenario, the attacker gained access to the encrypted container or the encrypted hard drive of the victim. Without the secret key, the attacker is unable to decrypt the data, which is therefore secure. However, as VeraCrypt keeps the secret key in the main memory, relying on memory isolation to protect the key from unauthorized access, an attacker can use Meltdown to recover the key. A naïve approach for that is to dump the entire physical memory of the computer and search in it. However, this approach is not practical. Instead, we show the attacker can recover the page mapping of the VeraCrypt process and, thus, limit the amount of data to leak. For our experiments, we used VeraCrypt 1.22.

Breaking KASLR. As KASLR is active on the attacked system, the attacker must first de-randomize the kernel address space layout to access internal kernel structures and arbitrary physical addresses using the direct mapping. By locating a known value inside the kernel, for example, the Linux banner, the randomization offset can be computed as the difference between the found address and the non-randomized base address. The Linux KASLR implementation only has an entropy of 6 bits,¹⁵ hence there are only 64 possible randomization offsets, making this approach practical.

Locating the VeraCrypt process. Linux manages processes in a linked list whose head is stored in the `init_task` structure. The structure's address is at a fixed offset that only depends on the kernel build and does not change when packages are loaded. Each entry in the task list points to the next element, allowing easy traversal. Entries further include the process id of the task, its name and the root of the multi-level page table, allowing the attacker to

identify the VeraCrypt process and to locate its page map.

Extracting the encryption key. The attacker can now traverse the paging structures and read the memory used by the process directly. VeraCrypt stores the DataAreaKey in a SecureBuffer in the VolumeHeader of a Volume. If ASLR is not active, the attacker can directly read the key from the offset where the key is located. Otherwise, the attacker searches the memory for a suitable pointer from which it can track the data structures to the stored key.

With the extracted key, the attacker can decrypt the container image, giving full access to the stored sensitive data. This attack does not only apply to VeraCrypt but to every software that keeps its key material stored in main memory.

Countermeasures

Fundamentally, Meltdown is a security issue rooted in hardware. Thus, to fully mitigate Meltdown, the hardware of modern CPUs must be modified. Indeed, since the original publication of Meltdown, Intel has released 9th-generation i-cores, which contain hardware mechanisms that mitigate Meltdown.


For older vulnerable hardware, lower performing software mitigations do exist. More specifically, Gruss et al.⁶ proposed KAISER, a kernel modification to not have the kernel mapped in the user space. While this modification was intended to prevent side-channel attacks breaking KASLR,^{7,11,15} it also prevents Meltdown, as it ensures there is no valid mapping to kernel space or physical memory available in user space. Since the publication of Meltdown, Kernel Page Table Isolation (which is an implementation of KAISER) has been adopted by all major operating systems.

Conclusion

Meltdown fundamentally changes our perspective on the security of hardware optimizations that change the state of microarchitectural elements. Meltdown and Spectre teach us that functional correctness is insufficient for security analysis and the micro-architecture cannot be ignored. They further open a new field of research to investigate the extent to which per-

formance optimizations change the microarchitectural state, how this state can be lifted into an architectural state, and how such attacks can be prevented. Without requiring any software vulnerability and independent of the operating system, Meltdown enables an adversary to read sensitive data of other processes, containers, virtual machines, or the kernel. KAISER is a reasonable short-term workaround to prevent large-scale exploitation of Meltdown until hardware fixes are deployed.

Acknowledgments. Several authors of this article found Meltdown independently, ultimately leading to this collaboration, and we thank everyone who helped make this collaboration possible. We thank Mark Brand from Google Project Zero, Peter Cordes and Henry Wong as well as Intel, ARM, Qualcomm, and Microsoft for feedback.

D. Gruss, M. Lipp, S. Mangard and M. Schwarz were supported by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No 681402). D. Genkin was supported by NSF awards #1514261 and #1652259, financial assistance award 70NANB15H328 from the U.S. Department of Commerce, NIST, the 2017-2018 Rothschild Postdoctoral Fellowship, and the Defense Advanced Research Project Agency (DARPA) under Contract #FA8650-16-C-7622. 

References

1. Bhattacharyya, A. et al. SMoTherSpectre: Exploiting speculative execution through port contention. In *Proceedings of 2019 CCS*, 785–800.
2. Burgess, B. Samsung Exynos M1 Processor. *IEEE Hot Chips* (2016).
3. Canella, C. et al. Fallout: Leaking data on Meltdown-resistant CPUs. In *Proceedings of 2019 CCS*.
4. Canella, C. et al. A systematic evaluation of transient execution attacks and defenses. *USENIX Sec* (2019), 249–266.
5. Ge, Q., Yarom, Y., Cock, D., and Heiser, G. A survey of microarchitectural timing attacks and countermeasures on contemporary hardware. *JCEN* 8, 1 (2018).
6. Gruss, D., Lipp, M., Schwarz, M., Fellner, R., Maurice, C., and Mangard, S. KASLR is Dead: Long Live KASLR. In *Proceedings of Intern. 2017 Symposium on Engineering Secure Software and Systems*. Springer, 161–176.
7. Gruss, D., Maurice, C., Fogh, A., Lipp, M., and Mangard, S. Prefetch side-channel attacks: Bypassing SMAP and Kernel ASLR. In *Proceedings of 2016 CCS*.
8. Gruss, D., Maurice, C., Wagner, K., and Mangard, S. Flush + Flush: A fast and stealthy cache attack. In *Proceedings of DIMVA*, 2016.
9. Gruss, D., Spreitzer, R., and Mangard, S. Cache template attacks: Automating attacks on inclusive last-level caches. In *Proceedings of USENIX Security Symposium*, 2015.
10. Hennessy, J.L., and Patterson, D.A. *Computer Architecture: A Quantitative Approach*, 5th Ed. Morgan

- Kaufmann, San Francisco, CA, USA, 2011.
11. Hund, R., Willems, C., and Holz, T. Practical timing side channel attacks against kernel space ASLR. S&P (2013).
12. IDRIX. VeraCrypt; <https://veracrypt.fr> 2018.
13. Intel. An introduction to the intel quickpath interconnect, Jan 2009.
14. Intel. Rogue system register read, 2018; <https://software.intel.com/security-software-guidance/software-guidance/>
15. Jang, Y., Lee, S., and Kim, T. Breaking kernel address space layout randomization with Intel TSX. In *Proceedings of 2016 CCS*.
16. Kiriansky, V., and Waldspurger, C. Speculative buffer overflows: Attacks and defenses, 2018; arXiv 1807.03757.
17. Kocher, P. et al. Spectre attacks: Exploiting speculative execution. S&P (2019).
18. Miller, M. Speculative store bypass, <https://blogs.technet.microsoft.com/srd/2018/05/21/analysis-and-mitigation-of-speculative-store-bypass/>
19. Osvik, D.A., Shamir, A. and Tromer, E. Cache attacks and countermeasures: The case of AES. In *Proceedings of 2006 CT-RSA*.
20. Phoronix. Linux 4.12 To Enable KASLR By Default; <http://bit.ly/2FVu0Xz>
21. Schwarz, M. et al. ZombieLoad: Cross-privilege-boundary data sampling. In *Proceedings of 2019 CCS*.
22. Schwarz, M., Schwarzl, M., Lipp, M., Masters, J., and Gruss, D. NetSpectre: Read arbitrary memory over network. In *Proceedings of ESORICS*, 2019.
23. Sorin, D.J., Hill, M.D., and Wood, D.A. *A Primer on Memory Consistency and Cache Coherence*. 2011.
24. Stecklina, J., and Prescher, T. LazyFP: Leaking FPU register state using microarchitectural side-channels, 2018; arXiv 1806.07480.
25. Van Bulck, J. et al. Foreshadow: Extracting the keys to the Intel SGX kingdom with transient out-of-order execution. *USENIX Sec.* (Aug. 2018).
26. van Schaik, S., Milburn, A., Österlund, S., Frigo, P., Maisuradze, G., Razavi, K., Bos, H., and Giuffrida, C. RIDL: Rogue in-flight data load. S&P (May 2019).
27. Weisse, O. et al. Foreshadow-NG: Breaking the virtual memory abstraction with transient out-of-order execution; <https://foreshadowattack.eu/foreshadow-NG.pdf> 2018.
28. Yarom, Y., and Falkner, K. Flush + Reload: A high resolution, low noise, L3 cache side-channel attack. In *Proceedings of USENIX Security Symposium*, 2014..
29. Zhang, Y., Juels, A., Reiter, M.K., and Ristenpart, T. Cross-tenant side-channel attacks in PaaS clouds. In *Proceedings of 2014 CCS*.

Moritz Lipp is a Ph.D. candidate at Graz University of Technology, Flanders, Austria.

Michael Schwarz is a postdoctoral researcher at Graz University of Technology, Flanders, Austria.

Daniel Gruss is an assistant professor at Graz University of Technology, Flanders, Austria.

Thomas Prescher is a chief architect at Cyberus Technology GmbH, Dresden, Germany.

Werner Haas is the Chief Technology Officer at Cyberus Technology GmbH, Dresden, Germany.

Jann Horn is a member of Project Zero at Google Project Zero, Zurich, Switzerland.

Stefan Mangard is a professor at Graz University of Technology, Flanders, Austria.

Paul Kocher is an entrepreneur and researcher focused on cryptography and data security, San Francisco, CA, USA.

Daniel Genkin is an assistant professor at the University of Michigan, Ann Arbor, MI, USA.

Yuval Yarom is a senior lecturer at the University of Adelaide and Data61, South Australia.

Mike Hamburg is a researcher in the Cryptography Research Division of Rambus, Sunnyvale, CA, USA.

Raoul Strackx is a senior engineer at Fortanix, Flanders, Belgium.

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

It's difficult to see the ecological impact of IT when its benefits are so blindingly bright.

BY ALAN BORNING, BATYA FRIEDMAN, AND NICK LOGLER

The 'Invisible' Materiality of Information Technology

THERE ARE SIGNIFICANT material impacts from extracting, processing, maintaining, and ultimately disposing of the materials used to support information technology, as well as from producing the energy used both by the devices in operation, as well as in their production and disposal. Yet these material impacts

are largely invisible and receive substantially less attention than discussions about the technical aspects and benefits of information technology. We use the term *materiality* to encompass all of these aspects and more—a comprehensive accounting of the ways in which information technology impinges on the physical world.

Consider as an example the Internet of Things (IoT). The number of IoT devices is growing rapidly, with projections by some analysts of 20 billion to 30 billion devices by year's end.¹¹ This sharp increase in the number of IoT devices, along with supporting infrastructure, will result in significant consumption of materials and energy and production of waste. Despite this, a recent U.S. Government Accountability Office report on IoT³⁶ has only a brief mention (two paragraphs in a 70-page

report) of the issue of electronic waste resulting from the increasing use of IoT technology, and nothing on the consumption of raw materials and energy. The top 10 results of an Internet search for "Internet of Things" shows a similar pattern: only two out of 10 of the results discussed any potential downsides with respect to materials, energy, and waste, and even within those two, there were five to 10 times more mentions of potential positives than negatives. And IoT is but one example among many.

Research, development, and uptake of computing and information technology has proceeded at an ever-accelerating rate, with only minimal consideration of material impacts, which might lead one to conclude that all is well. However, this great success carries with it an increasing negative mate-

rial impact. At one time, such impacts could safely be absorbed in the Earth's natural processes and ecosystems; but we are now in an era (sometimes aptly named the Anthropocene) in which the safe operating boundaries for many of these processes and ecosystems are being transgressed by human activity.³⁰

Information technology (IT) is a significant contributor to activities of these sorts.^{2,3,10,18} This is not to say that many of these activities are inappropriate—to the contrary, many are appropriate, and in some cases are making a positive contribution to limiting environmental impacts of human activities. What we are instead arguing is the material side is largely invisible. There are some important examples of grappling with issues around sustainability and the material side of IT,²⁰ but by and large the result of this invisibility is that discussions and debates about its positive versus negative material impacts are often simply not occurring.

Why *are* these material impacts largely invisible? Is this simply a result of highlighting where most of the attention of technologists, business people, consumers, and others is focused, or are there structural forces that more actively push toward minimizing the visibility and consideration of these implications? We bring a broad and long-term view on human values and technology (value sensitive design)^{9,15-17} to these questions and suggest that in fact there are such forces,

and they work together, reinforcing each other. Specifically, we call out five overarching categories of forces: disciplinary norms and practices of computer science, metaphor, utopian visions, visibility of hardware, and economics. Finally, we argue the computing and information community is uniquely positioned to respond to these challenges, both in substance and in public understanding, and explore what (if anything) might be done.

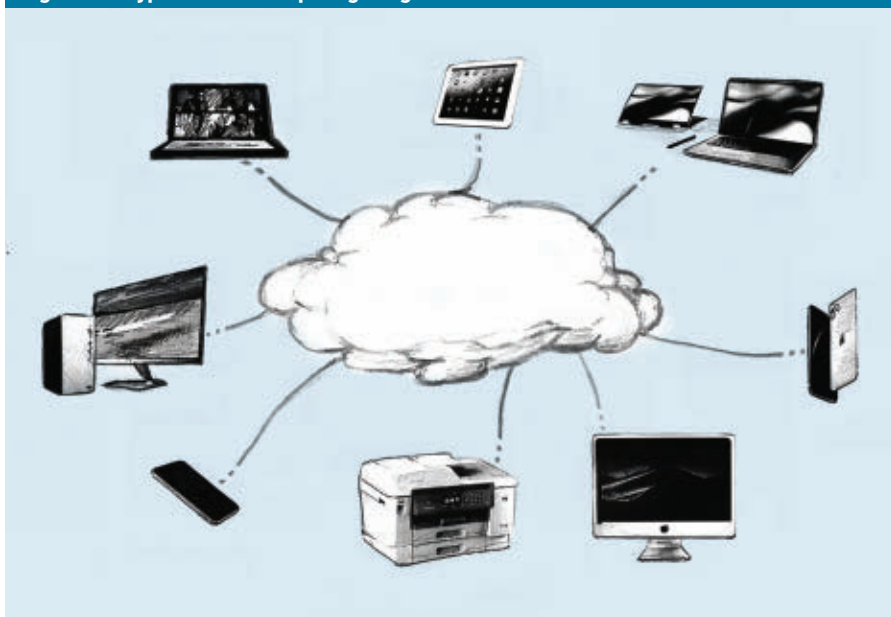
Disciplinary Norms and Practices of Computer Science

One force arises from certain key intellectual moves in computer science. Many parts of the discipline involve setting aside the materiality of computing, abstracting away the physical manifestations to concentrate on information and computation. These powerful intellectual moves allow us to concentrate on key aspects of the phenomenon under consideration and to ignore irrelevant or lower order ones. For example, in theoretical computer science, the concept of Turing equivalence captures the fact that many classes of computers have equivalent computational power (setting aside the limitations of finite memory), even though the machines might have totally different sorts of material realizations or might be purely theoretical constructs. Similarly, complexity theory encompasses results such as the time and space requirements to solve particular problems, independent of the physi-

cal machine on which the algorithm is being run. In programming languages, high-level languages abstract away the details of memory allocation and deallocation, let alone the physical manifestation of that memory; in computer networking, protocols often abstract away the physical substrates that implement the networks. There are enormous benefits flowing from these moves to both computer science as a field of research and to information technology as an economic sector—research results apply much more generally, for example, and existing protocols, APIs, and software can take advantage of different and improved hardware.

As with all abstractions and models, a key aspect of using them well is to understand when it is appropriate to simply use them, and when one needs to peer into the black box, that is, bring back into view some of the properties that were abstracted away. The vast majority of experienced programmers at some point will have needed to address issues of efficient use of memory, which is ultimately a consequence of the realization of the computation on physical devices. Similarly, skilled Web designers often need to consider download speeds for different possible layouts and choices of content. As a third and perhaps less familiar example, for applications that can tolerate inaccuracies, programmers can sometimes trade-off energy use and accuracy, that is, reducing the energy consumption of a given program at a cost of lower accuracy. We also note that for some subfields of computer science (for example, robotics and computer architecture), the material is very salient. Similarly, on the engineering side, corporations running large datacenters, for example, are very aware of the energy consumption of these centers and strive to minimize those costs. Moreover, even the more abstract subfields of the discipline, in particular computer science theory, have been influenced by material considerations. For example, several of the founding papers in the famous *Automata Studies* collection³⁴ start from clearly material considerations, for example, von Neumann's paper on the synthesis of reliable organisms from unreliable components, which is motivated by the unreliability of vacuum tubes and biological components.

Figure 1. A typical cloud computing image.



These disciplinary norms and practices of computer science do not per se hide the materiality of IT—as noted earlier, good engineering practice dictates that on some occasions we must peer into the black box—but these intellectual moves do make this materiality easier to minimize or to ignore. In addition, they ready the terrain for other forces, as we will discuss, to push back directly on considering the material side of information and computation.

Metaphors We Compute By

Lakoff and Johnson²³ discuss a central property of metaphors: the systematicity that allows us to comprehend one concept in terms of another necessarily highlights some aspects while hiding others. For IT, computing metaphors that hide the material impacts of IT constitute another force.

One important metaphor here is that of “cloud computing,” which conjures up images of something light and insubstantial, floating up in the sky. This metaphor highlights that the servers and their supporting infrastructure are located someplace else, and that users of the cloud need not concern themselves with how they are maintained, monitored, powered, cooled, and so forth; it tends to hide that they are even material at all.

Visual representations of the cloud often make explicit these metaphorical implications. Figure 1, for example, shows a typical diagram for the cloud. Notice the peripherals are highlighted (for example, laptops, mobile phones, and printers); however, the cloud itself is represented as a blob that hides its materiality in toto (for example, no servers, cables, cooling, energy sources, and so forth).

The origins of the term “cloud computing” are disputed. One early and influential use was by Eric Schmidt, then CEO of Google, in a Search Engine Strategies Conference conversation in 2006:³³

It starts with the premise that the data services and architecture should be on servers. We call it cloud computing—they should be in a “cloud” somewhere. And that if you have the right kind of browser or the right kind of access, it doesn’t matter whether you have a PC or a Mac or a mobile phone or a BlackBerry or what have you—or new devices still to be developed—you can get access to the cloud.

However, there are earlier uses of the term. Business plans from 1996 use the term “cloud computing” in a way that would be familiar today.²⁹ The authors of those plans state that it was born as a marketing term, which suggests there may have been some awareness of the implications of the metaphor. This usage in turn drew on a convention, used by network design engineers, to loosely sketch the other networks that theirs hooked into as a rough cloud-shaped blob.³¹ On the one hand, this convention represents a powerful design technique—abstracting away irrelevant details (as noted in the section “Disciplinary Norms and Practices of Computer Science”)—but at the same time, it implies these details, including the material implications of these other networks, are not relevant to the task at hand.

Another term with metaphorical connotations is “ethernet,” named after the “luminiferous ether” (or “luminiferous aether”),²⁴ a hypothesized medium through which light travels. While the existence of the luminiferous ether was disproven by the famed Michelson-Morley experiment in 1887, the metaphor of an omnipresent, passive medium lives on in networking terminology, perhaps suggesting, as did the luminiferous ether, something almost invisible. The term thus highlights the ubiquity and convenience of the network while hiding its material manifestations as cables, routers, and other hardware.

The “agent” metaphor, as exemplified in early presentations such as the Apple Knowledge Navigator video^{1,39} and the Starfire concept video from Sun Microsystems,³⁵ and now at least partially realized in systems such as Siri from Apple and Alexa from Amazon, centers on an intellectual rather than a material view of computation. Software agents (“bots”) en masse roaming the Internet embed the metaphor of agents within the cloud. The result is a densely populated society of disembodied agents with limited if no material reality. In all these cases, hidden from view is the vast material apparatus underneath.

Utopian Visions

Another force derives from utopian visions of technology. One notable example is Mark Weiser’s vision of ubiq-

uitous computing in which the technology “fades into the background.” Weiser writes:³⁸

The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it. ... We are therefore trying to conceive a new way of thinking about computers, one that takes into account the human world and allows the computers themselves to vanish into the background.

Such a disappearance is a fundamental consequence not of technology but of human psychology. Whenever people learn something sufficiently well, they cease to be aware of it.

Thus, Weiser’s vision is even broader: as this technology becomes truly embedded in human activity we won’t be aware of it at all. As the field of ubiquitous computing has evolved, with computation embedded in walls, clothes, and so forth, the materiality to support it is often physically and intentionally hidden from the user. Indeed, this material disappearance is often considered evidence of good design.

The “agent” metaphor, in particular in its early presentations such as the Knowledge Navigator and Starfire, is also another utopian vision. These virtual agents are typically accessible via peripherals such as screens or phones, doing the bidding of those they serve. In some current systems, Weiser’s vision of ubiquitous computing blends with that of agents with, for example, personal software agents that remotely control household thermostats and lighting. To our point, no hint of the infrastructure to support such computing visions is apparent.

Yet another utopian vision underlies an “information society” in which bits replace atoms—ignoring the material underpinnings of those bits.^{6,12} Nicholas Negroponte in *Being Digital*,²⁵ for example, writes:

World trade has traditionally consisted of exchanging atoms. ... This is changing rapidly. The methodical movement of recorded music as pieces of plastic, like the slow human handling of most information in the form of books, magazines, newspapers, and videocassettes, is about to become the instantaneous and inexpensive transfer of electronic data that move at the speed

of light. ... This change from atoms to bits is irrevocable and unstoppable.

It is worth emphasizing this last sentence. It claims there is an irrevocable and unstoppable change from atoms to bits: the material vanishes.


“Data”—instantly, expansively, anywhere, anytime—is still another utopian vision in which materiality has little presence. Versions of this utopia can be found in Berners-Lee’s vision for the World Wide Web with open access for all⁵ and in the “cornucopian design paradigm”²⁸ found in human-computer interaction.

All of these metaphors and visions—cloud computing, ethernet, agents, ubiquitous computing as technology that fades into the background, an information society in which bits replace atoms, and data accessible anywhere and anytime—have substantial currency in popular and business culture and consequent impacts on the visibility of the materiality of IT.


Where Has All the Hardware Gone? (And energy? And water?)

But if the digital is in actuality grounded in the material, where is all that materiality hiding? Buildings full of servers are generally out of sight and out of mind, even without the highlighting and hiding arising from the cloud computing metaphor. Very likely most people have never been in a server warehouse. Even for people who live near one, in some cases its considerable consumption of water and electricity is concealed in part due to secrecy agreements signed with local governments.¹³ There is also a disconnect between physical infrastructure and personal use—it is also likely that most people do not know where their data is stored or requests are being processed (the privacy-conscious might at most know which nation the servers are in, given the different regulations); and of course one of the typical features of cloud computing is that service providers can seamlessly shift the storage and processing to different locations.

The devices that users do own and see, such as smartphones and tablets, have become smaller and smaller. These devices are largely sealed as well—a typical smartphone, for example, is a thin, sleek case with a glass screen, with no external suggestion of



Growth that requires ever-more material resources cannot continue forever in a finite world.



the tightly packed chips, batteries, sensors, and so on that are inside that case. Taken together, these designs nudge toward minimizing awareness of their overall material footprint.

Economic Forces

There is also a constellation of economic forces that work against recognizing materiality in IT.

In the developed world, there is a powerful and widespread culture of consumption and rapid obsolescence around electronic devices, with pressures to have the latest devices, including as part of one’s self- and public image. This culture of consumption is accompanied by a throw-away mentality that often makes older devices almost worthless—but still needing disposal. For example, consumers by and large are disinterested in heirloom digital devices (for example, an older iPad), both because companies do not provide ongoing support for hardware or software and because, seemingly by intention, designs fall out of fashion.⁷ Moreover, according to Koebler²² many large companies (for example, Apple, LG, John Deere) make their devices difficult or impossible for most people to fix (for example, creating software that limits repair, restricting the availability of replacement parts, and minimizing authorized repair programs).

Smartphones and other personal electronic devices can also be highly addictive. This problem is seeing increased attention, both in general and for particularly problematic situations, such as texting while driving, parents at playgrounds, and students in classrooms or while studying, among many others. However, the focus of this attention is primarily on the impacts for social interaction, self-image and self-esteem, safety, child development, effects on learning, ability to think deeply and in a sustained fashion, and the like. These are all important concerns. But the material and energy impacts of this addictive and pervasive use are often ignored, although that is a significant result as well.

A related force arises from one of the widespread business models for providing software and services, which Zuboff has called *surveillance capitalism*.⁴⁰ Currently, many consumer services, such as search, personal email,

social media, news, and others, are paid for by accumulating vast amounts of personal information about end users and targeted advertising. These advertising and marketing schemes, powered by user data, in turn feed into and reinforce consumerism. Implementing these schemes also results in a powerful set of price and social signals to consume lots of these services: more use implies more advertising revenue and more data about user activities, interests, and preferences, thus motivating companies to encourage consumption. And so the circle goes.

For example, Facebook devotes a huge amount of effort toward hooking in its users to maximize user time on the site and generate massive amounts of user data—the addictive nature of smartphones and other personal electronic devices is not entirely an accident. In promoting gmail, Google advertises “never delete another message—just archive it!” Note that the gmail slogan celebrates the opportunity to save all that email (and implicitly messages “no need to consider the material aspects”). The implication is that people are being archaic if they worry about how much storage email consumes or how much processing is needed when searching a large email archive; and they are probably going to delete something they will eventually want if they don’t just archive it. These business practices, and the many others like them, require more and more data centers and processing power.

Prices can provide strong economic signals about materiality, but the current advertising-supported business models for infrastructure makes those signals largely invisible to the end user. For example, there is no fee per search for Web searches, and users can save vast numbers of email messages without incurring a fee. Of course, the price is still paid in the end, even if it is not readily apparent to the end user—it is built into the cost of the goods and services purchased from the advertisers, and also manifests in the negative externalities of vast collections of personal information and in the environmental impacts of producing and eventually disposing of the underlying hardware.

To connect with the earlier discussion on metaphor, Eric Schmidt in the interview cited there states: “And so

A Claim and Three Questions

- ▶ The materiality of information technology is largely invisible.
- ▶ What’s at stake with this invisibility (and why should we care)?
- ▶ What forces contribute to this invisibility?
- ▶ What (if anything) should the computing and information community do about it?

what’s interesting is that the two—cloud computing and advertising—go hand-in-hand.” Indeed.

More broadly, our overall economic system is currently predicated on unending growth. The IT industry has linked itself strongly to this ethos, with some particular manifestations being the constant need for novelty, the accompanying throw-away culture around consumer electronics, and the glorification of disruption for its own sake. Yet growth that requires evermore material resources cannot continue forever in a finite world.

One response to this observation about unending growth is the idea of decoupling: despite the limitations of the physical world, we can still have unending economic growth because we can separate growth from the use of materials, for example, we could grow a service economy rather than a material economy. Making decoupling work requires absolute decoupling (using fewer materials in total), not just relative decoupling (using materials more efficiently, but potentially still using more materials in total). An in-depth discussion of decoupling is beyond the scope of this essay—but to date there has been no evidence of absolute global decoupling,¹⁹ the relevant sphere given our globalized economy in which material flows occur. See Jackson²¹ for more on this issue and additional references.

What Could Be Done?

Here, we sketch some ideas for what might be done to counter the forces that work against considering the materiality of IT, and when appropriate to increase the visibility of the negative impacts in terms of materials, energy, and waste. While the focus of this article is visibility, there are also a few thoughts here on how increased visibility might translate to mitigations.

Of course, increasing visibility is not always desirable; however, making

some phenomena visible can be useful if doing so surfaces important considerations. The considerations might be important for any number of reasons, including economic, engineering, environmental, or moral ones. Given the very large impacts of information technology in terms of raw materials, pollutants, energy, and waste, we argue this is one such case.

Disciplinary norms and practices. We earlier noted how certain key intellectual moves in computer science involve setting aside the materiality of computation, abstracting away the physical manifestations to concentrate on information and computation; at the same time, a key aspect of using this abstraction well is to understand when it is appropriate to simply use and when one needs to peer into the black box, that is, bring back into view some of the properties that were abstracted away. Currently, the properties brought back into view are typically such factors as bandwidth, power consumption (particularly for mobile devices), processing time, and memory use.

Within fields and subfields, we can scrutinize our work for ways in which it masks or minimizes materiality. For example, in the HCI sustainability community, up until recently the material side of cloud and other digital infrastructure has received significantly less attention than that of the devices that users see and use, despite its having a similar or even larger environmental footprint.²⁸

There are also opportunities to incorporate this materiality perspective into ongoing research activities. For example, a topic of current research is developing algorithms that allow for trade-offs between energy use and accuracy, and further, adding support to high-level programming languages for making these trade-offs^{28,32}—so allowing some of the benefits of reasoning

with higher-level abstractions while at the same time being able to consider the consequences for energy use of trade-offs. As another example, there could be similar efforts to make the energy use by servers visible while still using higher level abstractions (either in debug mode for developers, or for helping instrument end-user applications).

In terms of training the next generation, one targeted educational approach is to focus on interface designers, software engineers, hardware engineers, and others, bringing in consideration of the materiality considerations of the technologies they are developing, along with developing engineering intuition and guidelines as to when considering these materialities is important. This overlaps with the properties already typically brought into view, but is not identical—for example, usually designers of mobile applications focus on how fast the battery in the mobile device is drained, rather than overall power consumption (including power used by servers, the device while plugged in, for manufacture and disposal, and so forth).

Metaphors. Turning now to metaphors, what characteristics would better metaphors have? What would they highlight, and what would they hide? Without being overly prescriptive, better metaphors from the perspective of materiality would appropriately highlight the materiality of IT, while hiding unnecessary details about that materiality. Consider the cloud computing metaphor. Instead of the typical fluffy cloud image that hides completely the material aspects of computing infrastructure (see, for example, Figure 1), alternative images that provide some indication of the material aspects of the infrastructure, such as servers and cables inside the cloud, would be an improvement (compare with, for example, Figure 2). If such images for the cloud became commonplace, they would go some distance toward increasing awareness of the materiality of the current IT ecosystem.

On the surface, the strategy of modifying existing, entrenched metaphors may seem the easiest approach, as it brings some aspects of materiality forward without disrupting shared cultural frames. However, not all metaphors will be amenable to such

“materialization” and, in some cases, alternative metaphors may be better for a host of reasons. Accordingly, another strategy entails developing new metaphors that better reflect materiality. These new metaphors could then co-exist with existing ones, or perhaps even supplant them.

New opportunities for designing and spreading well-chosen metaphors arise alongside new technologies. Sometimes the new metaphors come first—providing utopian visions that help guide technical work—and sometimes they arise alongside or after the technology, for example as part of an effort to provide a popular explanation of a new technology. As the field moves forward with future innovations and metaphors are generated to communicate about those visions and innovations, attention should be paid to how materiality is reflected. We can intentionally develop future metaphors to help highlight materiality in an appropriate manner. Overall, whatever metaphors the field settles on going forward should not so comprehensively hide the materiality of IT.

Envisioned futures. Utopian visions of technology can inspire and help guide our technical work. All too often, these visions neglect to bring forward the materiality of IT in meaningful ways. But what if this were not the case? Imagine a utopian vision that is intentionally grounded in the natural world as a base for any anticipatory future. Within and bounded by the natural world is human society, with a goal of prosperity and supporting human flourishing—but likely with a different understanding of prosperity than at present, one that respects the inherent worth, regenerative cycles, and limits of the natural world. Government in turn is subservient to society, and finally the economy is subservient to all three other systems.⁴ This vision is in sharp contrast to our current system, in which the economy takes priority. With the natural world at the center, material implications would be forefront. Impacts on the material and social worlds would drive and constrain IT development. Such a vision stands in stark contrast to those such as Weiser’s disappeared technical infrastructure or Negroponte’s atom-less bits. Instead, utopian visions like this

and others can help the field imagine and innovate within the bounds of an actual realism that is the materiality of the natural world we live in.

In the meantime, for anticipatory futures that require less far-reaching transformations, we can uncover the hidden materiality of those visions and weave those into these futures. For example, we can amend the Weiser ubiquitous computing vision to foreground the enormous IT ecosystem needed to support such disappearance of technology for the end user—we can tell stories of where the massive amounts of data are stored, processed, and reconfigured and of how much and what sorts of data are collected and by what means. Similarly, we can amend the Negroponte vision to foreground the atoms (for example, servers, cables, satellites, and so on) needed in reality to support visionary interaction models (but no longer of “atom-less bits”).

More generally, for any anticipatory IT future, we can press the futurists to tell us about the materiality of their visions—the physical components, the infrastructure, the energy use, the frequency of replacement, the waste and the disposal. To be meaningful, such discussions will need to encompass the appropriate scope and scale—for example, a vision of every home in the U.S. equipped with sensors and smart controls must also include the material consequences of this vision both in the U.S. and also worldwide, including wherever the hardware is manufactured and eventually disposed of, the energy use, and so on.

Visibility of hardware. The material impacts of designed artifacts, including energy use, can be made more visible.²⁷ For example, some paper goods now advertise how using recycled products reduce consumers’ carbon footprints; likewise, light switches can be labeled to remind users of the energy requirements. In an effort to make IT hardware more visible, technology development proposals could include a “materiality impact statement” (either standalone or as part of a larger social impact statement). Such statements might include a list of the metals, minerals, plastics, and other materials contained in the hardware, a description of their energy

consumption, provenance, and processing, and a plan for reuse, recycling, or disposal. For IT professionals, the targeted educational approaches discussed earlier also help to make the hardware more visible in design and other discussions.

Economic models, laws, and regulations. Prices as economic signals often imply it is appropriate to minimize consideration of the material impacts of IT, since typically these prices do not account for the full life cycle costs of the technology, which includes the environmental and societal costs of material extraction, energy use, and e-waste disposal. Taxes or fees could help change this. A goal could be that prices more accurately account for these full life cycle costs, with revenues going to mitigations of different kinds. This could be complemented by public information, supported by regulation or other mechanisms, for example, by labeling new products with information about their eventual disposal and how to do it well, or explaining how certain full life-cycle taxes or fees are being used, or encouraging reuse and repair.

Services that are paid for by accumulating vast amounts of personal information about the end users and tar-

geted advertising erase price as an economic signal visible to the end user. There are increasing calls to regulate this industry³⁷ or to treat particular corporations as monopolies subject to antitrust action. However, it seems likely that regulation can only go so far in addressing the material impacts of this portion of the IT industry. We should therefore at least consider alternatives for funding these services, such as government or other societal support (such as co-ops or volunteers) as a part of a civic commons infrastructure. Another is corporations that provide the services on a pay-per-use basis, perhaps using a utility rather than a content model.

More accurate accounting for life cycle costs (“let prices tell the truth”) has a strong appeal within the currently prevailing worldview, in which economics plays a foundational role. However, as discussed previously, one can imagine other societies in which the economy is subservient to other systems (the natural world, society, government). In such a society, one goal of such taxes and fees could be to help put bounds on activities that have larger downsides than society as a whole wants to bear. In such a society, these taxes, for example, might be set consid-

erably higher than would be done simply to have them reflect full life-cycle costs and be part of a larger coordinated strategy to enable humanity to live more lightly on the Earth (other components being education, and even attempts to shift culture).

Conclusion and Directions for Future Work

This article has presented some ideas on the forces that push toward minimizing the visibility and consideration of the materiality of digital technologies, in particular their environmental impacts. All of the topics noted here would benefit from exploration in much greater depth.

One direction for future investigation is understanding the mental models that people have of information technology, both of the devices they have personally and also of the server and networking infrastructure that backs them. In the section on metaphor we suggest that the cloud computing metaphor conjures up images of something light and insubstantial, somewhere up in the sky. What mental models do people have of the devices and infrastructure? What is the impact of the cloud computing metaphor on these mental

Figure 2. A revised cloud computing image, with some indication of the functionality and material side of the cloud's contents.



models? Do the models adequately represent the material side?

Do the five overarching categories of forces identified here (disciplinary norms and practices of computer science, metaphor, utopian visions, visibility of hardware, and economics) capture the range of forces, or are there other important ones? Within each category, certainly additional forces could be identified and investigated empirically. We want to highlight the economic forces in particular as central and needing much additional investigation, as well as the possibility of unending economic growth and the decoupling of economic from material growth. This recommendation echoes a key point made by Ekbia and Nardi¹⁴ in connection with human computer interaction research: “Computing and political economy are much more intertwined than current discourse in HCI admits. Our contention is not that HCI researchers and practitioners are unaware of the relationship between economy and technology; rather, that this does not typically figure in any deep way into our theories, practices, and designs. ... Researchers tend to focus on the cultural aspects of technology at the expense of the more material and economic facets.”

This article is primarily written from the perspective of the developed world. How do these issues play out in the developing world? For example, connectivity is often a challenge there, so assumptions of seamless integration with cloud services break down; but on the other hand, the material side of the cloud may be even less visible than in the developed world. We have also touched on the issue of recycling or disposal of e-waste—and it is often in the developing world where this recycling or disposal happens.

Finally, there should be much more work on specific policies and other approaches to what can be done. Again, visibility is not an end in itself: we want to lessen the negative impacts of IT. While increased visibility may help with this, the results from such interventions by themselves will be quite limited in comparison with what actually must be done to live within the Earth’s limits. The dark side of IT’s materiality is due in part to particular characteristics of

the technology, industry, and discipline, but is also a manifestation of integrated and systemic environmental, economic, and political problems, which must be addressed in a similarly integrated fashion. Imagine for a moment a different society in which there is a larger movement to enable humanity to live more lightly on Earth and within its limits, respecting not only our present generation but ones to come.¹⁷ What is a proper role for IT, industry, and research? And how could we get there?

Acknowledgments. An earlier version of this article⁸ was presented at the 2018 LIMITS Conference in Toronto, Canada. Thanks to Deric Gruen, Lance Bennett, Eli Blevis, Richard Landler, Barath Raghavan, Nithya Sambasivan, members of the TroubledWorlds seminar at UW led by Megan Finn and Daniela Rosner, the anonymous *Communications’* reviewers, and to the Tech Policy Lab at the University of Washington for funding. C

References

1. Apple Computer. Knowledge navigator. YouTube video, 1987.
2. Baldé, C.P., Forte, V., Gray, V., Kuehr, R., and Stegmann, P. *The Global E-waste Monitor*. United Nations University (UNU), International Telecommunication Union (ITU) & International Solid Waste Association (ISWA), Bonn/Geneva/Vienna, 2017.
3. Belkhir, L. and Elmelig, A. Assessing ICT global emissions footprint: Trends to 2040 & recommendations. *J. Cleaner Production* 177 (2018), 448–463.
4. Bennett, W.L., Borning, A., and Gruen, D. Solutions for environment, economy, and democracy (SEED): A manifesto for prosperity. *ACM interactions* 25, 1 (Dec. 2017), 74–76.
5. Berners-Lee, T. and Fischetti, M. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor*. HarperCollins, New York, 2000.
6. Blanchette, J. A material history of bits. *J. American Society for Information Science and Technology* 62, 6 (2011), 1042–1057.
7. Blevis, E. Sustainable interaction design: Invention & disposal, renewal and reuse. In *Proceedings of the 2007 SIGCHI Conf. on Human Factors in Computing Systems*. ACM, New York, NY, 503–512.
8. Borning, A., Friedman, B., and Gruen, D. What pushes back from considering materiality in it? In *Proceedings of the 2018 Workshop on Computing Within Limits*. ACM, New York, NY.
9. Borning, A. and Muller, M. Next steps for value sensitive design. In *Proceedings of the 2012 SIGCHI Conf. Human Factors in Computing Systems*. ACM, New York, NY, 1125–1134.
10. Chien, A.A. Owning computing’s environmental impact. *Commun. ACM* 62, 3 (Mar. 2019), 5.
11. Columbus, L. Roundup of Internet of things forecasts and market estimates. *Forbes*, 2016.
12. Dourish, P. *The Stuff of Bits: An Essay on the Materialities of Information*. MIT Press, Cambridge, MA, 2017.
13. Dwoskin, E. Google reaped millions in tax breaks as it secretly expanded its real estate footprint across the U.S. *Washington Post*, Feb 15, 2019.
14. Ekbia, H. and Nardi, B. The political economy of computing: The elephant in the HCI room. *interactions* 22, 6 (Oct. 2015), 46–49.
15. Friedman, B. and Hendry, D. *Value Sensitive Design: Shaping Technology with Moral Imagination*. MIT Press, Cambridge, MA, 2019.
16. Friedman, B., Hendry, D., and Borning, A. A survey of value sensitive design methods. *Foundations and Trends in Human-Computer Interaction* 11, 23 (2017), 63–125.
17. Friedman, B. and Nathan, L.P. Multi-lifespan information system design: A research initiative for the HCI community. In *Proceedings of the 2010 SIGCHI Conf. Human Factors in Computing Systems*. ACM, New York, NY, 2243–2246.
18. Gelenbe, E. and Caseau, Y. The impact of information technology on energy consumption and carbon emissions. *Ubiquity* (June 2015), 1:1–1:15.
19. Giljum, S., Dittich, M., Lieber, M., and Lutter, S. Global patterns of material flows and their socio-economic and environmental implications: A MFA study on all countries world-wide from 1980 to 2009. *Resources* 3, 1 (2014), 319–339.
20. Hazas, M. and Nathan, L. Eds. *Digital Technology and Sustainability: Engaging the Paradox*. Routledge, U.K., 2018.
21. Jackson, T. *Prosperity without Growth: Foundations for the Economy of Tomorrow*. 2nd Edition. Routledge, U.K., 2016.
22. Koebler, J. Internal documents show Apple is capable of implementing right to repair legislation. *Motherboard Tech by Vice*, Mar. 2019.
23. Lakoff, G. and Johnson, M. *Metaphors We Live By*. University of Chicago Press, Chicago, IL, 2003.
24. Metz, C. Ethernet—A networking protocol name for the ages. *The Register*, Mar. 2009.
25. Negroponte, N. *Being Digital*. Knopf, New York, NY, 1995.
26. Park, J., Esmailzadeh, H., Zhang, X., Naik, M., and Harris, W. Flexjava: Language support for safe and modular approximate programming. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*. ACM, New York, NY, 745–757.
27. Pierce, J. and Paulos, E. Materializing energy. In *Proceedings of the 8th ACM Conf. Designing Interactive Systems*, 2010. ACM, New York, NY, 113–122.
28. Preist, C., Schien, D., and Blevis, E. Understanding and mitigating the effects of device and cloud service design decisions on the environmental footprint of digital infrastructure. In *Proceedings of the 2016 CHI Conf. Human Factors in Computing Systems*. ACM, New York, NY, 1324–1337.
29. Regalado, A. Who coined ‘cloud computing’? *MIT Technology Review*, Oct. 2011.
30. Rockström, J. et al. A safe operating space for humanity. *Nature* 461 (Sept. 2009), 472–475.
31. Rosen, R.J. Clouds: The most useful metaphor of all time. *The Atlantic*, Sept. 2011.
32. Sampson, A., Dielt, W., Fortuna, E., Gnanapragasam, D., Ceze, L., and Grossman, D. Enerj: Approximate data types for safe and general low-power computation. In *Proceedings of the 32nd ACM SIGPLAN Conf. Programming Language Design and Implementation*, 2011. ACM, New York, NY, 164–174.
33. Schmidt, E. Conversation with Eric Schmidt hosted by Danny Sullivan. Search Engine Strategies Conference, Aug. 2006.
34. Shannon, C. and McCarthy, J. Automata studies. *Annals of Mathematical Studies*, Princeton University Press, 1956.
35. Tognazzini, B. The “Starfire” video prototype project: A case history. In *Proceedings of the SIGCHI Conf. Human Factors in Computing Systems*. ACM, New York, 1994.
36. United States Government Accountability Office. Internet of things: Status and implications of an increasingly connected world. Technical Report GAO-17-75, Government Accountability Office, May 2017.
37. Vardi, M.Y. Are we having an ethical crisis in computing? *Commun. ACM* 62, 1 (Jan. 2019), 7.
38. Weiser, M. The computer for the 21st century. *Scientific American* 265, 3 (Sept. 1991), 94–105.
39. Wikipedia contributors. Knowledge navigator, 2018.
40. Zuboff, S. *The Age of Surveillance Capitalism: The Fight for a Human Future at the New Frontier of Power*. Public Affairs Books, New York, NY, 2019.

Alan Borning is Professor Emeritus in the Paul G. Allen School of Computer Science and Engineering at the University of Washington, Seattle, WA, USA.

Batya Friedman is a professor in the Information School at the University of Washington, Seattle, WA, USA, where she directs the Value Sensitive Design Research Lab.

Nick Logler is a Ph.D. candidate in the Information School at University of Washington, Seattle, WA, USA.

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

The Essentials of Modern Software Engineering

Free the Practices from the Method Prisons!

This text/reference is an in-depth introduction to the systematic, universal software engineering kernel known as “Essence.” This kernel was envisioned and originally created by Ivar Jacobson and his colleagues, developed by Software Engineering Method and Theory (SEMAT) and approved by The Object Management Group (OMG) as a standard in 2014. Essence is a practice-independent framework for thinking and reasoning about the practices we have and the practices we need. **It establishes a shared and standard understanding of what is at the heart of software development. Essence is agnostic to any particular methods, lifecycle independent, programming language independent, concise, scalable, extensible, and formally specified.** Essence frees the practices from their method prisons.

HIGH PRAISE FOR THE ESSENTIALS OF MODERN SOFTWARE ENGINEERING

“Essence is an important breakthrough in understanding the meaning of software engineering. It is a key contribution to the development of our discipline and I’m confident that this book will demonstrate the value of Essence to a wider audience. It too is an idea whose time has come.” – Ian Somerville, St. Andrews University, Scotland (author of *Software Engineering, 10th Edition*, Pearson)

“What you hold in your hands (or on your computer or tablet if you are so inclined) represents the deep thinking and broad experience of the authors, information you’ll find approachable, understandable, and, most importantly, actionable.”
– Grady Booch, IBM Fellow, ACM Fellow, IEEE Fellow, BCS Ada Lovelace Award, and IEEE Computer Pioneer

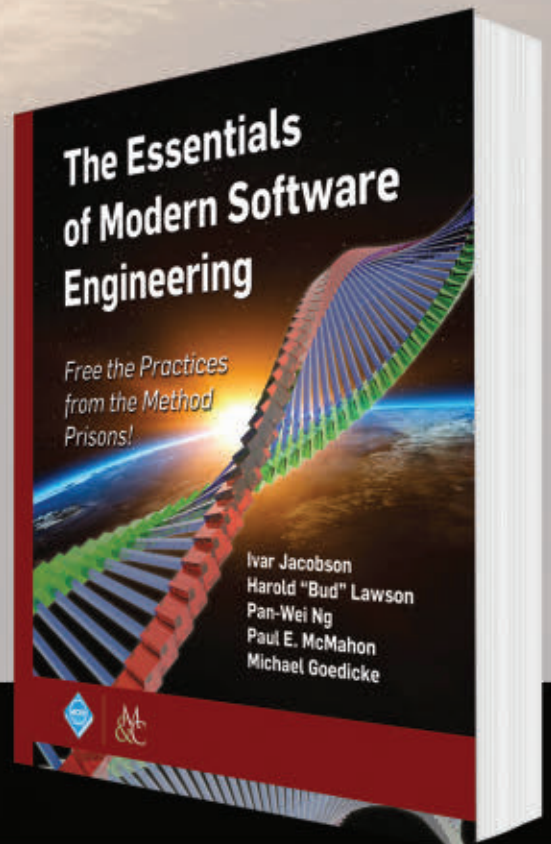
**Ivar Jacobson, Harold “Bud” Lawson,
Pan-Wei Ng, Paul E. McMahon,
Michael Goedicke**

ISBN: 978-1-947487-24-6

DOI: 10.1145/3277669

<http://books.acm.org>

<http://store.morganclaypool.com/acm>



ACM BOOKS
Collection I

Advances in how programs treat natural language words have a big impact in AI.

BY NOAH A. SMITH

Contextual Word Representations: Putting Words into Computers

THIS ARTICLE AIMS to tell the story of how we put words into computers. It is part of the story of the field of natural language processing (NLP), a branch of artificial intelligence.^a It targets a wide audience with a basic understanding of computer programming, but avoids a detailed mathematical treatment, and it does not present any algorithms. It also does not focus on any particular NLP application, such as translation, question answering, or information extraction. The ideas presented here were developed by many researchers over many decades, so the citations are not exhaustive but rather direct the reader to a handful of papers that are, in the author's view, seminal. After reading this article, you should have a general understanding of word vectors (also known as word embeddings): why they exist, what problems they solve, where they come from, how they have changed over time, and what open questions exist about them.

^a Recommended NLP textbooks: Jurafsky and Martin¹⁹ and Eisenstein.¹⁰

There are two ways to talk about words:

- ▶ A *word token* is a word observed in a piece of text. In some languages, identifying the boundaries of the word tokens is a complicated procedure (and speakers of the language may not agree on the “correct” rules for splitting text into words), but in English we tend to use whitespace and punctuation to delimit words, and in this article we assume this problem, known as tokenization, is “solved.” For example, the first sentence of this paragraph is typically tokenized as follows (with the end-of-sentence punctuation treated as a separate token):

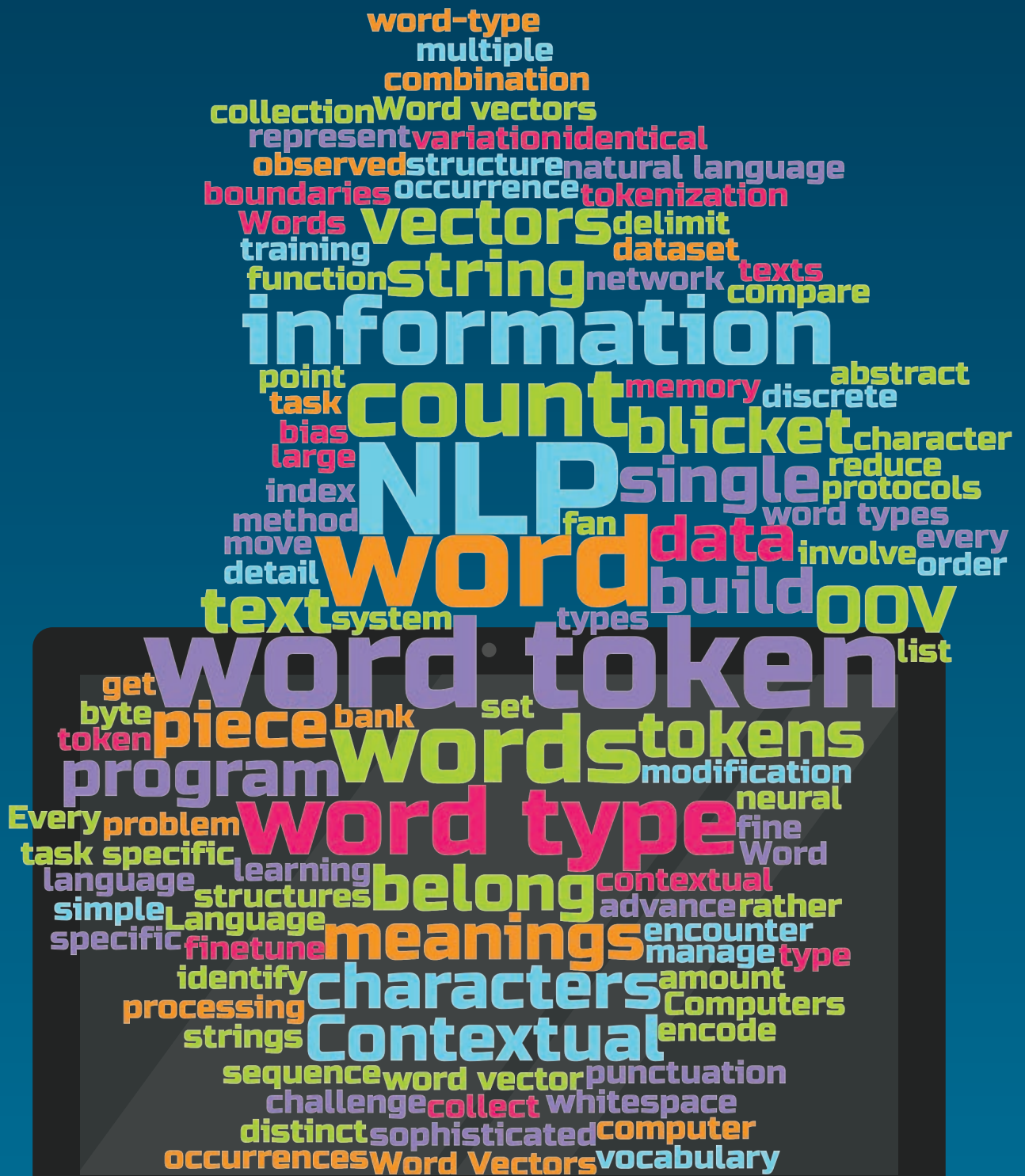
A word token is a word observed in a piece of text.

There are 13 tokens in the sentence.

- ▶ A *word type* is a distinct word, in the abstract, rather than a specific instance. Every word token is said to “belong” to its type. In the example, there are only 11 word types, since the two instances of word share the same type, as do the two instances of *a*. (If we ignored the distinction between upper- and lower-case letters, then there would only be 10 types, since the first word *A* would have the same type as the fifth and ninth words.) When we count the occurrences of a vocabulary word in a collection of texts (known as a corpus, plural corpora), we are counting the tokens that belong to the same word type.

» key insights

- **Even at the most fundamental level of words, representing the meaning of natural language text computationally is a difficult challenge.**
- **Different words' meanings can be more or less similar. Continuous vectors have been used to effectively capture this property, and large collections of text have made it possible to automate discovery of many aspects of word meaning similarity. Conventionally, every word in the vocabulary got a single, fixed vector.**
- **A word's meaning can also vary greatly based on the contexts it appears in. The latest advances recognize and learn about this variation using classical tools from NLP and ML. These methods have shown large improvements across many benchmarks.**



Discrete Words

In a computer, the simplest representation of a piece of text is a sequence of characters (depending on the encoding, a character might be a single byte or several). A word type can be represented as a string (ordered list of characters) but comparing whether two strings are identical is costly.

Not long ago, words were usually *integerized*, so that each word type was given a unique (and more or less arbitrary) nonnegative integer value. This had the advantages that every word type was stored in the same amount of memory, and array-based data structures could be used to index other information by word types (like the string for the word, or a count of its tokens, or a richer data structure containing detailed information about the word's potential meanings). The vocabulary could be continuously expanded as new word types were encountered (up to the range of the integer data type, over four billion for 4-byte unsigned integers). And, of course, testing whether two integers are identical is very fast.

The integers themselves did not mean anything; the assignment might be arbitrary, alphabetical, or in the order word tokens were observed in a reference text corpus from which the vocabulary was derived (that is, the type of the first word token observed would get 0, the type of the second word token would get 1 if it was different from the first, and so on). Two word types with related meanings might be assigned distant integers, and two “adjacent” word types in the assignment might have nothing to do with each other. The use of integers is only a convenience following from the data types available in the fashionable programming languages of the day; for example, in Lisp “gensym” would have served the same purpose, although perhaps less efficiently. For this reason, we refer to integer-based representations of word types as *discrete* representations.

Words as Vectors

To see why NLP practitioners no longer treat word types as discrete, it's useful to consider how words get used in NLP programs. Here are some examples:

► Observing a word token in a given document, use it as evidence to help predict a category for the document.

The idea that words can be more or less similar is critical when we consider that NLP programs, by and large, are built using supervised machine learning.

For example, the word *delightful* appearing in a review of a movie is a cue the reviewer might have enjoyed the film and given it a positive rating.^b

► Observing a word token in a given sentence, use it as evidence to predict a word token in the translation of the sentence. For example, the appearance of the word *cucumber* in an English sentence is a cue that the word *concombre* might appear in the French translation.

► Conversely, given the full weight of evidence, choose a word type to write as an output token, in a given context.

In each of these cases, there is a severe shortcoming to discrete word types: information about how to use a particular word as evidence, or whether to generate a word as an output token, cannot be easily shared across words with similar properties. As a simple example, consider filling in the blank in the following sentence:

S. will eat anything, but V. hates ___

Given your knowledge of the world, you are likely inclined to fill in the blank with high confidence as a token of a type like *peas*, *sprouts*, *chicken*, or some other mass or plural noun that denotes food. These word types share something (together with the other words for food), and we would like for a model that uses words to be able to use that information.^c To put it another way, our earlier interest in testing whether two words are identical was perhaps too strict. Two non-identical words may be more or less similar.

The idea that words can be more or less similar is critical when we consider that NLP programs, by and large, are built using supervised machine learning, that is, a combination of examples demonstrating the inputs and

^b Context matters. “The most delightful part of seeing this movie was the popcorn” would signal just the opposite. See Pang and Lee²⁷ for a detailed treatment of the problems of sentiment and opinion analysis.

^c One situation where this lack of sharing is sorely noticed is the case of new words, sometimes called “out of vocabulary” (OOV) words. When an NLP program encounters an OOV word token, say *blicket*, what should it do? By moving away from discrete words, we have managed to reduce the occurrence of truly OOV word types by collecting information about an increasingly large set of words in advance of building the NLP program.

outputs to a task (at least one of which consists of words) and a mechanism for generalizing from those input-output pairings. Such a mechanism should ideally exploit similarity: anything it discovers about one word should transfer to similar words.

Where might this information about similarity come from? There are two strands of thought about how to bring such information into programs. We might trace them back to the rationalist and empiricist traditions in philosophy, though I would argue it's unwise to think of them in opposition to each other.

One strand suggests that humans, especially those trained in the science of human language, know this information, and we might design data structures that encode it explicitly, allowing our programs to access it as needed. An example of such an effort is WordNet,¹³ a lexical database that stores words and relationships among them such as synonymy (when two words can mean the same thing) and hyponymy (when one word's meaning is a more specific case of another's). WordNet also explicitly captures the different senses of words that take multiple meanings, such as *fan* (a machine for blowing air, or someone who is supportive of a sports team or celebrity). Linguistic theories of sentence structure (syntax) offer another way to think about word similarity in the form of categories like “noun” and “verb.”

The other strand suggests the information resides in artifacts such as text corpora, and we can use a separate set of programs to collect and suitably organize the information for use in NLP. With the rise of ever-larger text collections on the Web, this strand came to dominate, and the programs used to draw information from corpora have progressed through several stages, from count-based statistics, to modeling using more advanced statistical methods, to increasingly powerful tools from machine learning.

From either of these strands (or, more commonly in practice, by intertwining them), we can derive a notion of a word type as a vector instead of an integer.^d In doing so, we can choose the

dimensionality of the vector and allocate different dimensions for different purposes. For example:

► Each word type may be given its own dimension and assigned 1 in that dimension (while all other words get 0 in that dimension). Using dimensions only in this way, and no other, is essentially equivalent to integerizing the words; it is known as a “one hot” representation, because each word type's vector has a single 1 (“hot”) and is otherwise 0.

► For a collection of word types that belong to a known class (for example, days of the week), we can use a dimension that is given binary values. Word types that are members of the class get assigned 1 in this dimension, and other words get 0.

► For word types that are variants of the same underlying root, we can similarly use a dimension to place them in a class. For example, in this dimension, *know*, *known*, *knew*, and *knows* would all get assigned 1, and words that are not forms of *know* get 0.

► More loosely, we can use surface attributes to “tie together” word types that look similar; examples include capitalization patterns, lengths, and the presence of a digit.

► If word types' meanings can be mapped to magnitudes, we might allocate dimensions to try to capture these. For example, in a dimension we choose to associate with “typical weight” *elephant* might get 12,000 while *cat* might get 9. Of course, it's not entirely clear what value to give *purple* or *throw* in this dimension.

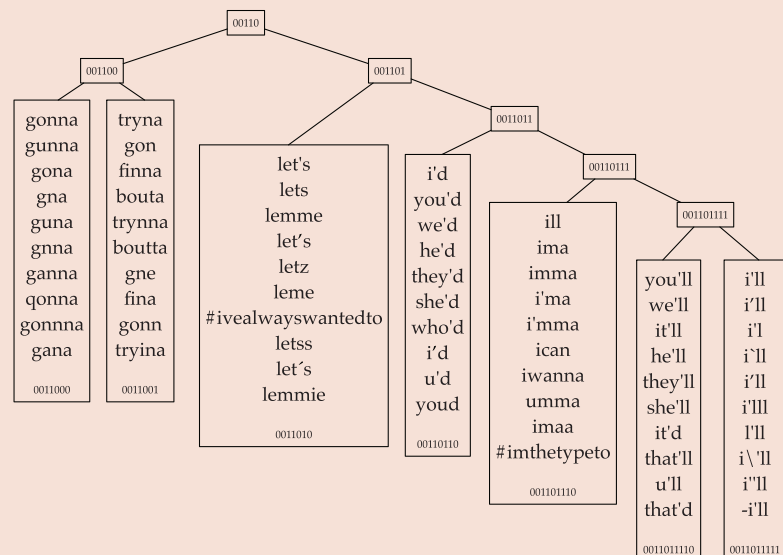
Examples abound in NLP of the allocation of dimensions to vectors representing word types (either syntactic, like “verb,” or semantic, like “animate”), or to multiword sequences (for example, *White House* and *hot dog*). The technical term used for these dimensions is *features*. Features can be designed by experts, or they can be derived using automated algorithms. Note that some features can be calculated even on out-of-vocabulary word types. For example, noting the capitalization pattern of characters in an out-of-vocabulary word might help a system guess whether it should be treated like a person's name.

Words as Distributional Vectors: Context as Meaning

An important idea in linguistics is that words (or expressions) that can be

Figure 1. Example Brown clusters.

These were derived from 56M tweets, see Owoputi et al.²⁶ for details. Shown are the 10 most frequent words in clusters in the section of the hierarchy with prefix bit string 001110. Intermediate nodes in the tree correspond to clusters that contain all words in their descendants. Note that differently spelled variants of words tend to cluster together, as do words that express similar meanings, including hashtags. The full set of clusters can be explored at http://www.cs.cmu.edu/~ark/TweetNLP/cluster_viewer.html. Note that there are several Unicode characters that are visually similar to the apostrophe, resulting in different strings with similar usage.



^d A vector is a list, usually a list of numbers, with a known length, which we call its dimensionality. It is often interpreted and visualized as a direction in a Euclidean space.

used in similar ways are likely to have related meanings¹⁴ (consider our day of the week example). In a large corpus, we can collect information about the ways a word type w is used, for

example, by counting the number of times it appears near every other word type. When we begin looking at the full distribution of contexts (nearby words or sequences of words) in a corpus

where w is found, we are taking a distributional view of word meaning.

One highly successful approach to automatically deriving features based on this idea is *clustering*; for example, the Brown et al.⁴ clustering algorithm automatically organized words into clusters based on the contexts they appear in, in a corpus. Words that tended to occur in the same neighboring contexts (other words) were grouped together into a cluster. These clusters could then be merged into larger clusters. The resulting hierarchy, while by no means identical to the expert-crafted data structure in WordNet, was surprisingly interpretable and useful (an example is shown in Figure 1). It also had the advantage that it could be rebuilt using any given corpus, and every word observed would be included. Hence, suitable word clusters could be built separately for news text, or biomedical articles, or tweets.

Another line of approaches started by creating word vectors in which each dimension corresponded to the frequency the word type occurred in some context.⁸ For instance, one dimension might correspond to *the* and contain the number of times the word occurred within a small window of a token *the*. Contextual patterns on the left or the right, and of varying distances and lengths, might be included. The result is a vector perhaps many times longer than the size of the vocabulary, in which each dimension contains a tiny bit of information that may or may not be useful. An example is shown in Figure 2. Using methods from linear algebra, aptly named *dimensionality reduction*, these vectors could be compressed into shorter vectors in which redundancies across dimensions were collapsed.

These reduced-dimensionality vectors had several advantages. First, the dimensionality could be chosen by the NLP programmer to suit the needs of the program. More compact vectors might be more efficient to compute with and might also benefit from the lossiness of the compression, since corpus-specific “noise” might fall away. However, there is a trade-off; longer, less heavily compressed vectors retain more of the original information in the distributional vectors. While the individual dimensions of

Figure 2. Example calculation of word vectors.

We consider three-word types occurring a science news story ([https:// bit.ly/2B9uaKr](https://bit.ly/2B9uaKr)): astronomers, bodies, and objects. The table above shows the frequency of each word occurring within two positions on either side of the word whose vector we are constructing, giving three (vertical) word vectors with 34 visible dimensions (zeroes and other dimensions not shown). Do you expect bodies to be more similar to astronomers or to objects? The calculation beneath is the cosine similarity score, applied to each word paired with the others (and itself, always giving similarity of one). In this tiny corpus, bodies are closer to objects than either is to astronomers.

Contextual Word Representations:

| context words | v(astronomers) | v(bodies) | v(objects) |
|---------------|----------------|-----------|------------|
| 't | | | 1 |
| , | | 2 | 1 |
| . | 1 | | 1 |
| 1 | | | 1 |
| And | | | 1 |
| Belt | | | 1 |
| But | 1 | | |
| Given | | | 1 |
| Kuiper | | | 1 |
| So | 1 | | |
| and | | 1 | |
| are | | 2 | 1 |
| between | | | 1 |
| beyond | | 1 | |
| can | | | 1 |
| contains | | 1 | |
| from | 1 | | |
| hypothetical | | | 1 |
| ice | | 1 | |
| including | | 1 | |
| is | 1 | | |
| larger | | 1 | |
| now | 1 | | |
| of | 1 | | |
| only | | | 1 |
| out | | 1 | |
| potential | | 1 | |
| the | 1 | | 1 |
| these | | 2 | 1 |
| they | 1 | | |
| think | 2 | | |
| those | | | 1 |
| thought | | | 2 |
| what | 1 | | |

$$\text{cosine_similarity}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \cdot \|\mathbf{v}\|}$$

| | astronomers | bodies | objects |
|-------------|--|--|---|
| astronomers | $\frac{14}{\sqrt{14} \cdot \sqrt{14}} = 1$ | $\frac{0}{\sqrt{24} \cdot \sqrt{14}} = 0$ | $\frac{1+1}{\sqrt{14} \cdot \sqrt{16}} \approx 0.134$ |
| bodies | | $\frac{24}{\sqrt{24} \cdot \sqrt{24}} = 1$ | $\frac{2+2+2}{\sqrt{24} \cdot \sqrt{16}} \approx 0.306$ |
| objects | | | $\frac{16}{\sqrt{16} \cdot \sqrt{16}} = 1$ |

the compressed vectors are not easily interpreted, we can use well-known algorithms to find a word’s nearest neighbors in the vector space, and these were often found to be semantically related words, as one might hope.

Indeed, these observations gave rise to the idea of *vector space semantics*,³³ in which arithmetic operations were applied to word vectors to probe what kind of “meanings” had been learned. Famously, analogies like “*man is to woman as king is to queen*” led to testing whether $v(\textit{man}) - v(\textit{woman}) = v(\textit{king}) - v(\textit{queen})$. Efforts to design word vector algorithms to adhere to such properties followed.

The notable disadvantage of reduced-dimensionality vectors is that the individual dimensions are no longer interpretable features that can be mapped back to intuitive building blocks contributing to the word’s meaning. The word’s meaning is distributed across the whole vector; for this reason, these vectors are sometimes called *distributed representations*.^e

As corpora grew, scalability became a challenge, because the number of observable contexts grew as well. Underlying all word vector algorithms is the notion that the value placed in each dimension of each word type’s vector is a parameter that will be optimized, alongside all the other parameters, to best fit the observed patterns of the words in the data. Since we view these parameters as continuous values, and the notion of “fitting the data” can be operationalized as a smooth, continuous objective function, selecting the parameter values is done using iterative algorithms based on gradient descent. Using tools that had become popular in machine learning, faster methods based on stochastic optimization were developed. One widely known collection of algorithms is available as the *word2vec* package.²⁴ A common pattern arose in which industry researchers with large corpora and powerful computing infrastructure would construct word vectors using an established (often expensive) iterative method, and then

publish the vectors for anyone to use.

There followed a great deal of exploration of methods for obtaining distributional word vectors. Some interesting ideas worth noting include:

- ▶ When we wish to apply neural networks to problems in NLP (see Figure 3), it’s useful to first map each input word token to its vector, and then “feed” the word vectors into the neural network model, which performs a task like translation. The vectors can be fixed in advance (or pretrained from a corpus, using methods like those above, often executed by someone else), or they can be treated as parameters of the neural network model, and adapted to the task specifically.⁶ Finetuning refers to initializing the word vectors by pretraining, then adapting them through task-specific learning algorithms. The word vectors can also be initialized to random values, then estimated solely through task learning, which we might call “learning from scratch”^f

- ▶ We can use expert-built data structures like WordNet as additional in-

put to creating word vectors. One approach, *retrofitting*, starts with word vectors extracted from a corpus, then seeks to automatically adjust them so that word types that are related in WordNet are closer to each other in vector space.¹¹

- ▶ We can use bilingual dictionaries to “align” the vectors for words in two languages into a single vector space, so that, for example, the vectors for the English word type *cucumber* and the French word type *concombre* have a small Euclidean distance.¹² By constructing a function that reorients all the English vectors into the French space (or vice versa), researchers hoped to align *all* the English and French words, not just the ones in the bilingual dictionary.

- ▶ A word’s vectors are calculated in part (or in whole) from its character sequence.²¹ These methods tend to make use of neural networks to map arbitrary-length sequences into a fixed-length vector. This has two interesting effects: in languages with intricate word formation systems (morphology),^g variants

f The result of vectors learned from scratch for an NLP task is a collection of distributed representations that were derived from something other than distributional contexts (the task data).

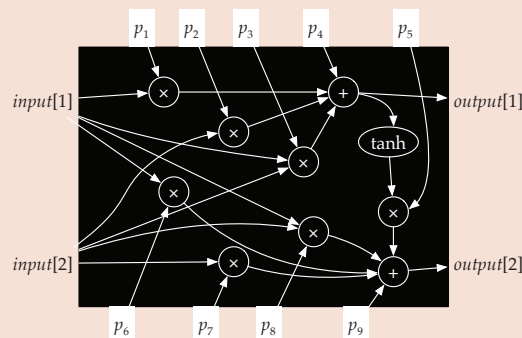
g For example, the present tense form of the French verb *manger* is *mange*, *manges*, *mangeons*, *mangez*, or *mangent*, depending on whether the subject is singular or plural, and first, second, or third person.

Figure 3. A simple neural network.

A neural network is a function from vectors to vectors. A very simple example is a function from two-dimensional inputs to two-dimensional outputs, such as:

$$\begin{aligned} \textit{output}[1] &= p_1 \times \textit{input}[1] + p_2 \times \textit{input}[2] + p_3 \times \textit{input}[1] \times \textit{input}[2] + p_4 \\ \textit{output}[2] &= p_5 \times \tanh(\textit{output}[1]) + p_6 \times \textit{input}[1] + p_7 \times \textit{input}[2] \\ &\quad + p_8 \times \textit{input}[1] \times \textit{input}[2] + p_9 \end{aligned}$$

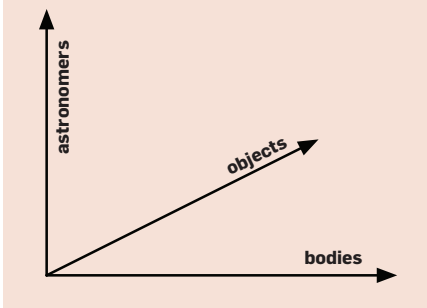
Neural networks are almost always defined in terms of parameters, here denoted by p_1, \dots, p_9 , which are automatically chosen using standard machine learning algorithms. Typically, they include at least one transformation that is not linear (for example, the hyperbolic tangent).



The illustration displays the toy neural network as a computation graph (inside the black box), with inputs on the left, outputs on the right, and each parameter corresponds to a gray box placed along the top and bottom. Round nodes inside the box correspond to intermediate operations (addition, multiplication, and tanh).

e Though distributional information is typically used to build distributed vector representations for word types, the two terms are not to be confused and have orthogonal meanings!

Figure 4. Approximate visualization of the relationships between the three-word vectors calculated in Figure 2.



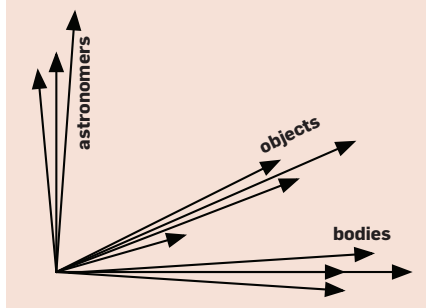
of the same underlying root may have similar vectors; and, differently spelled variants of the same word will have similar vectors. This kind of approach was quite successful for social media texts, where there is rich spelling variation. For example, these variants of the word *would*, all attested in social media messages, would have similar character-based word vectors because they are spelled similarly: *would*, *wud*, *wld*, *wuld*, *wouldd*, *woud*, *wudd*, *whould*, *would*, and *wOuld*.

Contextual Word Vectors

We started this discussion by differentiating between word tokens and word types. All along, we have assumed each word type was going to be represented using a fixed data object (first an integer, then a vector) in our NLP program. This is convenient, but it makes some assumptions about language that do not fit with reality. Most importantly, words have different meanings in different contexts. At a coarse-grained level, this was captured by experts in crafting WordNet, in which, for example, *get* is mapped to over 30 different meanings (or senses). It is difficult to obtain widespread agreement on how many senses should be allocated to different words, or on the boundaries between one sense and another; word senses may be *fluid*.^h Indeed, in many NLP programs based on neural networks, the very first thing that happens is that each word token's type vector is passed into a function that transforms it based on the words in

^h For example, the word *bank* can refer to the side of a river or to a financial institution. When used to refer to a blood bank, we can debate whether the second sense is evoked or a third, distinct one.

Figure 5. Hypothetical visualization of contextual vectors for tokens of astronomers, bodies, and objects from figures 2 and 4.



its nearby context, giving a new version of the word vector, now specific to the token in its particular context. In our example sentence earlier, the two instances of *a* will therefore have different vectors, because one occurs between *is* and *word* and the other occurs between *in* and *piece*; for example, compare figures 4 and 5.

With hindsight, we can now see that by representing word types independent of context, we were solving a problem that was more difficult than it needed to be. Because words mean different things in different contexts, we were requiring that type representations capture *all* of the possibilities (for example, the 30 meanings of *get*). Moving to word token vectors simplifies things, asking the word token representation to capture only what a word means *in this context*. For the same reasons the collection of contexts a word type is found in provide clues about its meaning(s), a particular token's context provides clues about its specific meaning. For instance, you may not know what the word *blicket* means, but if I tell you that I ate a strawberry blicket for dessert, you likely have a good guess.ⁱ

Returning to the fundamental notion of similarity, we would expect words that are similar to each other to be good substitutes for each other. For example, what are some good substitutes for the word *gin*? This question is difficult to answer about the word type (WordNet tells us that *gin* can refer to a liquor for drinking, a trap for hunting, a machine for separating seeds from cotton fibers, or a card game), but easy in a given context (for example, “I use two parts *gin* to one part *vermouth*.”). Indeed, *vodka* might

ⁱ Though such examples abound in linguistics, this one is due to Chris Dyer.

even be expected to have a similar contextual word vector if substituted for *gin*.^j

ELMo, which stands for “embeddings from language models,”²⁸ brought a powerful advance in the form of word *token* vectors—that is, vectors for words in context, or contextual word vectors—that are pretrained on large corpora. There are two important insights behind ELMo:

- If every word token is going to have its own vector, then the vector should depend on an arbitrarily long context of nearby words. To obtain a “context vector,” we start with word type vectors, and pass them through a neural network that can transform arbitrary-length sequences of left- and/or right-context word vectors into a single fixed-length vector. Unlike word type vectors, which are essentially lookup tables, contextual word vectors are built from both type-level vectors and neural network parameters that “contextualize” each word. ELMo trains one neural network for left contexts (going back to the beginning of the sentence a token appears in) and another neural network for right contexts (up to the end of the sentence). Longer contexts, beyond sentence boundaries, are in principle possible as well.

- Recall that estimating word vectors required “fitting the data” (here, a corpus) by solving an optimization problem. A longstanding data-fitting problem in NLP is language modeling, which refers to predicting the next word given a sequence of “history” words (briefly alluded to in our filling-in-the-blank example). Many of the word (type) vector algorithms already in use were based on a notion fixed-size contexts, collected across all instances of the word type in a corpus. ELMo went farther, using arbitrary-length histories and directly incorporating the language models known at the time to be most effective (based on recurrent neural networks³²). Although recurrent networks were already widely used in NLP (see Goldberg¹⁵ for a thorough introduction), training them as language models, then using the context vectors they provide for each word token as pretrained word (token) vectors was novel.

It's interesting to see how the ideas around getting words into computers


^j The author does not endorse this substitution in actual cocktails.

have come full circle. The powerful idea that text data can shed light on a word's meaning, by observing the contexts in which a word appears, has led us to try to capture a word token's meaning primarily through the specific context it appears in. This means that every instance of *plant* will have a different word vector; those with a context that look like a context for references to vegetation are expected to be close to each other, while those that are likely contexts for references to manufacturing centers will cluster elsewhere in vector space. Returning to the example in Figure 2, while instances of *bodies* in this article will likely remain closer to *objects* than to *astronomers*, in a medical news story, we might expect instances of *bodies* to be closer to humans (and by extension *astronomers*).


Why is this advance so exciting? Whether the development of contextual word vectors completely solves the challenge of ambiguous words remains to be seen. New ideas in NLP are often tested on benchmark tasks with objectively measurable performance scores. ELMo was shown to be extremely beneficial in NLP programs that:

- ▶ answer questions about content in a given paragraph (9% relative error reduction on the SQuAD benchmark),
- ▶ label the semantic arguments of verbs (16% relative error reduction on the Ontonotes semantic role labeling benchmark),
- ▶ label expressions in text that refer to people, organizations, and other named entities (4% relative error reduction on the CoNLL 2003 benchmark), and
- ▶ resolve which referring expressions refer to the same entities (10% relative error reduction on the Ontonotes coreference resolution benchmark).

Gains on additional tasks were reported by Peters et al.²⁸ and later by other researchers. Howard and Ruder¹⁸ introduced a similar approach, ULMFiT, showing a benefit for text classification methods. A successor approach, bidirectional encoder representations from transformers (BERT⁹) that introduced several innovations to the learning method and learned from more data, achieved a further 45% error reduction (relative to ELMo) on the first task and 7% on the second. On the SWAG benchmark,



An important idea in linguistics is that words (or expressions) that can be used in similar ways are likely to have related meanings.



recently introduced to test grounded commonsense reasoning,³⁵ Devlin et al.⁹ found that ELMo gave 5% relative error reduction compared to non-contextual word vectors, and BERT gave another 66% relative to ELMo. A stream of papers since this article was conceived have continued to find benefits from creative variations on these ideas, resulting in widely adopted models like GPT-2,³⁰ RoBERTa,²³ T5,³¹ XLM,²⁰ and XLNet.³⁴ It is rare to see a single conceptual advance that consistently offers large benefits across so many different NLP tasks.

At this writing, there are many open questions about the relative performance of the different methods. A full explanation of the differences in the learning algorithms, particularly the neural network architectures, is out of scope for this introduction, but it's fair to say that the space of possible learners for contextual word vectors has not yet been fully explored; see Peters et al.²⁹ for some exploration. Some of the findings on BERT suggest that the role of finetuning may be critical; indeed, earlier work that used pre-trained language models to improve text classification *assumed* finetuning was necessary.⁷ While ELMo is derived from language modeling, the modeling problem solved by BERT (that is, the objective function minimized during estimation) is rather different.^k The effects of the dataset used to learn the language model have not been fully assessed, except for the unsurprising pattern that larger datasets tend to offer more benefit.

Cautionary Notes

Word vectors are biased. Like any engineered artifact, a computer program is likely to reflect the perspective of its builders. Computer programs that are built from data will reflect what is in the data—in this case, a text corpus. If the text corpus signals associations between concepts that reflect cultural biases, these associations should be expected to persist in the word vectors and any system that uses them. Hence, it is not surprising that NLP programs that use corpus-derived word vectors associate,

^k BERT pretraining focuses on two tasks: prediction of words given contexts on both sides (rather than one or the other) and predicting the words in a sentence given its preceding sentence.

for example, *doctor* with male pronouns and *nurse* with female ones. Methods for detecting, avoiding, and correcting unwanted associations are an active area of research.^{3,5} The advent of contextual word vectors offers some possibility of new ways to avoid unwanted generalization from distributional patterns.

Language is a lot more than words. Effective understanding and production of language is about more than knowing word meanings; it requires knowing how words are put together to form more complicated concepts, propositions, and more. This is not nearly the whole story of NLP; there is much more to be said about approaches to dealing with natural language syntax, semantics, and pragmatics, and how we operationalize tasks of understanding and production that humans perform into tasks for which we can attempt to design algorithms. One of the surprising observations about contextual word vectors is that, when trained on very large corpora, they make it easier to disambiguate sentences through various kinds of syntactic and semantic parsing; it is an open and exciting question how much of the work of understanding can be done at the level of words in context.

NLP is not a single problem. While the gains above are quite impressive, it's important to remember that they reflect only a handful of benchmarks that have emerged in the research community. These benchmarks are, to varying degrees, controversial, and are always subject to debate. No one who has spent any serious amount of time studying NLP believes they are "complete" in any interesting sense. NLP can only make progress if we have ways of objectively measuring progress, but we also need continued progress on the design of the benchmarks and the scores we use for comparisons. This aspect of NLP research is broadly known as evaluation and includes both human-judgment-based and automatic methods. Anyone who is an enthusiast of NLP (or AI, more generally) should take the time to learn how progress is measured and understand the shortcomings of evaluations currently in use.

What's Next

Over the next few years, I expect to see new findings that apply variations on contextual word vectors to new problems and that explore modifications to the learning methods. For example, build-

ing a system might involve sophisticated protocols in which finetuning and task-specific training are carried out on a series of dataset/task combinations. Personally, I'm particularly excited about the potential for these approaches to improve NLP performance in settings where relatively little supervision is available. Perhaps, for example, ELMo-like methods can improve NLP for low-resource genres and languages.²⁵ Likewise, methods that are computationally less expensive have the potential for broad use (for example, Gururangan et al.¹⁷). I also expect there will be many attempts to characterize the generalizations that these methods are learning (and those that they are not learning) in linguistic terms; see for example Goldberg¹⁶ and Liu et al.²²

Further Reading

An introductory guide to linguistics for those interested in NLP is provided by Bender¹ and Bender and Lascarides.² A more thorough mathematical treatment of the topics noted here is given in chapter 14 of Eisenstein.¹⁰ For contextual word vectors, the original papers are recommended at this writing.^{9,28}

Acknowledgments. Thanks to Oren Etzioni, Chris Dyer, and students from the University of Washington's Winter 2019 CSE 447 class. NSF grant IIS-1562364 supported research related to this article. C

References

- Bender, E.M. *Linguistic Fundamentals for Natural Language Processing: 100 Essentials from Morphology and Syntax*. Morgan & Claypool, 2013.
- Bender, E.M. and Lascarides, A. *Linguistic Fundamentals for Natural Language Processing II: 100 Essentials from Semantics and Pragmatics*. Morgan & Claypool, 2019.
- Bolukbasi, T., Chang, K., Zou, J.Y., Saligrama, V., and Kalai, A.T. Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In *Proceedings of Advances in Neural Information Processing Systems*, 2016, 4349–4357.
- Brown, P.F., Desouza, P.V., Mercer, R.L., Della Pietra, V.J., and Lai, J.C. Class-based n-gram models of natural language. *Computational Linguistics* 18, 4 (1992), 467–479.
- Caliskan, A., Bryson, J.J., and Narayanan, A. Semantics derived automatically from language corpora contain human biases. *Science* 356, 6334 (2017), 183–186.
- Collbert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. 2011. Natural language processing (almost) from scratch. *J. Machine Learning Research* 12 (2011), 2493–2537.
- Dai, A.D. and Le, Q.V. Semisupervised sequence learning. In *Proceedings of Advances in Neural Information Processing Systems*, 2015.
- Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., and Harshman, R.A. Indexing by latent semantic analysis. *J. Amer. Soc. Information Science* 41, 6 (1990), 391–407.
- Devlin, J., Chang, M.W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of 2019 NAACL*.
- Eisenstein, J. *Introduction to Natural Language Processing*. MIT Press, 2019.

- Faruqui, M., Dodge, J., Jauhar, S.K., Dyer, C., Hovy, E., and Smith, N.A. Retrofitting word vectors to semantic lexicons. In *Proceedings of 2016 NAACL*.
- Faruqui, M. and Dyer, C. Improving vector space word representations using multilingual correlation. In *Proceedings of 2014 EACL*.
- Fellbaum, C. (Ed.). *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
- Firth, J.R. A synopsis of linguistic theory 1930–1955. *Studies in Linguistic Analysis*, Blackwell, 1957, 1–32.
- Goldberg, Y. *Neural Network Methods for Natural Language Processing*. Morgan & Claypool, 2017.
- Goldberg, Y. Assessing BERT's syntactic abilities. 2019; arXiv:1901.05287.
- Gururangan, S., Dang, T., Card, D., and Smith, N.A. Variational pretraining for semi-supervised text classification. In *Proceedings of 2019 ACL*.
- Howard, J. and Ruder, S. Universal language model finetuning for text classification. 2018; arXiv:1801.06146.
- Jurafsky, D. and Martin, J.H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (3rd Ed). Prentice Hall, forthcoming.
- Lample, G. and Conneau, A. Cross-lingual language model pretraining. In *Proceedings of 2019 NeurIPS*.
- Ling, W. et al. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of 2015 EMNLP*.
- Liu, N.F., Gardner, M., Belinkov, Y., Peters, M.E. and Smith, N.A. Linguistic knowledge and transferability of contextual representations. In *Proceedings of 2019 NAACL*.
- Liu, Y. et al. RoBERTa: A robustly optimized BERT pretraining approach. 2019; arXiv:1907.11692.
- Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. In *Proceedings of 2013 ICLR*.
- Mulcaire, P., Kasai, J., and Smith, N.A. Polyglot contextual representations improve crosslingual transfer. In *Proceedings of 2019 NAACL*.
- Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., and Smith, N.A. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of 2013 NAACL*.
- Pang, B. and Lee, L. *Opinion Mining and Sentiment Analysis. Foundations and Trends in Information Retrieval, Vol. 2*. Now Publishers, Inc. 2008.
- Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations. In *Proceedings of 2018 NAACL*.
- Peters, M.E., Neumann, M., Zettlemoyer, L., and Yih, W. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of 2018 EMNLP*.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. and Sutskever, I. Language models are unsupervised multitask learners. 2019.
- Raffel, C. et al. Exploring the limits of transfer learning with a unified text-to-text transformer. 2019; arXiv:1910.10983.
- Sundermeyer, M., Schlüter, R., and Ney, H. LSTM neural networks for language modeling. In *Proceedings of 2012 Interspeech*.
- Turney, P.D. and Pantel, P. From frequency to meaning: Vector space models of semantics. *J. Artificial Intelligence Research* 37, 1 (2010), 141–188.
- Yang, Z., Dai, Y., Yang, Y., Carbonell, J., Salakhutdinov, R. and Le, Q.V. XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of Advances in Neural Information Processing Systems*, 2019.
- Zellers, R., Bisk, Y., Schwartz, R., and Choi, Y. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of 2018 EMNLP*.

Noah A. Smith (nasmith@cs.washington.edu) is a professor in the School of Computer Science and Engineering at the University of Washington, and senior research manager for the AllenNLP team at the Allen Institute for Artificial Intelligence, Seattle, WA, USA.

© 2020 ACM 0001-0782/20/6 \$15.00.



Watch the author discuss this work in the exclusive *Communications* video. <https://cacm.acm.org/videos/contextual-word-representations>

Strategically augmented street lamps can become the key enabling technology in smart cities.

BY MAX MÜHLHÄUSER, CHRISTIAN MEURISCH, MICHAEL STEIN, JÖRG DAUBERT, JULIUS VON WILLICH, JAN RIEMANN, AND LIN WANG

Street Lamps as a Platform

STREET LAMPS CONSTITUTE the densest electrically operated public infrastructure in urban areas. Their changeover to energy-friendly LED light quickly amortizes and is increasingly leveraged for smart city projects, where LED street lamps double, for example, as wireless networking or sensor infrastructure.

We make the case for a new paradigm called SLaaP—street lamps as a platform. SLaaP denotes a considerably more dramatic changeover, turning urban light poles into a versatile computational infrastructure. SLaaP is proposed as an open, enabling platform, fostering innovative citywide services for the full range of stakeholders and end users—seamlessly extending from everyday use to emergency response. In this article, we first describe the role and potential of street lamps and introduce one novel base service as a running example. We then discuss citywide infrastructure design and operation, followed by addressing the major layers of a SLaaP infrastructure: hardware, distributed software platform, base services, value-added services and applications for users and ‘things.’ Finally, we discuss the crucial roles and participation of major stakeholders: citizens, city, government, and economy.

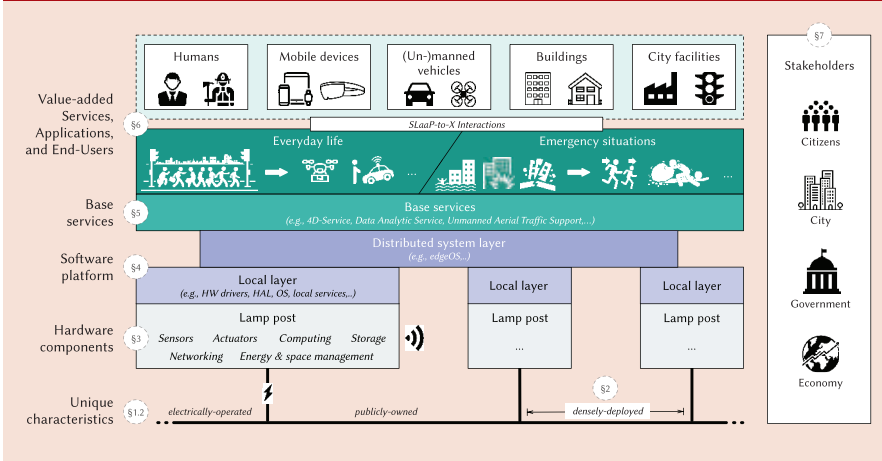
Recent years have seen the emer-

gence of smart street lamps, with very different meanings of ‘smart’—sometimes related to the original purpose as with usage-dependent lighting, but mostly as add-on capabilities like urban sensing, monitoring, digital signage, WiFi access, or e-vehicle charg-

» key insights

- **As the densest electrically operated public infrastructure, interconnected augmented street lamps are the most promising scalable platform for countless innovative smart city services.**
- **Street lamps are currently projected in an adhoc way; neglected sovereign duties may inhibit crucial potentials. We propose street lamps as a platform (termed SLaaP) that could become a true novel city infrastructure and discuss its major constituents, including the computational platform and base services.**
- **The case for SLaaP as proposed here clearly leads to interesting research challenges, both in ubiquitous computing and of interdisciplinary nature.**

Figure 1. An overview of the proposed citywide infrastructure based on augmented street lamps, its potential end-users/'things' and the stakeholders, as well as the remaining structure of this article (gray circles).



ing.^a Research about their use in settings for edge computing¹⁴ or car-to-infrastructure communication (for example, traffic control, hazard warnings, or autonomous driving)⁶ hints at their great potential as computing resources. The future holds even more use cases: for example, after a first wave of 5G mobile network rollouts from 2020 onward, a second wave shall apply mm-wave frequencies for which densely deployed light poles can be appropriate ‘cell towers.’

Street lamps: A (potential) true infrastructure. Given the huge potential of street lamps evident already today and given the broad spectrum of use cases, a city’s street lamps may obviously constitute a veritable infrastructure. However, cities today do not consider street lamps—beyond the lighting function—as an infrastructure in the strict sense. Like road, water, energy, or telecommunication, infrastructures constitute a sovereign duty: provision and appropriate public access must be regulated, design and operation must balance stakeholder interests, careful planning has to take into account present and future use cases and demands, maintenance, threat protection, and more. Well-considered outsourcing or privatization may be aligned with these public interests.

The LED dividend: A unique opportunity. The widespread lack of such considerations in cities is even more dramatic since a once-in-history oppor-

tunity opens up with the changeover to energy efficient LED lighting, expected to save large cities millions in terms of energy cost, as we will discuss, called ‘LED dividend’ in the following. Given their notoriously tight budgets, cities urgently need to dedicate these savings if they want to ‘own’ and control an infrastructure, which, once built, can foster innovation and assure royalties and new business as sources of city and citizen prosperity.

Computing platform and extensible hardware: The indispensable core. It is common knowledge that in the current era of digitization, computers are at the core of most innovation and added value. Large consortia and initiatives like OpenFog and MEC (mobile edge computing)³⁰ arose around the conviction that ‘the cloud’ will, at least in part, move closer to the user and to the applications. Reasons include the increasing need for real-time (short delay, reliably connected) computing and resource-demanding AI algorithms that overstrain mobile devices’ batteries or compute power but are too bandwidth-demanding to be offloaded to a distant cloud. Many services and applications discussed in the article, such as our running example, support these arguments.

It seems obvious that computing and storage resources should be the core of a true smart street lamp infrastructure, at least in part directly integrated with the lamp posts. A second absolutely crucial characteristic of such a true infrastructure is obviously the versatility and extensibility of lamp posts. They must be prepared for exchange and extension

with respect to electricity, mounting space (partly with line-of-sight to the urban scene), weather/vandalism protection, and ease of (dis-)mounting, among others. These characteristics are not automatically a primary interest of street lamp customers and providers as they cannot be easily matched with aesthetics and competitive pricing.

Our call stands in stark contrast to reality, namely mostly project-based extension of street lamps. Such projects follow opportunities, perceived pressing needs, or selective flagship efforts. Moreover, the LED dividend is often spent in a short-sighted manner. Most city councils ignore reasons to become edge-computing providers; just as many cities took too long to realize the need to become WiFi providers, they may ‘wake up’ too late here, this time losing a historic opportunity to be in the driver’s seat of a novel true infrastructure and its potentials (for innovation, citizen wealth, or city income) as well as its risks (to privacy, dependability, and so on). This may also erect a classical innovation barrier where application providers wait for infrastructures (and hence, a customer base) and infrastructure providers wait for application demands.²⁶

To summarize these arguments, we proposed a citywide true infrastructure based on augmented street lamps as a platform (termed SLaaP) that can bootstrap smart cities and enrich them with novel services, ensure stakeholder trade-offs (such as data analytics and novel services versus privacy risks) and extend seamlessly to sovereign interests such as emergency preparedness and response, safety, and security. Figure 1 presents an overview of this new infrastructure and the remaining structure of this article. Authorities must define and assume their sovereign role and ensure the dedication of the LED dividend to this historic duty. Versatile extensible hardware and integrated compute infrastructure and base services must form the base of this effort.

Unique characteristics and opportunity. Street lamps are a basic and important facility of cities, illuminating roads and sidewalks in order to increase the safety of road users and pedestrians’ sense of security. This leads to three characteristics that make them highly attractive from an ICT perspective and for smart city concepts.

a <https://smight.com>; <https://www.schreder.com>; <https://www.stengg.com>

► *Electrically operated.* In most cities, street lamps are connected to subterranean power grids, which are usually separated from the main power grid. Most of them are still only powered from dusk till dawn since the lamps lack operable switches (power-on means light-on). Later, we will discuss the resulting considerations for power supply, but the existing power lines definitely predestine street lamps as a digital infrastructure, augmented with computing, networking, and Internet-of-Things (IoT) components.¹

► *Densely deployed.* Due to their dense deployment along roads and sidewalks, street lamps are already ubiquitous in our everyday urban life. In view of the use as digital infrastructure, this dense deployment makes them accessible anywhere in the city and provides high scalability due to tens and hundreds of thousands of instances per city.

► *Publicly owned.* Public ownership is ideal for assuring a true infrastructure as explained (compared to sovereign duties like assuring no access or use-case discrimination, or emergency operation). When regulations are established and enforced, privatization is possible, but the inverse, that is, turning private goods into a veritable infrastructure is socially unacceptable. This notion rules out the consideration of electrically operated and densely deployed devices under private ownership like wireless routers, which could, in principle, be qualified as ICT infrastructure²² (yet hardly in the IoT respect that is relevant for cities).

These three characteristics qualify street lamps as the scalable vehicle for a novel urban digital infrastructure—SLaaP. As discussed, such an infrastructure has extensible lamp post hardware and a general-purpose computing platform (a distributed edge cloud) as its basic constituents (as illustrated in Figure 2). However, the success of this infrastructure comes from enablements and services, such that the selection of provided hardware add-ons and base services as well as a proper bootstrapping of them will be crucial—and hence the key aspects of this article.

Running example: 4D-service for urban scene modeling. Since many arguments and sample calculations can be backed by relating them to one of the SLaaP base services we will discuss

Figure 2. Our visionary view of augmented street lamps (top); acting as a citywide infrastructure in future smart cities for its new kinds of applications and end-users/'things' (bottom).

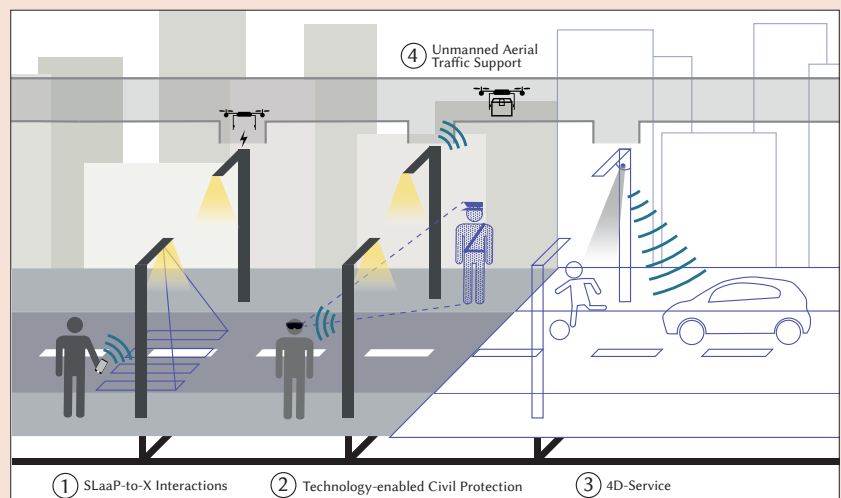
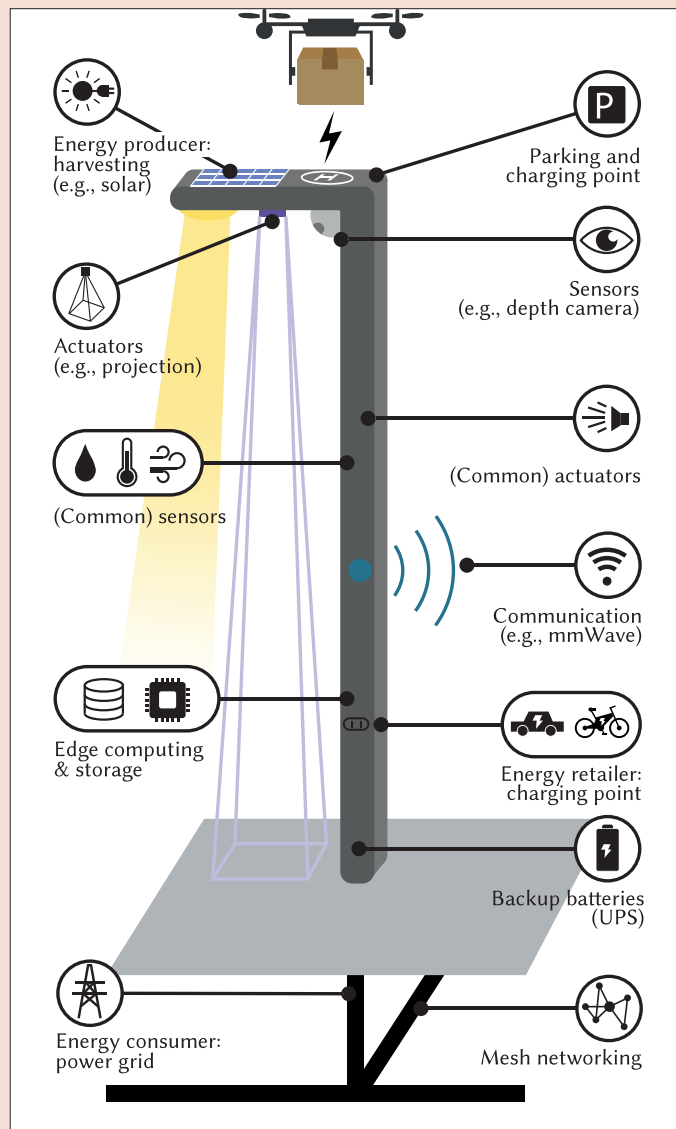


Figure 3. Geographic density of public street lamps in urban areas (left) and resulting wireless mesh networks ($d = 50\text{m}$) of interconnected inner-city street lamps (right), presented by the example of Darmstadt (Germany).



later, we focus on a particular one here called 4D-service. This service is particularly demanding and innovative, thus serving as a challenge for other SLaaP components and aspects. It can drive innovation in virtually every application domain of smart cities. It is related to the fact that augmented and virtual reality (AR/VR) are so eagerly pursued in industry that press tends to speak of a race between major companies.^b In contrast to this race for 3D models and corresponding AR/VR apps in many domains, the smart city domain is still one where software is for the most part bound to 2D maps: navigation and routes, location information (with respect to businesses, or friends, among others), situational awareness, city planning, and operations software, and so on. The 4D-service shall enable smart cities to move from 2D map-based apps to 3D AR/VR based ones.

To this end, street lamps equipped with 3D capturing hardware (LiDAR and/or cameras) shall provide a constant stream of visual “city situation” information, distinguishing the immobile city scenery (buildings, roads, or trees) from mobile elements (pedestrians or vehicles). The captured scenes are automatically processed, populating a semantic-annotated 3D model of the city. Since time-series of 3D models are captured, a fourth dimension comes into play (hence the term 4D). For public and standard app use, the mobile elements

are replaced by semantically equivalent models, such that individual persons, cars, etc. cannot be detected. Access to privacy-sensitive information is tightly controlled, especially to the encrypted true representation of mobile entities stored at the edge and kept for a determined time span (rolling overwrite). The resulting 4D model enables or supports various urban use cases over multiple timescales (see Figure 2), for example:

- ▶ highly demanding, real-time services, especially for 3D-AR apps serving mobile users (pedestrian, public/individual transport, or handicapped) with smartphones or head-mounted/head-up displays;
- ▶ realistic simulations for urban development/planning, people flow, or impact analyses;
- ▶ reliable, multi-perspective model of the “street situation,” enhancing the on-board model of autonomous vehicles for making faster, safer decisions²⁴ or enabling mobility control and traffic support (for example, parking lot search) without specialized hardware;
- ▶ emergency response and management support, for example, in detecting buried people, assessing their health states (triage), selectively searching missing people through face recognition; and,
- ▶ immersive visualization and interaction in VR, for example, planning and performing evacuations, time travel, city walks, tourism, gaming, sales, and marketing, among others.

The following section discusses the

layers of SLaaP from the bottom to top, preceded by a consideration of the overall design and operation. We will elaborate on new (interdisciplinary) research opportunities linked to the opportunities and challenges of SLaaP as a true citywide digital infrastructure.

SLaaP Infrastructure: Design and Operation

A citywide infrastructure requires geospatial planning and operation; in particular, coverage of components (computing resources, scene capturing hardware, and others) must tradeoff economics with technical and application necessities (for example, sufficient density for meshing or coverage of hotspots interesting for 4D-modeling) sporadic addition of lamp posts may be needed.

In order to provide insights into this issue, we conducted a case study in Darmstadt^c—a typical mid-size city (some 160,000 inhabitants) in Germany. Figure 3 shows the real-world geographic distribution of the 14,331 publicly owned street lamps (yellow dots) in Darmstadt. We see the distribution of public street lamps is particularly dense along streets and sidewalks in public or populated areas (for example, downtown, parks, and residential areas) sparse in industrial or privately owned areas, where property owners are responsible for the lighting. This distribution reflects a representative street lamp deployment pattern in urban environments; cities only vary in the regulated distance between two neighboring street lamps. In our Darmstadt example, the distance is about $25.2 \pm 9.2\text{m}$ and normally distributed (see Figure 4). These short distances between neighboring street lamps allow the interconnecting of them via high-bandwidth wired (for example, optical fiber, Powerline) or wireless technologies (for example, WiFi, mmWave) to mesh networks across the city (as shown in Figure 3), addressing scalability. A street lamp of the citywide mesh network can further act as a nearby access point of urban services to address accessibility. For instance, assuming a

^c Darmstadt has won the prestigious national competition “Digital City” in 2017, aiming to become the digital model city for Germany and even for Europe; <https://digitalstadt-darmstadt.de>

^b For example, see <https://cnet.co/38XAbcK>.

realistic outdoor wireless range of 50m and if only 30% (70%) of all 5,608 street lamps in the inner city (14.57km²) are upgraded, that is, 1,682 (3,926) street lamps or 115.4/km² (269.5/km²), nearly half (two thirds) of the inner city can already be covered spatially,¹¹ where (mobile) endusers can access services with low latency and high bandwidth (see Figure 4). This case study gives an impression of SLaaP’s potential as a decentralized, stationary, and predictable infrastructure for urban services.

SLaaP Hardware Components

To realize the infrastructure envisioned here, conventional lamp posts first must be upgraded or replaced by augmented street lamps (termed ASL), pursuing a modular and exchangeable design that is appropriate to the desired services. For instance, the proposed 4D-service requires sensors (for example, depth cameras) that need a free field of view on the traffic from the top; such components should either be attached within a kind of crow’s nest to the pole or integrated into it with inspection windows. It is important to understand that not all of the following elaborated (exemplary) components must be part of each lamp post but can be strategically distributed across the city to well-selected lamp posts, depending on the service demands and budget:

Sensors and actuators. Today’s lighting industry starts equipping street lamps with first sensors and actuators, for example, for a smart lighting control.¹⁰ Recently, an EU-wide research initiative has proposed The Humble Lamppost project that aims to deploy 10 million ‘smart’ lamp posts across EU

cities, addressing first use cases, such as traffic monitoring or environmental data acquisition.²⁰ Besides the currently considered sensors (for example, air quality, noise, and temperature) and actuators (for example, displays or speakers), new promising hardware components should be investigated to enable innovative applications. The proposed 4D-service, for instance, requires economic, compact depth cameras like range-extended Intel RealSense (\$200) or 360° Velodyne Puck Lite LiDAR (\$3k)—collecting point clouds up to 20Hz—to capture the dynamic 3D urban scene over time. Since the latter only has 16 slices at $\pm 15^\circ$ vertical FoV, challenges must be solved to get a complete scene model: for example, using a continuously rotating/tilting mount, or fusing point clouds from well-aligned LiDAR systems of neighboring street lamps. Next-generation short-throw or laser based projectors may serve as actuators, providing situation-aware, personalized information to citizens (for example, in evacuation situations).

Computing and storage. The emergence of latency-critical use cases like AR or autonomous driving and resource-intensive services like the proposed 4D-service requires nearby computational resources (fog/edge computing, edge-Clouds, cloudlets^{23,26}). Integrating computing power into street lamps is a completely undervalued aspect: street lamps with their unique characteristics can actually enable an economic large-scale deployment of cloudlets (despite their range restrictions), making the breakthrough of practicable edge computing at long last.

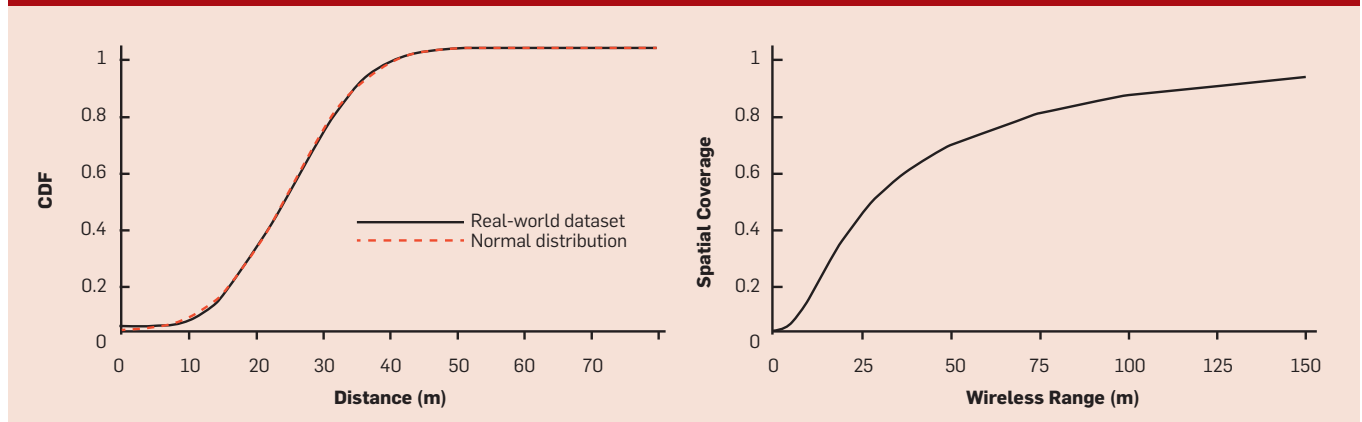
We propose upgrading street lamps

should start with single-board computers (SBC)—having a fair trade-off between good performance, low energy consumption, compact size, and low prize.¹⁵ For instance, low-energy ARM-based Raspberry PI (\$40) or Odroid (\$60) would be fine for a minimalist setup to basically analyze sensor data or reduce network traffic. Applying advanced machine learning (ML) or computer vision-based algorithms, as required for the privacy-preserving modeling or semantic scene analysis of the 4D-service,²⁴ performant but economic hardware like 64-bit x86 quad-core Udoo Ultra (\$300) or Udoo Bolt with GPU support (\$600) is required for real-time capabilities. It is further important that the processors used support trusted computing features (for example, ARM TrustZone, Intel SGX³) to protect the provider’s intellectual property—that is, their ML algorithms/models—on (untrustworthy) devices.

Caching and storage integrated into street lamps can further enable novel research concepts,⁹ reducing redundant or unnecessary network traffic, and thus, preventing traffic (over)load in the core network. Related to the 4D-service example, privacy-critical scene elements should not be transferred (for example, to the cloud) but stored locally in encrypted form to protect privacy; for crime or accident investigations, access to the full imagery of the scene may be granted but restricted to law enforcement agencies.

Networking and communication. Connectivity to the Internet, end-user/things, or other street lamps is an essential ability of the SLaaP concept; however, to this day, the possible advantages of

Figure 4. Case study in the city of Darmstadt (Germany): distance distribution of public street lamps to their nearest neighbor (left), and spatial coverage of augmented street lamps as function of wireless ranges (right).



Numerical Example Illustrating the Feasibility of a Resilient Infrastructure

Energy harvesting. Theoretically, a solar cell generates about $100\text{mW}/\text{cm}^2$ during the daytime, which is however highly influenced by many factors (for example, time of day, or seasonal weather).¹⁸ Assuming a realistic conversion efficiency rate of 10%-20% on the Earth's surface and a typical LED lamp head of $75 \times 40 \text{cm} = 0.3\text{m}^2$, on which a solar panel can be mounted 'invisibly,' we get an average energy harvesting level of $100\text{mW}/\text{cm}^2 * 0.3\text{m}^2 * 0.1-0.2 = 30\text{W}-60\text{W}$. Often used but 'visible' solar panels have an area of 1m^2 (\$100) and would generate about the triple: $100\text{W}-200\text{W}$.

Energy consumption. Despite new LED technology, the lightning still consumes $15\text{W}-100\text{W}$ depending on the model and luminance. A proposed lightweight upgrade required for our 4D-service example, consisting of Raspberry PI3 ($\leq 5\text{W}$) and Intel RealSense depth camera ($2\text{W}-3\text{W}$), would consume less than 10W in total.¹⁵ A more powerful yet compact upgrade, consisting of Udoo Ultra/Bolt ($\leq 12\text{W}/25\text{W}$) and Velodyne Puck Lite (8W), would require around $20\text{W}-33\text{W}$.

this decentralized infrastructure are not nearly exploited as they could be (see Figure 3). One reason could be that there is no economic one-size-fits-all solution to act as high-performance backhaul technologies for decentralized street lamp networks. For latency-critical applications, either fiber or 5G connectivity would be a viable option, but at a high cost.²⁷ Also, a subset of street lamps could be connected to a fiber network and act as gateway nodes for the street lamp network. The fiber connected lamps can wirelessly relay traffic to other lamps in proximity by either WiFi or mmWave communications (street lamps are typically in line of sight with each other; as illustrated in Figure 4). Depending on the technologies used (and their inherited wireless range), strategic placement on street lamps is crucial; combined with mobile base stations, a nearly citywide spatial coverage can be reached, and many street lamps can be left out—a more detailed coverage analysis is given in Gedeon et al.¹¹ Depending on the depth of relaying, support of low-latency applications might be limited. Since mmWave offers multi-G throughput,³² bandwidth should not be an issue. In a second wave (after the first wave of 5G mobile network rollouts from 2020 onward), densely deployed street lamps can be appropriate cell towers for applying such mm-wave frequencies. A WiFi-mesh could act as backhaul technology between street lamps for various best-effort services; this would not allow offering hundreds of M to/from individual

lamp posts. However, such a decentralized mesh network would allow vital emergency communications, avoiding dysfunctions and supporting the ICT resilience of smart cities.

Energy and space management. An ASL can play three different roles—consumer, producer and retailer—at the same time. As a consumer, street lamps need power to primarily operate their lighting but also the aforementioned ICT components. For this, an ASL relies on a balanced combination of power from the subterranean power grid, as noted earlier, integrated backup batteries (UPS), and energy harvesting (for example, solar) acting as a producer. The latter two are optional but can transform an ASL into a largely energy self-sufficient infrastructure when the regular power source fails, say, in emergency situations like blackouts—supporting the ICT resilience of digital cities by avoiding dysfunctions and providing critical emergency services (for example, decentralized emergency communication) in an accelerated way. Although the harvested energy can be sufficient to cover the ASL's power consumption under best conditions, the harvesting level varies widely and is not reliable; therefore, an ASL must manage the available energy and balance the load by limiting non-critical ICT functionalities or dimming the light situation-consciously. As a retailer, an ASL can provide charging points for electric vehicles (cars or drones) requiring permanent current from the fixed power grid. In UAV research, in

particular, challenges such as 'endless' flying can now be practically addressed, for example, by recharging or automatically replacing the battery⁸ at the designed parking place on top of ASLs. (For more details, see the accompanying sidebar.)

SLaAP Software Platform

To exploit the full potential of this decentralized infrastructure of interconnected ASLs, distributed concepts/methods need to be applied and refined to its characteristics. We recommend using a two-layered software platform—consisting of the local layers and a distributed system layer (see Figure 1).

Local layer. This layer running on a single street lamp should consist of three sublayers: A modular hardware abstraction layer—to communicate with the various lower level components while preventing direct access to the hardware; an appropriate operating system (OS)—to manage the underlying hardware resources, enable controlled software executions (for example, permission model, process isolation, low-level power management), and support container-based virtualization, such as HypriotOS or balenaOS for low-energy ARM-based SBCs like Raspberry PI or Odroid, Alpine or CoreOS for x86-SBCs like Udoo; and, a local service layer—providing software-defined basic cloudlet functionalities, storage, energy, and user/things management locally. Depending on the given real-time requirements (for example, in the order of 0.01s for supporting autonomous driving), optimizing these sublayers by an accelerated computing pipeline is crucial for specific urban services.

Distributed system layer. This layer runs as an overlay over the street lamps' local layers and the interconnecting mesh networks, being responsible for the routing, distributed processing, and storage. We envision a distributed OS at the edge (termed *edgeOS*), supporting both vertical and horizontal balancing in the given 3n-tier architecture.^d Vertically, communication issues (such as latency, network traffic, bandwidth)

^d We introduce a novel 3n-tier architecture, extending the traditional "device-cloud" paradigm to a three-tier "device-edge cloud" setting (vertical), where n interconnected edge nodes (that is, street lamps) participate in the middle tier (horizontal).

should be primarily addressed, while horizontally computing issues (for example, distributed in-network processing) play a primary role. Although latest research,²⁶ large consortia and initiatives like OpenFog and MEC³⁰ have already investigated and standardized partial aspects for each direction, important challenges are still open for making the breakthrough. Particularly, we recommend focusing on three neglected aspects: The intelligent, economic interplay between both directions to incorporate the resources from all three tiers; result delivery in high-dynamic environments, for example, through mobility predictions; and the methods to efficiently distribute AI algorithms in the 3n-tier architecture.

Due to the strict responsiveness requirements and high mobility of end users, these aspects are difficult to achieve with traditional ways of developing an application as a single monolith. We propose to base this layer on two concepts namely microservices² and serverless computing:²⁸ applications are developed as a set of small individual functions that are instantiated and executed on demand. SLaaP must take care of the operation of the edge resources through unified resource management, the provisioning of the functions (for example, cold launch times in microseconds), and the optimization of the system to achieve global goals, such as high resource utilization and guaranteed service quality.²⁹ Additionally, secure network channels and so-called enclaves—private, encrypted memory regions on corresponding SGX-enabled processors—should be used (termed trusted edge computing) to protect the provider's intellectual property.³

SLaaP Base Services

Besides our running example of the 4D-service discussed previously, we now illustrate two more exemplary base services to show how they can be enabled at all or applied in a novel large-scale way. Such services can be used by other base services or (third-party) value-added applications. Hence, it is crucial that these services must be highly reliable and controlled by a certified/government agency.

Unmanned aerial traffic (UAV) support. Latest industry research of logistic companies (for example, DHL Parcelcopter,



Street lamps with their unique characteristics can actually enable an economic large-scale deployment of cloudlets, making the breakthrough of practicable edge computing at long last.



Amazon Prime Air)^e uses UAVs for an innovative parcel delivery service. However, a limited flying range and unresolved safety issues arising from the necessity of flying over residential areas technically block the breakthrough in all these cases.¹⁹ ASLs can notably accelerate the everyday suitability of UAVs in urban areas by playing a key role in overcoming both issues: Street lamps provide recharging stations mounted on top of them, extending the UAV's flying range; and street lamps acting as waypoints^f span a virtual air corridor along them, that is, a citywide network of air routes, limiting the possible crash sites. Flying along these defined and controlled flight paths in combination with the 4D-service for tracking and positioning can enable fully automated “aerial highways” (noted in Figure 2). For safety and security—the main reasons blocking their breakthrough—UAVs may have an integrated or parachute/airbag system; street lamps can operate a catching and safety system; a “traffic police” service¹⁹ can intervene by controlling a prescribed, independently working UAV security chip that overwrites the general software and enforces controlled landing. Using these air routes, UAV job services such as an address-independent people-to-people packet transport, or event filming can be established. In emergencies, these UAVs can also assigned to a secondary use, supporting disaster management by flying rescue missions,⁷ providing projected evacuation advice,¹⁷ and so on.

Technology-enabled civil protection. ASLs acting as a citywide surveillance infrastructure may support counterterrorism and crime fighting. For instance, a combination of sensors (acoustic, optical, others) mounted on the street lamp could detect gunshots or explosions, allowing fast response and the identification of perpetrators. Assessing situations with regard to risks and avoiding these risky areas, a remote or virtual companion can escort and guide a user to his or her destination. To prevent or detect a serious crime, police can appear by combining so-called holoportation,²¹ which can rely on the proposed 4D-ser-


e <http://bit.ly/2M4sis5>

f A waypoint is similar to the reference point of landing within the radio-navigation system (called ILS) for aircraft, which provides aircraft with horizontal and vertical guidance.


vice and thus become ubiquitous, with 3D projection techniques to produce a hologram of a police officer. The holographic officer, who can be seen, heard, and interacted with almost as if he was present, serves as a deterrent and victim support until the real police officers or paramedics arrive on the scene. Moreover, an injured person gets holographic victim support in disasters until the rescuers arrive physically. However, systematic further development of such promising techniques is necessary to make the holographic scene more realistic and immersive.

SLaaP Value-Added Services, Applications, and End Users

More than the base services, SLaaP facilitates novel value-added services and applications for both humans and things, covering or digitally developing several application domains: tourism, urban mobility, marketing, sales, gaming, city planning, logistics, environmental affairs, sustainable cities, to name a few.¹⁶ These new (qualities of) applications enabled by SLaaP lead to a novel kind of interactions, called *SLaaP-to-X* interactions; *X* represents the possible end users: Humans and mobile devices; vehicles of all kind; and city inventory (for example, buildings, places, hotspots) and facilities. For instance, SLaaP can stage an intelligent personal cloudlet that moves along the user's path and personalizes his environment, such as, displaying personal navigation/evacuation advice through laser-based projection; humans can further interact with ASLs in novel ways (say, through vision-based gesture interactions), raising challenges like obtrusiveness, occlusion, and privacy in the public multi-user environment. As a static supportive infrastructure, SLaaP can complement autonomous driving/flying and tackle safety and security issues by reliably detecting obstacles (for example, pedestrians, accidents, roadworks) or the road surface state using the proposed 4D-service. Finally, SLaaP can be the entry point into the smart city for smart buildings/homes, exchanging data for intelligent lighting and energy management, among others. As the city's nervous system, SLaaP can detect environmental changes that impact the city and autonomously adapt facilities to respond to such events (traffic regulation).



A holographic police officer, who can be seen, heard, and interacted with almost as if he were present, serves as a deterrent and victim support until the real officers or paramedics arrive on the scene.



Governance and Stakeholder Perspective

SLaaP promises an outstanding technological progress for smart city concepts and clearly offers many benefits for its end users, but there are still interdisciplinary research and actions required. Besides research, we identify four major stakeholders: citizens, city, government, and economy.

From the citizens' perspective. As we have learned from Gascó-Hernandez,¹⁰ a smart city must involve its most important component—its citizens—not only as recipients of its services but also as partners (civic participation) in deciding the type of city they want to live in. One crucial challenge lies in designing the ASL in ways that citizens socially and ethically accept—or merely tolerate—the new technology, considering a socio-technical design. At the same time, we need to be careful about security and privacy, especially if the infrastructure collects and processes user-related data.³¹ Otherwise, the consequences may be disastrous if the infrastructure is hacked or misused. Therefore, a secure data management (Databox⁵) and privacy-preserving data analysis techniques (for example, replacing privacy-critical 'objects' in the 4D-service by means of semantic modeling directly 'at the source') are desirable while ensuring a two-way data tracking (based on blockchains¹²): Access and usage control to ensure the traceability of the data use (say, for the compliance with the GDPR) and enforcing contracts or any data usage fees; and non-repudiation of data authorship through smart contracts to ensure the undeniable proof of the guaranteed data confidence.

From the city perspective. Approximately, 60–90 million street lamps exist across Europe; 75% are over 25 years old.⁸ It is therefore common for street lamps to consume up to 50% of a city's energy budget. Large-scale retrofitting/renewal programs (in LA or NYC^h) to highly efficient LED lamps quickly amortize and can save up to 70% of the energy costs, representing a once-in-history opportunity for economically upgrading to the proposed ASLs. On the

g <http://eu-smartcities.eu/initiatives/78/description>

h <http://bit.ly/34qDd5G>

other hand, there are challenges in energy-efficiently designing SLaaP, and finding a good cost-benefit trade-off of the deployed ICT components that SLaaP does not lead to disproportionately high costs. For instance, major cities can team up with industry and further push a standard (ImHLA¹³) to open up market competition in this segment, and to ensure compatibility and interoperability between different manufacturers.⁴ Beyond that, a city can offer economic incentives to companies and rent out infrastructure resources to (partly) cover the costs. Other important challenges include a strategic rethinking of urban development and planning—concerning the new augmented infrastructure and air corridors—and a reliable integration of city facilities (power plants, traffic facilities) and rescue organizations (fire/ambulance service, police).

From the government perspective. The government's role is to timely adopt laws that allow SLaaP with all its benefits, but at the same time, and to review legal issues with respect to privacy, ethics, applicability, and liability coming with the new (technological) opportunities. For instance, one could get the idea to use the 4D-service for solving the question of fault in road accidents or other incidents (similar to dash cams). For promises of safety, many citizens are likely to accept sacrificing—at least some—privacy. A governed disclosure strategy can be realized in serious threats only (for example, terrorist attacks, natural and human/technology-caused disasters): it would allow authorities (requiring N parties) and data owners (for alibi reasons) to carefully and selectively unhide private information to enable fast and focused emergency responses or counter-terrorism. To establish citywide air corridors for UAVs, corresponding regulations must be adjusted so that they can fly near people and safely coexist in everyday urban life.¹⁹ The prevailing legal situation will surely have the decisive influence on how UAVs can use and benefit from the new infrastructure.

From the economy perspective. SLaaP should be open for third parties, targeting a comprehensive involvement of industry. The challenges lie in providing economic incentives for companies to support the new infrastructure; and in

establishing an open urban platform with standardized interfaces,²⁵ allowing companies to offer their novel services and business models for different smart cities without the need for redeveloping. Overcoming these challenges, connected urban platforms available in smart cities within a country would finally pave the way for a smart nation.

Conclusion: A Unique Historic Opportunity

We argued and envisioned that augmented street lamps as a platform can be the key driver and enabling technology for smart cities, proposing a citywide true infrastructure for innovative urban applications. Versatile extensible hardware, integrated computer platform and base services must form the base of this effort. We further pointed out the once-in-history opportunity with the 'LED dividend' and made the case for paying more attention to this—seriously undervalued—public facility of street lamps from several research fields.

Acknowledgments

Some research reported in this article is based upon interdisciplinary work supported by the LOEWE initiative (Hessen, Germany) within the NICER project as well as by the German Research Foundation (DFG) as part of the Collaborative Research Center (CRC) 1053 – MAKI and as part of the project D4 within the RTG 2050 "Privacy and Trust for Mobile Users." Large parts of this work have been performed in the context of the LOEWE centre emergenCITY (for more details, please visit <https://www.emergencity.de>).

We also thank *Communications'* reviewers, Matthias Hollick, and Andrea Ortiz; the City of Darmstadt for the valuable street lamp data; and designer Simone Schroeder for her great graphics. **□**

References

- Atzori, L. et al. The Internet of Things: A survey. *Computer Networks* 54, 15 (2010), 2787–2805.
- Boucher, S. et al. Putting the "micro" back in microservice. *USENIX ATC*, 2018, 645–650.
- Brasser, F. et al. VoiceGuard: Secure and private speech processing. *INTERSPEECH*, 2018, 1303–1307.
- Breining, C. et al. Orchestrating Infrastructure for Sustainable Smart Cities. White Paper, IEC (2014).
- Chaudhry, A. et al. Personal data: Thinking inside the box. *Critical Alternatives*. Aarhus University Press, 2015, 29–32.
- Coppola, R. et al. Connected car: Technologies, issues, future trends. *ACM Computing Surv.* 49, 3 (2016), 46.
- Erdelj, M. et al. Help from the sky: Leveraging UAVs for disaster management. *IEEE Pervasive Computing* 16, 1 (2017), 24–32.
- Fujii, K. et al. Endless flyer: A continuous flying drone

- with automatic battery replacement. *UIC/ATC*. IEEE, 2013, 216–223.
- Garcia Lopez, P. et al. Edge-centric computing: Vision and challenges. *ACM SIGCOMM Computer Communication Review* 45, 5 (2015), 37–42.
- Gascó-Hernandez, M. Building a smart city: Lessons from Barcelona. *Commun. ACM* 61, 4 (Apr. 2018), 50–57.
- Gedeon, J. et al. From cell towers to smart street lamps: Placing cloudlets on existing urban infrastructures. *SEC*, 12018, IEEE, 87–202.
- Halaburda, H. Blockchain revolution without the blockchain? *Commun. ACM* 61, 7 (July 2018), 27–29.
- Heuser, L. et al. Humble Lamppost DIN SPEC 91347: Integrated Smart Lighting Infrastructure., 2017.
- Jia, G. et al. SSL: Smart street lamp based on fog Computing for Smarter Cities. *IEEE Trans. Industrial Informatics* (2018).
- Kaup, F. et al. Energy models for NFV and service provisioning on fog nodes. In *Proceedings of NOMS'18*. IEEE, 2018, 1–7.
- Khatoun, R. et al. Smart cities: Concepts, architectures, research opportunities. *Commun. ACM* 59, 8 (Aug. 2016), 46–57.
- Knierim, P. et al. Quadcopter-projected in-situ navigation cues for improved location awareness. In *Proceeding of CHI 2018*. ACM, 433.
- Ku, M.-L. et al. Advances in energy harvesting communications: Past, present, and future challenges. *IEEE Communications Surveys & Tutorials* 18, 2 (2016), 1384–1412.
- Menouar, H. et al. 2017. UAV-enabled intelligent transportation systems for the smart city: Applications and challenges. *IEEE Commun.* 55, 3 (2017), 22–28.
- Nahrstedt, K. et al. Smart communities Internet of Things, 2016; arXiv:1604.02028.
- Orts-Escolano, S. et al. Holoportation: Virtual 3D teleportation in real-time. In *Proceedings of UIST 2016*. ACM, 741–754.
- Panitzek, K. et al. Can we use your router, please? Benefits and implications of an emergency switch for wireless routers. *IJISCRAM* 4, 4 (2012), 59–70.
- Perera, C. et al. Fog computing for sustainable smart cities: A survey. *ACM Computing Surv.* (2017), 32:1–43.
- Qiu, H. et al. AVR: Augmented vehicular reality. In *Proceedings of MobiSys 2018*. ACM.
- Santana, E. et al. Software platforms for smart cities: Concepts, requirements, challenges, and a unified reference architecture. *ACM Computing Surv.* 50, 6 (2017), 78.
- Satyanarayanan, M. The emergence of edge computing. *Computer* 50, 1 (2017), 30–39.
- Simsek, M. et al. 5G-enabled tactile Internet. *IEEE J-SAC* 34, 3 (2016), 460–473.
- Wang, L. et al. Peeking behind the curtains of serverless platforms. *USENIX ATC*, 2018, 133–146.
- Wang, L. et al. Service entity placement for social virtual reality applications in edge computing. In *Proceedings of INFOCOM 2018*. IEEE, 468–476.
- Wang, S. et al. A survey on mobile edge networks: Convergence of computing, caching and communications. *IEEE Access* 5 (2017), 6757–6779.
- Zhang, K. et al. Security and privacy in smart city applications: Challenges and solutions. *IEEE Commun.* 1 (2017), 122–129.
- Zheng, K. et al. 10 Gb/s hetsnets with millimeter-wave communications: Access and networking challenges and protocols. *IEEE Commun.* 53, 1 (2015), 222–231.

Max Mühlhäuser is a full professor at TU Darmstadt, Germany. He served as co-first author of this article.

Christian Meurisch is a research assistant at TU Darmstadt, Germany. He served as co-first author of this article.

Michael Stein is a postdoctoral researcher at TU Darmstadt, Germany.

Jörg Daubert is a postdoctoral researcher at TU Darmstadt, Germany.

Julius Von Willich is a research assistant at TU Darmstadt, Germany.

Jan Riemann is a postdoctoral researcher at TU Darmstadt, Germany.

Lin Wang is a postdoctoral researcher at TU Darmstadt, Germany.

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

Attention: Undergraduate and Graduate Computing Students

There's an **ACM Student Research Competition (SRC)**
at a SIG Conference of interest to you!



Association for Computing Machinery
Advancing Computing as a Science & Profession



It's hard to put the **ACM Student Research Competition** experience into words, but we'll try...



"Attending ACM SRC was a transformative experience for me. It was an opportunity to take my research to a new level, beyond the network of my home university. Most important, it was a chance to make new connections and encounter new ideas that had a lasting impact on my academic life. I can't recommend ACM SRC enough to any student who is looking to expand the horizons of their research endeavors."

David Mueller
North Carolina State University | SIGDOC 2018



"Participating in the ACM SRC was a unique opportunity for practicing my presentation skills, getting feedback on my work, and networking with both leading researchers and fellow SRC participants. Winning the competition was a great honor, a motivation to continue working in research, and a useful boost for my career. I highly recommend any aspiring student researcher to participate in the SRC."

Manuel Rigger
Johannes Kepler University Linz, Austria | Programming 2018



"The SRC was a great chance to present early results of my work to an international audience. Especially the feedback during the poster session helped me to steer my work in the right direction and gave me a huge motivation boost. Together with the connections and friendships I made, I found the SRC to be a positive experience."

Matthias Springer
Tokyo Institute of Technology | SPLASH 2018



"I have been a part of many conferences before both as an author and as a volunteer but I found SRC to be an incredible conference experience. It gave me the opportunity to have the most immersive experience, improving my skills as a presenter, researcher, and scientist. Over the several phases of ACM SRC, I had the opportunity to present my work both formally (as a research talk and research paper) and informally (in poster or demonstration session). Having talked to a diverse range of researchers, I believe my work has much broader visibility now and I was able to get deep insights and feedback on my future projects. ACM SRC played a critical role in facilitating my research, giving me the most productive conference experience."

Muhammad Ali Gulzar
University of California, Los Angeles | ICSE 2018



"At the ACM SRC, I got to learn about the work done in a variety of different research areas and experience the energy and enthusiasm of everyone involved. I was extremely inspired by my fellow competitors and was happy to discover better ways of explaining my own work to others. I would like to specifically encourage undergraduate students to not hesitate and apply! Thank you to all those who make this competition possible for students like me."

Elizaveta Tremsina
UC Berkeley | TAPIA 2018



"The ACM SRC was an incredible opportunity for me to present my research to a wide audience of experts. I received invaluable, supportive feedback about my research and presentation style, and I am sure that the lessons I learned from the experience will stay with me for the rest of my career as a researcher. Participating in the SRC has also made me feel much more comfortable speaking to other researchers in my field, both about my work as well as projects I am not involved in. I would strongly recommend students interested in research to apply to an ACM SRC—there's really no reason not to!"

Justin Lubin
University of Chicago | SPLASH 2018



"Joining the Student Research Competition of ACM gave me the opportunity to measure my skills as a researcher and to carry out a preliminary study by myself. Moreover, I believe that "healthy competition" is always challenging in order to improve yourself. I suggest that every Ph.D. student try this experience."

Gemma Catolino
University of Salerno | MobileSoft 2018

Check the SRC Submission Dates: <https://src.acm.org/submissions>

- ◆ Participants receive: \$500 (USD) travel expenses
- ◆ All Winners receive a medal and monetary award. First place winners advance to the SRC Grand Finals
- ◆ Grand Finals Winners receive a handsome certificate and monetary award at the ACM Awards Banquet

Questions? Contact Nanette Hernandez, ACM's SRC Coordinator: hernandez@hq.acm.org



research highlights

P. 86

Technical Perspective Algorithm Selection as a Learning Problem

By Avrim Blum

P. 87

Data-Driven Algorithm Design

By Rishi Gupta and Tim Roughgarden

Technical Perspective

Algorithm Selection as a Learning Problem

By Avrim Blum

THE FOLLOWING PAPER by Gupta and Roughgarden—"Data-Driven Algorithm Design"—addresses the issue that the best algorithm to use for many problems depends on what the input "looks like." Certain algorithms work better for certain types of inputs, whereas other algorithms work better for others. This is especially the case for NP-hard problems, where we do not expect to ever have algorithms that work well on all inputs: instead, we often have various heuristics that each work better in different settings. Moreover, heuristic strategies often have parameters or hyperparameters that must be set in some way.

Traditional theoretical analysis of such problems would select among algorithms or parameter choices by looking at their worst-case performance, such as via their approximation ratio. Another traditional alternative would be to consider average-case analysis, looking at performance on some clean (typically uniform) distribution over inputs. However, the inputs arising in a given domain are typically neither worst-case nor aver-

The authors present a theoretical formulation and analysis of algorithm selection using the well-developed framework of PAC-learning to analyze fundamental learning questions.

age-case from a clean distribution: instead, they often have structure and can be thought of as drawn from some complex and difficult-to-characterize probability distribution over inputs associated with the given domain. This fact suggests treating algorithm selection as a learning problem—using previously seen examples of inputs from a given domain to tune parameters of a heuristic or to select among various choices of algorithm. Indeed, this approach has been used in practice for many years, for example, Hutter⁵ and Smith-Miles.⁶ What the authors present is a theoretical formulation and analysis of this process, using the well-developed framework of PAC-learning from learning theory to analyze fundamental questions. For instance, how many examples of previous inputs does one need to see to have confidence that optimizing parameters over these examples will produce a rule that is near optimal on average for future inputs drawn from the same (complex and difficult-to-characterize) probability distribution? The paper then provides answers to such questions by analyzing formal measures of complexity (in particular, the pseudo-dimension) of various natural algorithm families such as greedy algorithms for knapsack and other object assignment problems, resulting in bounds on the number of previous inputs that must be seen for good generalization.

Subsequent to the initial publication of this paper there have been a number of exciting results building on the framework developed here, showing how principled algorithm selection and automated algorithm design more generally can be applied to a wide variety of other important problems. For instance, Balcan et al.^{3,4} analyze a large number of classic clustering problems, giving algorithms and sample complexity guarantees for learning near-optimal parameters

and hyperparameters for many popular clustering methods. Balcan, Dick, Sandholm, and Vitercik¹ show how learning-theoretic methods can be used to optimize branching in integer linear programming solvers. Most recently, Balcan, Dick, and Vitercik² show how principled data-driven algorithm design can be extended to the online setting in which there is no distribution over problem instances at all, just an arbitrary stream of examples, and one wishes to achieve low regret with respect to the best parameter setting in hindsight. While the highly discontinuous nature of the cost functions make this impossible in the worst case due to well-known lower bounds, this work identifies a new notion called dispersion that enables positive results. They also show how these same ideas can address problems of preserving privacy, which can be a concern if algorithm parameters used on one input are set based on possibly sensitive information from previously seen inputs. **□**

References

1. Balcan, M.-F., Dick, T., Sandholm, T. and Vitercik, E. Learning to branch. In *Proceedings of the 35th Intern. Conf. Machine Learning*, (Stockholm, Sweden, July 10–15, 2018), 353–362.
2. Balcan, M.-F., Dick, T., and Vitercik, E. Dispersion for data-driven algorithm design, online learning, and private optimization. In *Proceedings of the 59th IEEE Annual Symp. Foundations of Computer Science*, (Paris, France, Oct. 7–9, 2018), 603–614.
3. Balcan, M.-F., Dick, T. and White, C. Data-driven clustering via parameterized Lloyd's families. In *Proceedings of the Annual Conf. on Neural Information Processing Systems* (Montreal, Canada, Dec. 3–9, 2018), 10664–10674.
4. Balcan, M.-F., Nagarajan, V., Vitercik, E. and White, C. Learning theoretic foundations of algorithm configuration for combinatorial partitioning problems. In *Proceedings of the 30th Conf. Learning Theory* (Amsterdam, The Netherlands, July 7–10, 2017), 213–274.
5. Hutter, F., Hoos, H.H., Leyton-Brown, K. and Stützle, T. ParamILS: An automatic algorithm configuration framework. *J. Artificial Intelligence Research* 36, (2009), 267–306.
6. Smith-Miles, K.A. Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Computing Surveys* 41, 1 (2009).

Avrim Blum is a professor and Chief Academic Officer at Toyota Technological Institute at Chicago, IL, USA.

Copyright held by author.

Data-Driven Algorithm Design

By Rishi Gupta and Tim Roughgarden

Abstract

The best algorithm for a computational problem generally depends on the “relevant inputs,” a concept that depends on the application domain and often defies formal articulation. Although there is a large literature on empirical approaches to selecting the best algorithm for a given application domain, there has been surprisingly little theoretical analysis of the problem.

We model the problem of identifying a good algorithm from data as a statistical learning problem. Our framework captures several state-of-the-art empirical and theoretical approaches to the problem, and our results identify conditions under which these approaches are guaranteed to perform well. We interpret our results in the contexts of learning greedy heuristics, instance feature-based algorithm selection, and parameter tuning in machine learning.

1. INTRODUCTION

Rigorously comparing algorithms is hard. Two different algorithms for a computational problem generally have incomparable performance: one algorithm is better on some inputs but worse on the others. How can a theory advocate one of the algorithms over the other? The simplest and most common solution in the theoretical analysis of algorithms is to summarize the performance of an algorithm using a single number, such as its worst-case performance or its average-case performance with respect to an input distribution. This approach effectively advocates using the algorithm with the best summarizing value (e.g., the smallest worst-case running time).

Solving a problem “in practice” generally means identifying an algorithm that works well for most or all instances of interest. When the “instances of interest” are easy to specify formally in advance—say, planar graphs, the traditional analysis approaches often give accurate performance predictions and identify useful algorithms. However, the instances of interest commonly possess domain-specific features that defy formal articulation. Solving a problem in practice can require designing an algorithm that is optimized for the specific application domain, even though the special structure of its instances is not well understood. Although there is a large literature, spanning numerous communities, on empirical approaches to data-driven algorithm design (e.g., Fink¹¹, Horvitz et al.¹⁴, Huang et al.¹⁵, Hutter et al.¹⁶, Kotthoff et al.¹⁸, Leyton-Brown et al.²⁰), there has been surprisingly little theoretical analysis of the problem. One possible explanation is that worst-case analysis, which is the dominant algorithm analysis paradigm in theoretical computer science, is intentionally application agnostic.

This paper demonstrates that application-specific algorithm selection can be usefully modeled as a

statistical learning problem, in the spirit of Haussler.¹³ We prove that many classes of algorithms have small pseudo-dimension; in these cases, an approximately best-in-class algorithm can be learned from a modest number of representative benchmark instances. We interpret our results in the contexts of learning greedy heuristics, instance feature-based algorithm selection, and parameter tuning in machine learning.

Dimension notions from statistical learning theory have been used almost exclusively to quantify the complexity of classes of prediction functions (e.g. Anthony and Bartlett², Haussler¹³). Our results demonstrate that these concepts are useful and relevant in a much broader algorithmic context while also providing a novel formalization of the oft-mentioned but rarely defined “simplicity” of a family of algorithms.

2. MOTIVATING SCENARIOS

Our learning framework sheds light on several well-known approaches, spanning disparate application domains, to the problem of learning a good algorithm from data. To motivate and provide interpretations of our results, we describe several of these in detail. There are also strong connections between data-driven algorithm design and *self-improving algorithms*,^{1,9} where the goal is to design an algorithm that uses a modest amount of space and, given a sequence of independent samples from an unknown input distribution, converges to the optimal algorithm for that distribution; the full version¹² elaborates on these connections.

2.1. Example 1: greedy heuristic selection

One of the most common and also most challenging motivations for data-driven algorithm design is presented by computationally difficult optimization problems. When the available computing resources are inadequate to solve such a problem exactly, heuristic algorithms must be used. For most hard problems, our understanding of when different heuristics work well remains primitive. For concreteness, we describe one recent and high-stakes example of this issue, which also aligns well with our model and results in Section 4.

In 2016–2017, the FCC ran a novel double auction to buy back licenses for spectrum from certain television broadcasters and resell them to telecommunication companies for wireless broadband use.¹⁹ The auction generated roughly \$10 billion for the US government. The “reverse”

A preliminary version of this article was published in the *Proceedings of the 7th Annual Innovations in Theoretical Computer Science Conference*, January 2016. A full version with the title “A PAC Approach to Application-Specific Algorithm Selection” was published in *SIAM Journal on Computing*, June 2017.

(i.e., buy back) phase of the auction determined which stations to buy out and what to pay them. The auction was tasked with buying out sufficiently many stations, so that the remaining stations (who keep their licenses) can be “repacked” into a small number of channels, leaving a target number of channels free to be repurposed for wireless broadband. To first order, the feasible repackings were determined by interference constraints between stations. Computing a repacking, therefore, resembles familiar hard combinatorial problems such as the independent set and graph coloring problems.

The reverse auction deployed by the FCC used a greedy heuristic to decide which stations would keep their licenses and stay on the air. The chosen heuristic favored stations with high value and discriminated against stations that interfered with a large number of other stations. There are many ways of combining these two criteria and no obvious reason to favor one implementation over another. The specific implementation in the FCC auction was justified through trial-and-error experiments using synthetic instances that were thought to be representative. One interpretation of our results in Section 4 is as a post hoc justification of this approach to algorithm design for sufficiently simple collections of algorithms, such as the family of greedy heuristics considered for this FCC auction.

2.2. Example 2: parameter tuning in optimization and machine learning

Many “algorithms” used in practice are really meta-algorithms, with a large number of free parameters that need to be instantiated by the user. For instance, implementing even the most basic version of gradient descent requires choosing a step size and error tolerance. For a more extreme example, CPLEX, a widely used commercial linear and integer programming solver, comes with a 221-page parameter reference manual describing 135 parameters.

An analogous problem in machine learning is “hyperparameter optimization,” where the goal is to tune the parameters of a learning algorithm, so that it learns (from training data) a model with high accuracy on test data, and in particular a model that does not overfit the training data. A simple example is regularized regression, such as ridge regression, where a single parameter governs the trade-off between the accuracy of the learned model on training data and its “complexity.” More sophisticated learning algorithms can have many more parameters.

Figuring out the “right” parameter values is notoriously challenging in practice. The CPLEX manual simply advises that “you may need to experiment with them.” In machine learning, parameters are often set by discretizing and then applying exhaustive search (a.k.a. “grid search”), perhaps with random subsampling (“random search”).⁷ When this is computationally infeasible, variants of gradient descent are often used to explore the parameter space, with no guarantee of convergence to a global optimum.

The results in Section 6 can be interpreted as a sample complexity analysis of grid search for the problem of choosing the step size in gradient descent to minimize the

expected number of iterations needed for convergence. We view this as a first step toward reasoning more generally about the problem of learning good parameters for machine learning algorithms.

2.3. Example 3: empirical performance models for SAT solvers

The examples above already motivate choosing an algorithm for a problem based on characteristics of the application domain. A more ambitious and refined approach is to select an algorithm on a *per-instance* (instead of a per-domain) basis. Although it is impossible to memorize the best algorithm for every possible instance, one might hope to use coarse *features* of a problem instance as a guide to which algorithm is likely to work well.

For example, Xu et al.²⁴ applied this idea to the satisfiability (SAT) problem. Their algorithm portfolio consisted of seven state-of-the-art SAT solvers with incomparable and widely varying running times across different instances. The authors identified a number of instance features, ranging from simple features such as input size and clause/variable ratio to complex features such as Knuth’s estimate of the search tree size¹⁷ and the initial rate of progress of local search.^a The next step involved building an “empirical performance model” (EPM) for each of the seven algorithms in the portfolio—a mapping from instance feature vectors to running time predictions. They then computed their EPMs using labeled training data and a suitable regression model. With the EPMs in hand, it is clear how to perform per-instance algorithm selection: given an instance, compute its features, use the EPMs to predict the running time of each algorithm in the portfolio, and run the algorithm with the smallest predicted running time. Using these ideas (and several optimizations), their “SATzilla” algorithm won numerous medals at the 2007 SAT Competition. Section 5 outlines how to extend our learning framework to reason about EPMs and feature-based algorithm selection.

3. A PAC LEARNING MODEL

This section casts the problem of choosing the best algorithm for a poorly understood application domain as one of learning the optimal algorithm with respect to an unknown instance distribution. Section 3.1 formally defines the basic model, and Section 3.2 reviews the relevant preliminaries from statistical learning theory. Sections 4–6 describe the applications of the model to learning greedy heuristics, instance feature-based algorithm selection, and hyperparameter tuning.

3.1. The basic model

Our basic model consists of the following ingredients.

1. A fixed computational or optimization problem Π . For example, Π could be the problem of computing a maximum-weight independent set of a graph (Section 4) or minimizing a convex function (Section 6).

^a It is important, of course, that computing the features of an instance is an easier problem than solving it.

2. An unknown distribution \mathcal{D} over instances $x \in \Pi$.
3. A set \mathcal{A} of algorithms for Π . For example, \mathcal{A} could be a finite set of SAT solvers or an infinite family of greedy heuristics.
4. A performance measure $\text{COST}: \mathcal{A} \times \Pi \rightarrow [0, H]$, indicating the performance of a given algorithm on a given instance. Two common choices for COST are the running time of an algorithm and, for optimization problems, the objective function value of the solution produced by an algorithm.

The “application-specific” information is encoded by the unknown input distribution \mathcal{D} , and the corresponding “application-specific optimal algorithm” $A_{\mathcal{D}}$ is the algorithm that minimizes or maximizes (as appropriate)

$$\mathbf{E}_{x \sim \mathcal{D}}[\text{COST}(A, x)]$$

over the algorithms $A \in \mathcal{A}$. The *error* of an algorithm $A \in \mathcal{A}$ for a distribution \mathcal{D} is

$$|\mathbf{E}_{x \sim \mathcal{D}}[\text{COST}(A, x)] - \mathbf{E}_{x \sim \mathcal{D}}[\text{COST}(A_{\mathcal{D}}, x)]|.$$

In our basic model, the goal is:

Learn the application-specific optimal algorithm from data (i.e., samples from \mathcal{D}).

More precisely, the learning algorithm is given m i.i.d. samples $x_1, \dots, x_m \in \Pi$ from \mathcal{D} and (perhaps implicitly) the corresponding performance $\text{COST}(A, x_i)$ of each algorithm $A \in \mathcal{A}$ on each input x_i . The learning algorithm uses this information to suggest an algorithm $\hat{A} \in \mathcal{A}$ to use on future inputs drawn from \mathcal{D} . We seek learning algorithms that almost always output an algorithm of \mathcal{A} that performs almost as well as the optimal algorithm in \mathcal{A} for \mathcal{D} —learning algorithms that are probably approximately correct (PAC).

DEFINITION 3.1 A learning algorithm $L(\epsilon, \delta)$ -learns the optimal algorithm in \mathcal{A} from m samples if, for every distribution \mathcal{D} over Π , with probability at least $1 - \delta$ over m samples $x_1, \dots, x_m \sim \mathcal{D}$, L outputs an algorithm $\hat{A} \in \mathcal{A}$ with error at most ϵ .

3.2. Pseudo-dimension and uniform convergence

PAC learning an optimal algorithm, in the sense of Definition 3.1, reduces to bounding the “complexity” of the class \mathcal{A} of algorithms. We next review the relevant definitions from statistical learning theory.

Let \mathcal{H} denote a set of real-valued functions defined on the set X . A finite subset $S = \{x_1, \dots, x_m\}$ of X is (*pseudo-*)*shattered* by \mathcal{H} if there exist real-valued *witnesses* r_1, \dots, r_m such that, for each of the 2^m subsets T of S , there exists a function $h \in \mathcal{H}$ such that $h(x_i) > r_i$ if and only if $i \in T$ (for $i = 1, 2, \dots, m$). The *pseudo-dimension* of \mathcal{H} is the cardinality of the largest subset shattered by \mathcal{H} (or $+\infty$, if arbitrarily large finite subsets are shattered by \mathcal{H}). The pseudo-dimension is a natural extension of the VC dimension from binary-valued to real-valued functions.^b

^b The *fat shattering dimension* is another common extension of the VC dimension to real-valued functions. It is a weaker condition, in that the fat shattering dimension of \mathcal{H} is always at most the pseudo-dimension of \mathcal{H} , and it is sufficient for sample complexity bounds. Most of our arguments give the same upper bounds on both notions, so we present the stronger statements.

To bound the sample complexity of accurately estimating the expectation of all functions in \mathcal{H} , with respect to an arbitrary probability distribution \mathcal{D} on X , it is enough to bound the pseudo-dimension of \mathcal{H} .

THEOREM 3.2 (UNIFORM CONVERGENCE (e.g. Anthony and Bartlett²)) *Let \mathcal{H} be a class of functions with domain X and range in $[0, H]$, and suppose \mathcal{H} has pseudo-dimension $d_{\mathcal{H}}$. For every distribution \mathcal{D} over X , every $\epsilon > 0$, and every $\delta \in (0, 1]$, if*

$$m \geq c \left(\frac{H}{\epsilon} \right)^2 \left(d_{\mathcal{H}} + \ln \left(\frac{1}{\delta} \right) \right) \quad (1)$$

for a suitable constant c (independent of all other parameters), then with probability at least $1 - \delta$ over m samples $x_1, \dots, x_m \sim \mathcal{D}$,

$$\left| \left(\frac{1}{m} \sum_{i=1}^m h(x_i) \right) - \mathbf{E}_{x \sim \mathcal{D}}[h(x)] \right| < \epsilon$$

for every $h \in \mathcal{H}$.

We can identify each algorithm $A \in \mathcal{A}$ with the real-valued function $x \mapsto \text{COST}(A, x)$. Regarding the class \mathcal{A} of algorithms as a set of real-valued functions defined on Π , we can discuss its pseudo-dimension, as defined above. We need one more definition before we can apply our machinery to learn algorithms from \mathcal{A} .

DEFINITION 3.3 (EMPIRICAL RISK MINIMIZATION (ERM)) Fix an optimization problem Π , a performance measure COST , and a set of algorithms \mathcal{A} . An algorithm L is an *ERM algorithm* if, given any finite subset S of Π , L returns an algorithm from \mathcal{A} with the best average performance on S .

For example, for any Π , COST , and finite \mathcal{A} , there is the trivial ERM algorithm that simply computes the average performance of each algorithm on S by brute force and returns the best one. The next corollary follows easily from Definition 3.1, Theorem 3.2, and Definition 3.3.

COROLLARY 3.4 *Fix parameters $\epsilon > 0$, $\delta \in (0, 1]$, a set of problem instances Π , and a performance measure COST . Let \mathcal{A} be a set of algorithms that has pseudo-dimension d with respect to Π and COST . Then, any ERM algorithm $(2\epsilon, \delta)$ learns the optimal algorithm in \mathcal{A} from m samples, where m is defined as in (1).*

The same reasoning applies to learning algorithms beyond ERM. For example, with the same assumptions as in Corollary 3.4, an algorithm from \mathcal{A} with approximately optimal (over \mathcal{A}) average performance on a set of samples is also approximately optimal with respect to the true input distribution. This observation is particularly useful when the ERM problem is computationally difficult but can be approximated efficiently.

Corollary 3.4 is only interesting if interesting classes of algorithms \mathcal{A} have small pseudo-dimension. In the simple case where \mathcal{A} is finite, as in our example of an algorithm portfolio for SAT (Section 2.3), the pseudo-dimension of \mathcal{A} is

trivially at most $\log_2 |\mathcal{A}|$. The following sections demonstrate the much less obvious fact that natural infinite classes of algorithms also have small pseudo-dimension.

4. LEARNING GREEDY HEURISTICS

The goal of this section is to bound the pseudo-dimension of many classes of greedy heuristics such as, as a special case, the family of heuristics relevant for the FCC double auction described in Section 2.1. Throughout this section, the performance measure cost is the objective function value of the solution produced by a heuristic on an instance, where we assume without loss of generality a maximization objective.

4.1. Definitions and examples

Our general definitions are motivated by greedy heuristics for NP-hard problems. For example:

1. *Knapsack*. The input is n items with values v_1, \dots, v_n , sizes s_1, \dots, s_n , and a knapsack capacity C . The goal is to compute a subset $S \subseteq \{1, 2, \dots, n\}$ with maximum total value $\sum_{i \in S} v_i$, subject to having total size $\sum_{i \in S} s_i$ at most C . Two natural greedy heuristics are to greedily pack items (subject to feasibility) in order of nonincreasing value v_i or in order of nonincreasing density v_i/s_i (or to take the better of the two).
2. *Maximum-Weight Independent Set (MWIS)*. The input is an undirected graph $G = (V, E)$ and a nonnegative weight w_v for each vertex $v \in V$. The goal is to compute an independent set—a subset of mutually nonadjacent vertices—with maximum total weight. Two natural greedy heuristics are to greedily choose vertices (subject to feasibility) in order of nonincreasing weight w_v or nonincreasing density $w_v/(1 + \deg(v))$. (The intuition for the denominator is that choosing v “uses up” $1 + \deg(v)$ vertices— v and all of its now-blocked neighbors.) The latter heuristic also has an adaptive variant, where the degree $\deg(v)$ is computed in the subgraph induced by the vertices not yet blocked from consideration, rather than in the original graph.

In general, we consider *object assignment problems*, where the input is a set of n objects with various attributes, and the feasible solutions consist of assignments of the objects to a finite set R , subject to feasibility constraints. The attributes of an object are represented as an element ξ of an abstract set. For example, in the Knapsack problem, ξ encodes the value and size of an object; in the MWIS problem, ξ encodes the weight and (original or residual) degree of a vertex. In the Knapsack and MWIS problems, $R = \{0, 1\}$, indicating whether or not a given object is selected.

By a *greedy heuristic*, we mean algorithms of the following form (cf., the “priority algorithms” of Borodin et al.⁸):

1. While there remain unassigned objects:
 - (a) Use a *scoring rule* σ (described below) to compute a score $\sigma(\xi_i)$ for each unassigned object i , as a function of its current attributes ξ_i .
 - (b) For the unassigned object i with the highest score,

use an *assignment rule* to assign i a value from R and, if necessary, update the attributes of the other unassigned objects.⁹ For concreteness, assume that ties are always resolved lexicographically.

A *scoring rule* assigns a real number to an object as a function of its attributes. Assignment rules that do not modify objects’ attributes yield nonadaptive greedy heuristics, which use only the original attributes of each object (such as v_i or v_i/s_i in the Knapsack problem, for instance). In this case, objects’ scores can be computed in advance of the main loop of the greedy heuristic. Assignment rules that modify object attributes yield adaptive greedy heuristics, such as the adaptive MWIS heuristic described above.

In a *single-parameter* family of scoring rules, there is a scoring rule of the form $\sigma(\rho, \xi)$ for each parameter value ρ in some interval $I \subseteq \mathbb{R}$. Moreover, σ is assumed to be continuous in ρ for each fixed value of ξ . Natural examples include Knapsack scoring rules of the form v_i/s_i^ρ and MWIS scoring rules of the form $w_v/(1 + \deg(v))^\rho$ for $\rho \in [0, 1]$ or $\rho \in [0, \infty)$. A single-parameter family of scoring rules is κ -crossing if, for each distinct pair of attributes ξ, ξ' , there are at most κ values of ρ for which $\sigma(\rho, \xi) = \sigma(\rho, \xi')$. For example, all of the scoring rules mentioned above are 1-crossing rules.

For an example assignment rule, in the Knapsack and MWIS problems, the rule simply assigns i to “1” if it is feasible to do so, and to “0” otherwise. In the adaptive greedy heuristic for the MWIS problem, whenever the assignment rule assigns “1” to a vertex v , it updates the residual degrees of other unassigned vertices (two hops away) accordingly.

We call an assignment rule β -bounded if every object i is guaranteed to take on at most β distinct attribute values. For example, an assignment rule that never modifies an object’s attribute is 1-bounded. The assignment rule in the adaptive MWIS algorithm is n -bounded, where n is the number of vertices, as it only modifies the degree of a vertex (which lies in $\{0, 1, 2, \dots, n-1\}$).

Coupling a single-parameter family of κ -crossing scoring rules with a fixed β -bounded assignment rule yields a (κ, β) -single-parameter family of greedy heuristics. All of our running examples of greedy heuristics are (1,1)-single-parameter families, except for the adaptive MWIS heuristic, which is a (1, n)-single-parameter family.

4.2. Upper bound on pseudo-dimension

We next show that every (κ, β) -single-parameter family of greedy heuristics has small pseudo-dimension. This result applies to all of the concrete examples mentioned above; additional examples are described in the full version.¹²

THEOREM 4.1 (PSEUDO-DIMENSION OF GREEDY ALGORITHMS)
If \mathcal{A} is a (κ, β) -single-parameter family of greedy heuristics for an object assignment problem with n objects, then the pseudo-dimension of \mathcal{A} is $O(\log(\kappa\beta n))$.

⁹ We assume that there is always at least one choice of assignment that respects the feasibility constraints; this holds for all of our motivating examples.

In particular, all of our running examples are classes of heuristics with pseudo-dimension $O(\log n)$.

PROOF. Recall from the definitions (Section 3.2) that we need to upper bound the size of every set that is shatterable using the greedy heuristics in \mathcal{A} . For us, a set is a fixed set of s inputs (each with n objects) $S = \{x_1, \dots, x_s\}$. For a potential witness $r_1, \dots, r_s \in \mathbb{R}$, every algorithm $A \in \mathcal{A}$ induces a binary labeling of each sample x_i , according to whether $\text{COST}(A, x_i)$ is strictly more than or at most r_i . We proceed to bound from above the number of distinct binary labelings of S induced by the algorithms of \mathcal{A} , for any potential witness.

Consider ranging over algorithms $A \in \mathcal{A}$ —equivalently, over parameter values $\rho \in I$. The trajectory of a greedy heuristic $A \in \mathcal{A}$ is uniquely determined by the outcome of the comparisons between the current scores of the unassigned objects in each iteration of the algorithm. Because the family uses a κ -crossing scoring rule, for every pair i, j of distinct objects and possible attributes ξ_i, ξ_j with $\xi_i \neq \xi_j$, there are at most κ values of ρ for which there is a tie between the score of i (with attributes ξ_i) and that of j (with attributes ξ_j). Because σ is continuous in ρ for every fixed ξ , the relative order of the score of i (with ξ_i) and j (with ξ_j) remains the same in the open interval between two successive values of ρ at which their scores are tied. The upshot is that we can partition I into at most $\kappa + 1$ intervals, such that the outcome of the comparison between i (with attributes ξ_i) and j (with attributes ξ_j) is constant on each interval. In the case that $\xi_i = \xi_j$, because we break ties between equal scores lexicographically, the outcome of the comparison between $\sigma(\rho, \xi_i)$ and $\sigma(\rho, \xi_j)$ is the same for all $\rho \in I$.

Next, the s instances of S contain a total of sn objects. Each of these objects has some initial attributes. Because the assignment rule is β -bounded, there are at most $sn\beta$ object-attribute pairs (i, ξ_i) that could possibly arise in the execution of any algorithm from \mathcal{A} on any instance of S . This implies that, ranging across all algorithms of \mathcal{A} on all inputs in S , comparisons are only ever made between at most $(sn\beta)^2$ pairs of object-attribute pairs (i.e., between an object i with current attributes ξ_i and an object j with current attributes ξ_j). We call these the *relevant comparisons*.

For each relevant comparison, we can partition I into at most $\kappa + 1$ subintervals such that the comparison outcome is constant (in ρ) in each subinterval. Intersecting the partitions of all of the at most $(sn\beta)^2$ relevant comparisons splits I into at most $(sn\beta)^2\kappa + 1$ subintervals such that every relevant comparison is constant in each subinterval. That is, all of the algorithms of \mathcal{A} that correspond to the parameter values ρ in such a subinterval execute identically on every input in S . The number of binary labelings of S induced by algorithms of \mathcal{A} is trivially at most the number of such subintervals. Our upper bound $(sn\beta)^2\kappa + 1$ on the number of subintervals exceeds 2^s , the requisite number of labelings to shatter S , only if $s = O(\log(\kappa\beta n))$. \square

Theorem 4.1 and Corollary 3.4 imply that, if κ and β are bounded above by a polynomial in n , then an ERM algorithm (ϵ, δ) -learns the optimal algorithm in \mathcal{A} from only $m = \tilde{O}(\frac{H^2}{\epsilon^2})$

samples,^d where H is the largest objective function value of a feasible solution output by an algorithm of \mathcal{A} on an instance of Π .^e

Not all classes of algorithms have such a small pseudo-dimension. Theorem 4.1 thus gives one quantifiable sense in which natural greedy algorithms are “simple algorithms.”

REMARK 4.2 (EXTENSIONS) Theorem 4.1 is robust, and its proof is easily modified to accommodate various extensions. For example, consider families of greedy heuristics parameterized by d real-valued parameters ρ_1, \dots, ρ_d . Here, an analog of Theorem 4.1 holds with the crossing number κ replaced by a more complicated parameter—essentially, the number of connected components of the co-zero set of the difference of two scoring functions (with ξ, ξ' fixed and variables ρ_1, \dots, ρ_d). This number can often be bounded (by a function exponential in d) in natural cases, for example, using Bézout’s theorem.

REMARK 4.3 (NONLIPSCHITZNESS) We noted in Section 3.2 that the pseudo-dimension of a finite set \mathcal{A} is always at most $\log_2 |\mathcal{A}|$. This suggests a simple discretization approach to learning the best algorithm from \mathcal{A} : take a finite “ ϵ -net” of \mathcal{A} and learn the best algorithm in the finite net. (Indeed, Section 6 uses precisely this approach.) The issue is that without some kind of Lipschitz condition—stating that “nearby” algorithms in \mathcal{A} have approximately the same performance on all instances—there is no reason to believe that the best algorithm in the net is almost as good as the best algorithm from all of \mathcal{A} . Two different greedy heuristics—say, two MWIS greedy algorithms with arbitrarily close ρ -values—can have completely different executions on an instance. This lack of a Lipschitz property explains why we take care in Theorem 4.1 to bound the pseudo-dimension of the full infinite set of greedy heuristics.

4.3. Computational considerations

The proof of Theorem 4.1 also demonstrates the presence of an efficient ERM algorithm: the $O((sn\beta)^2)$ relevant comparisons are easy to identify, the corresponding subintervals induced by each are easy to compute (under mild assumptions on the scoring rule), and brute-force search can be used to pick the best of the resulting $O((sn\beta)^2\kappa)$ algorithms (with one arbitrary representative from each subinterval). This algorithm runs in polynomial time as long as β and κ are polynomial in n , and every algorithm of \mathcal{A} runs in polynomial time.

For example, for the family of Knapsack scoring rules described above, implementing this ERM algorithm reduces to comparing the outputs of $O(n^2m^2)$ different greedy heuristics (on each of the m sampled inputs), with $m = O(\log n)$. For the adaptive MWIS heuristics, where $\beta = n$, it is enough to compare the sample performance of $O(n^4m^2)$ different greedy algorithms, with $m = O(\log n)$.

^d The notation $\tilde{O}(\cdot)$ suppresses logarithmic factors.

^e Alternatively, the dependence of m on H can be removed if learning error ϵH (rather than ϵ) can be tolerated—for example, if the optimal objective function value is expected to be proportional to H anyways.

5. FEATURE-BASED ALGORITHM SELECTION

The previous section studied the problem of choosing a single algorithm for use in an application domain—of using training data to make an informed commitment to a single algorithm from a class \mathcal{A} , which is then used for all future instances. A more refined and ambitious approach is to select an algorithm based on both the previous experience *and the current instance to be solved*. This approach assumes, as in the scenario in Section 2.3, that it is feasible to quickly compute some features of an instance and then to select an algorithm as a function of these features.

Throughout this section, we augment the basic model of Section 3.1 with:

5. A set \mathcal{F} of possible instance feature values and a map $f: X \rightarrow \mathcal{F}$ that computes the features of a given instance.^f

For example, if X is the set of SAT instances, then $f(x)$ might encode the clause/variable ratio of the instance x , Knuth's estimate of the search tree size,¹⁷ etc.

We focus on the case where \mathcal{A} is small enough that it is feasible to learn a separate performance prediction model for each algorithm $A \in \mathcal{A}$. (The full version also discusses the case of large \mathcal{A} .¹²) This is exactly the approach taken in the motivating example of empirical performance models (EPMs) for SAT described in Section 2.3. We then augment the basic model to include a family of performance predictors.

6. A set \mathcal{P} of *performance predictors*, with each $p \in \mathcal{P}$ a function from \mathcal{F} to \mathbb{R} .

The goal is to learn, for each algorithm $A \in \mathcal{A}$, among all permitted predictors $p \in \mathcal{P}$, the one that minimizes some loss function. Like the performance measure COST , we take this loss function as given. For example, for the squared error loss function, for each $A \in \mathcal{A}$, we aim to compute the function that minimizes

$$\mathbb{E}_{x \sim \mathcal{D}}[(\text{COST}(A, x) - p_A(f(x)))^2]$$

over $p_A \in \mathcal{P}$. For a fixed algorithm A , this is a standard regression problem, with domain \mathcal{F} , real-valued labels, and a distribution on $\mathcal{F} \times \mathbb{R}$ induced by \mathcal{D} via $x \mapsto (f(x), \text{COST}(A, x))$. Bounding the sample complexity of this learning problem reduces to bounding the pseudo-dimension of \mathcal{P} . For standard choices of \mathcal{P} , such bounds are well known. For example, suppose the set \mathcal{P} is the class of *linear predictors*, with each $p \in \mathcal{P}$ having the form $p(f(x)) = a^T f(x)$ for some coefficient vector $a \in \mathbb{R}^d$. (The EPMs used by Xu et al.²⁴ are linear predictors.) The pseudo-dimension of \mathcal{P} is well known to be d . If all functions in \mathcal{P} map all possible inputs to $[0, H]$, then Corollary 3.4 implies a sample complexity bound of $\tilde{O}(\frac{H^4}{\epsilon^2} d)$ for (ϵ, δ) learning the predictor with minimum expected square error.

^f Defining a good feature set is a notoriously challenging and important problem, but it is beyond the scope of our model—we take the set \mathcal{F} and map f as given.

6. CHOOSING THE STEP SIZE IN GRADIENT DESCENT

For our last example, we give sample complexity results for the problem of choosing the best step size in gradient descent. When gradient descent is used in practice, the step size is generally taken much larger than the upper limits suggested by theoretical guarantees and often converges in many fewer iterations than with the step size suggested by theory. This motivates the problem of learning the best step size from examples. We view this as a first step toward reasoning more generally about the problem of learning good hyperparameters for machine learning algorithms.

6.1. Gradient descent preliminaries

Recall the basic gradient descent algorithm for minimizing a function f given an initial point z_0 over \mathbb{R}^n :

1. Initialize $z := z_0$.
2. While $\|\nabla f(z)\|_2 > \nu$:
 - (a) $z := z - \rho \cdot \nabla f(z)$.

We take the error tolerance ν as given and focus on the more interesting parameter, the step size ρ . Bigger values of ρ have the potential to make more progress in each step but run the risk of overshooting a minimum of f .

We instantiate the basic model (Section 3.1) to study the problem of learning the best step size. There is an unknown distribution \mathcal{D} over instances, where an instance $x \in \Pi$ consists of a function f and an initial point z_0 . Each algorithm A_ρ of \mathcal{A} is the basic gradient descent algorithm above, with some choice ρ of a step size drawn from some fixed interval $[\rho_\rho, \rho_u] \subset (0, \infty)$. The performance measure $\text{COST}(A, x)$ is the number of iterations (i.e., steps) taken by the algorithm for the instance x . Because gradient descent is translation invariant, we can assume for the sake of analysis that 0 is a minimum of f , with $f(0) = 0$. (We consider only instances that have a global minimum.)

To obtain positive results, we impose three further assumptions on problem instances, analogous to those used in the classical convergence analysis of gradient descent for smooth and strongly convex functions.

- (A1) Every function f is L -smooth for a known L , meaning that f is everywhere differentiable with $\|\nabla f(z_1) - \nabla f(z_2)\| \leq L \|z_1 - z_2\|$ for all z_1 and z_2 . (Every norm in this section is the ℓ_2 norm.)
- (A2) Every function f is m -strongly convex for a known $m \leq L$, meaning that it is continuously differentiable with $f(z_2) \geq f(z_1) + \nabla f(z_1)^T (z_2 - z_1) + \frac{m}{2} \|z_2 - z_1\|^2$ for all z_1 and z_2 .
- (A3) The magnitudes of the initial points are bounded, with $\|z_0\| \leq Z$ for some known constant $Z > \nu$.

These assumptions imply a “guaranteed progress toward 0” property: There is a known constant $\gamma \in (0, 1)$ such that $\|z - \rho \nabla f(z)\| \leq (1 - \gamma)\|z\|$ for all $\rho \in [\rho_\rho, \rho_u]$. (The standard analysis of gradient descent implies that $\gamma \geq \rho m$ for every $\rho \leq 2/(m + L)$ over this class of functions.) Our assumptions imply that gradient descent halts within $H := \log_{1/(1-\gamma)}(\frac{LZ}{\nu})$ iterations.

6.2. Technical lemma

The heart of the analysis is the following technical lemma, which bounds how far two gradient descent paths with different step sizes can diverge when initialized at the same point. Define $g(z, \rho) := z - \rho \nabla f(z)$ as the result of taking a single gradient descent step from the point z with the step size ρ , and let $g^j(z, \rho)$ denotes the result of taking j gradient descent steps.

LEMMA 6.1 *With assumptions (A1)–(A3), for every starting point z , iteration number j , and step sizes $\rho \leq \eta$,*

$$\|g^j(z, \rho) - g^j(z, \eta)\| \leq (\eta - \rho) \frac{c_\rho^j LZ}{\gamma},$$

where $c_\rho = \max\{1, L\rho - 1\} \geq 1$ is a constant that is a function only of L and ρ .

The most important point is that the right-hand side of the inequality in Lemma 6.1 goes to 0 with the step size difference $\eta - \rho$. The proof of Lemma 6.1 can be found in the full version.¹²

Lemma 6.1 easily implies a Lipschitz-type condition on $\text{COST}(A_\rho, x)$ as a function of ρ .

LEMMA 6.2 *For every input x satisfying (A1)–(A3) and step sizes $\rho \leq \eta$ with $\eta - \rho \leq \frac{\gamma^v}{mLZ} c_\rho^{-H}$,*

$$|\text{COST}(A_\rho, x) - \text{COST}(A_\eta, x)| \leq \lceil \log_{1/(1-\gamma)}(1 + \frac{L}{m}) \rceil.$$

PROOF. Suppose $\text{COST}(A_\eta, x) \leq \text{COST}(A_\rho, x)$; the argument in the other case is similar. Let $j = \text{COST}(A_\eta, x)$, and recall that $j \leq H$. By the stopping condition of gradient descent, $\|\nabla g^j(z_\eta, \eta)\| \leq v$. By the m -strong convexity of f , $\|g^j(z_\eta, \eta)\| \leq \frac{v}{m}$. By Lemma 6.1 and the assumed gap between ρ and η ,

$$\|g^j(z_\eta, \rho)\| \leq \frac{v}{m} + \gamma \frac{v}{m} = (1 + \gamma) \frac{v}{m}.$$

Set $\tau = \lceil \log_{1/(1-\gamma)}(1 + \frac{L}{m}) \rceil$. By the guaranteed progress condition,

$$\|g^{j+\tau}(z_\eta, \rho)\| \leq \frac{v}{L}.$$

By L -smoothness, $\|\nabla g^{j+\tau}(z_\eta, \rho)\| \leq v$. This implies that gradient descent with step size ρ requires at most τ more iterations to converge than with step size η . \square

6.3. Learning the best step size

We can now apply the discretization approach suggested by Remark 4.3. Let $K = \frac{\gamma^v}{mLZ} c_\rho^{-H}$. Let N denotes the set of all integer multiples of K that lie in the interval $[\rho_\rho, \rho_u]$ of possible step sizes. Note that $|N| \leq \rho_u/K + 1$. We can now state the main result of this section:

THEOREM 6.3 (LEARNABILITY OF STEP SIZE) *There is a learning algorithm that $(\lceil \log_{1/(1-\gamma)}(1 + \frac{L}{m}) \rceil + \epsilon, \delta)$ learns the optimal algorithm in \mathcal{A} using $m = \tilde{O}(H^3/\epsilon^2)$ samples from \mathcal{D} .[§]*

[§] We use the $\tilde{O}(\cdot)$ notation to suppress logarithmic factors in $Z, L, m, \rho_u, \frac{1}{\gamma}$, and $\frac{1}{\epsilon}$.

PROOF. Because $\mathcal{A}_N = \{A_\rho : \rho \in N\}$ is a finite set, its pseudo-dimension is at most $\log_2 |N|$. Let L_N denotes the ERM algorithm for this set. Corollary 3.4 implies that $L_N(\epsilon, \delta)$ -learns the optimal algorithm in \mathcal{A}_N using $m = \tilde{O}(H^2 \log |N| / \epsilon^2)$ samples. Because $\log_2 |N| = \tilde{O}(H)$, m is $\tilde{O}(H^3/\epsilon^2)$.

Now, Lemma 6.2 implies that for every ρ , there is an $\eta \in N$ such that, for every input distribution \mathcal{D} , the difference in expected costs of A_η and A_ρ is at most $\lceil \log_{1/(1-\gamma)}(1 + \frac{L}{m}) \rceil$. Thus, $L_N(\lceil \log_{1/(1-\gamma)}(1 + \frac{L}{m}) \rceil + \epsilon, \delta)$ -learns the optimal algorithm in \mathcal{A} using $\tilde{O}(H^3/\epsilon^2)$ samples. \square

7. CONCLUSION

Empirical work on data-driven algorithm design has far outpaced theoretical analysis of the problem, and this paper takes an initial step toward redressing this imbalance. We formulated the problem as one of learning a best-in-class algorithm with respect to an unknown input distribution. Many state-of-the-art empirical approaches to the problem map naturally to our learning framework. This paper demonstrates that many well-studied classes of algorithms have small pseudo-dimension, and thus, it is possible to learn a near-optimal algorithm from a relatively modest number of representative benchmark instances.

Our work suggests numerous wide-open research directions worthy of further study. For example:

1. Which other classes of algorithms have small pseudo-dimension? There has been recent progress on this question for local search algorithms,¹² clustering and partitioning algorithms,⁴ auctions and mechanisms,^{5, 6, 21, 22} and mathematical programs.³
2. In the *online* version of data-driven algorithm design, instances to a problem arrive one by one. The goal is to choose an algorithm to use at each time step, before the instance at that time step arrives, so that the average performance of the chosen algorithms is close to that of the best fixed algorithm in hindsight. When does such an online learning algorithm exist? The full version of this paper¹² gives positive results for classes of greedy algorithms; these are generalized in Cohen-Addad and Kanade¹⁰ and further in Balcan et al.³
3. When is it possible to learn a near-optimal algorithm using only a polynomial amount of computation, ideally with a learning algorithm that is better than brute-force search? Alternatively, are there (conditional) lower bounds stating that brute-force search is necessary for learning? See Roughgarden and Wang²³ for progress on these questions for learning revenue-maximizing auctions.
4. Are there any nontrivial relationships between statistical learning measures of the complexity of an algorithm class and more traditional computational complexity measures?

Acknowledgments

This research was supported in part by NSF awards CCF-1215965 and CCF-1524062. \square

References

1. Ailon, N., Chazelle, B., Clarkson, K.L., Liu, D., Mulzer, W., Seshadhri, C. Self-improving algorithms. *SIAM J. Comput.* 2, 40 (2011) 350–375.
2. Anthony, M., Bartlett, P.L. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.
3. Balcan, M.-F., Dick, T., Vitercik, E. Dispersion for data-driven algorithm design, online learning, and private optimization. In *Proceedings of the 59th Annual Symposium on Foundations of Computer Science (FOCS)* (2018), 603–614.
4. Balcan, M.-F., Nagarajan, V., Vitercik, E., White, C. Learning-theoretic foundations of algorithm configuration for combinatorial partitioning problems. In *Proceedings of the 30th Conference on Learning Theory (COLT)* (2017), 213–274.
5. Balcan, M.-F., Sandholm, T., Vitercik, E. Sample complexity of automated mechanism design. In *Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NeurIPS)* (2016), 2083–2091.
6. Balcan, M.-F., Sandholm, T., Vitercik, E. A general theory of sample complexity for multi-item profit maximization. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC)* (2018), 73–174.
7. Bergstra, J., Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* 1, 13 (2012), 281–305.
8. Borodin, A., Nielsen, M.N., Rackoff, C. (Incremental) priority algorithms. *Algorithmica* 4, 37 (2003), 295–326.
9. Clarkson, K.L., Mulzer, W., Seshadhri, C. Self-improving algorithms for coordinatewise maxima and convex hulls. *SIAM J. Comput.* 2, 43 (2014), 617–653.
10. Cohen-Addad, V., Kanade, V. Online optimization of smoothed piecewise constant functions. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)* (2017), 412–420.
11. Fink, E. How to solve it automatically: Selection among problem solving methods. In *Proceedings of the International Conference on Artificial Intelligence Planning Systems* (1998), 128–136.
12. Gupta, R., Roughgarden, T. A PAC approach to application-specific algorithm selection. *SIAM J. Comput.* 3, 46 (2017), 992–1017.
13. Haussler, D. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inform. Comput.* 1, 100 (1992), 78–150.
14. Horvitz, E., Ruan, Y., Gomes, C.P., Kautz, H.A., Selman, B., Chickering, D.M. A Bayesian approach to tackling hard computational problems. In *Proceedings of the Conference in Uncertainty in Artificial Intelligence (UAI)* (2001), 235–244.
15. Huang, L., Jia, J., Yu, B., Chun, B., Maniatis, P., Naik, M. Predicting execution time of computer programs using sparse polynomial regression. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)* (2010), 883–891.
16. Hutter, F., Xu, L., Hoos, H.H., Leyton-Brown, K. Algorithm runtime prediction: Methods & evaluation. *Artif. Intell.*, 206 (2014), 79–111.
17. Knuth, D.E. Estimating the efficiency of backtrack programs. *Math. Comput.*, 29 (1975), 121–136.
18. Kotthoff, L., Gent, I.P., Miguel, I. An evaluation of machine learning in algorithm selection for search problems. *AI Commun.* 3, 25 (2012), 257–270.
19. Leyton-Brown, K., Milgrom, P., Segal, I. Economics and computer science of a radio spectrum reallocation. *Proc. Natl. Acad. Sci.* 28, 114 (2017), 7202–7209.
20. Leyton-Brown, K., Nudelman, E., Shoham, Y. Empirical hardness models: Methodology and a case study on combinatorial auctions. *J. ACM* 4, 56 (2009).
21. Morgenstern, J., Roughgarden, T. On the pseudo-dimension of nearly optimal auctions. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NeurIPS)* (2015), 136–144.
22. Morgenstern, J., Roughgarden, T. Learning simple auctions. In *Proceedings of the 29th Conference on Learning Theory (COLT)* (2016), 1298–1318.
23. Roughgarden, T., Wang, J.R. Minimizing regret with multiple reserves. In *Proceedings of the 17th ACM Conference on Economics and Computation (EC)* (2016), 601–616.
24. Xu, L., Hutter, F., Hoos, H.H., Leyton-Brown, K. SATzilla: Portfolio-based algorithm selection for SAT. *J. Artif. Intell. Res.*, 32 (2008), 565–606.

Rishi Gupta (rishig@cs.stanford.edu), Department of Computer Science, Stanford University, Stanford, CA, USA.

Tim Roughgarden (tr@cs.columbia.edu), Department of Computer Science, Columbia University, New York, USA.

© 2020 ACM 0001-0782/20/6 \$15.00



上海科技大学
ShanghaiTech University

TENURE-TRACK AND TENURED POSITIONS
School of Information Science and Technology (SIST)

ShanghaiTech University invites highly qualified candidates to fill multiple tenure-track/tenured faculty positions as its core founding team in the School of Information Science and Technology (SIST). We seek candidates with exceptional academic records or demonstrated strong potentials in all cutting-edge research areas of information science and technology. They must be fluent in English. English-based overseas academic training or background is highly desired.

ShanghaiTech is founded as a world-class research university for training future generations of scientists, entrepreneurs, and technical leaders. Boasting a new modern campus in Zhangjiang Hightech Park of cosmopolitan Shanghai, ShanghaiTech shall trail-blaze a new education system in China. Besides establishing and maintaining a world-class research profile, faculty candidates are also expected to contribute substantially to both graduate and undergraduate educations.

Academic Disciplines: Candidates in all areas of information science and technology shall be considered. Our recruitment focus includes, but is not limited to: computer science and technology, electronic science and technology, information and communication engineering, applied mathematics and statistics, data science, robotics, bioinformatics, biomedical engineering, internet of things, smart energy, computer systems and security, operation research, mathematical optimization and other interdisciplinary fields involving information science and technology, especially areas related to AI.

Compensation and Benefits: Salary and startup funds are highly competitive, commensurate with experience and academic accomplishment. We also offer a comprehensive benefit package to employees and eligible dependents, including on-campus housing. All regular ShanghaiTech faculty members will join its new tenure-track system in accordance with international practice for progress evaluation and promotion.

Qualifications:

- Strong research productivity and demonstrated potentials;
- Ph.D. (Electrical Engineering, Computer Engineering, Computer Science, Statistics, Applied Math, or related field);
- A minimum relevant (including PhD) research experience of 4 years.

Applications: Submit (in English, PDF version) a cover letter, a 2-page research plan, a CV plus copies of 3 most significant publications, and names of three referees to: sist@shanghaitech.edu.cn

For more information, please visit: <http://sist.shanghaitech.edu.cn/>

Deadline: December 31, 2020



**ADVERTISING
IN CAREER
OPPORTUNITIES**

How to Submit a Classified Line Ad: Send an e-mail to acmm mediasales@acm.org. Please include text, and indicate the issue/or issues where the ad will appear, and a contact name and number.

Estimates: An insertion order will then be e-mailed back to you. The ad will be typeset according to CACM guidelines. NO PROOFS can be sent. Classified line ads are NOT commissionable.

Deadlines: 20th of the month/2 months prior to issue date. For latest deadline info, please contact: acmm mediasales@acm.org

Career Opportunities Online: Classified and recruitment display ads receive a free duplicate listing on our website at: <http://jobs.acm.org>

Ads are listed for a period of 30 days.

For More Information Contact:

**ACM Media Sales
at 212-626-0686 or
acmm mediasales@acm.org**

[CONTINUED FROM P. 96] come up with a standard like PostScript for 3D rendering, we could catalyze a whole industry—not just movie-making, but product design, architecture, and so on.

CATMULL: The intent was to make it an industry standard. The concept of open software, where everybody can make changes, wasn't really around at that time. For us, it was like, "let's make it an open standard, and then anybody can implement the standard." Steve Jobs didn't object, because he understood that our real goal was to attract the best people.

To say that Pixar attracted top talent seems like an understatement.

HANRAHAN: Pixar was the place. There was just such a concentration of great people. But to come back to Reyes for a moment, I think everybody up to that point had been concentrated on what it would take to do real-time graphics. Obviously, games had to be real-time, and if you're building a CAD program, you need to be able to interact with the model. What was amazing about Pixar is that people took this long view of the problem. There was this picture they had made of the road to Point Reyes [a national seashore and park preserve in Marin County, CA]. The idea, as I understood it, was that if we just went outside and looked around, we saw this unbelievable landscape. Suppose you wanted to make a movie that was located there; what would it take to do that? It was this amazing moonshot idea.

CATMULL: We picked a hard goal. In our case, it was the replication of reality. And now things look so realistic that I often can't tell in films, and I've got a very good eye.

Pixar's culture was also unique in giving artists and technologists equal footing.

CATMULL: From the beginning, we were careful to make sure that they looked at each other as peers. Each film had challenges, and the technical people loved the challenges. They couldn't do everything, but they did what they figured they could within the constraints. Then, in trying to solve those problems, they often came up with something that the artists hadn't thought of. It was a very positive back-and-forth, where the artists would look at something and say,

"We picked a hard goal. In our case, it was the replication of reality. And now things look so realistic that I often can't tell in films, and I have a pretty good eye."

"Oh, I'd like to put this in a movie," and ask for more. And the technical people would say, "Oh, that sounds exciting. I'll try to do that." This dynamic led to a stunning increase in the visual complexity of the films, while at the same time everybody was clear that the goal was to make a good movie and not let anything else get in the way.

HANRAHAN: A lot of technologists forget the context in which they're operating. I think we all had a great appreciation for art and the importance of movies in our culture. And that's why we didn't get stuck on one specific technical feature that magically solved all the problems. Because it's not just about creating photorealistic images, it's about telling stories and depicting things we care about.

CATMULL: It's about communicating. That's why ARPA [the Advanced Research Projects Agency of the U.S. Defense Department] funded graphics in the first place. I think the relationship between computer graphics and the rest of the computing industry is poorly understood. We were a small industry, but we published at SIGGRAPH, and then the engineers at companies like NVIDIA and AMD went to SIGGRAPH and used those ideas to keep advancing their chips, because the game industry is so big that they could afford to. Of course, those chips also went into boards in the high-end workstations that academia and entertainment were using, which meant *they* could continue to advance the state of the art with new algorithms and then publish them at SIGGRAPH. We had this virtuous cycle.

Eventually, GPUs became more powerful than CPUs.

CATMULL: Pat was one of the first to realize that this immense computing power could be used in other areas. The concept of neural networks had been around a long time, but there just wasn't enough processing power to do it. Suddenly, as a result of something happening in this cycle with the chip companies, the game industry, academia, entertainment, and SIGGRAPH, there was enough to power a whole new field.

HANRAHAN: Remember when Ed said we needed to do 80 million micro-polygons? We had a budget of 400 floating point operations per pixel when we were building RenderMan. We were always trying to get more compute cycles. When GPUs came along, and they literally had thousands of core machines instead of one core machine, I immediately realized that this was going to be the key to the future.

Ed, you've thought about how to create a culture that outlives you for a while. What's it like now that you're retired, and you're actually watching it happen?

CATMULL: I'd put my successors in place, and it was time, and it felt like the fulfillment of an arc. I fully expect that the people at Pixar and Disney Animation will continue to do things that I can't even foresee. That's the nature of creativity: you've got a glimmer of something, but you can't completely see it. And when you get surprised, that's when it's gratifying.

Pat, are you ready to retire?

HANRAHAN: Well, yes and no. When Salesforce acquired Tableau, I retired as Chief Scientist. My full-time job all along has been teaching and doing research at Stanford. If you want to maintain a good attitude in the world, just hang out with students. I teach systems programming to freshmen and sophomores these days, and they have such a positive attitude, are eager to learn, and work so hard. I love going into school and being surrounded by my students.

Leah Hoffmann is a technology writer based in Piermont, NY, USA.

© 2020 ACM 0001-0782/20/6 \$15.00

Q&A

Attaining The Third Dimension

ACMA.M. Turing Award recipients Ed Catmull and Pat Hanrahan discuss how they helped to bring the power of three-dimensional imagery to computer graphics.

THREE-DIMENSIONAL (3D) computer imagery is now central to the global film industry, not to mention games, design, and the emerging fields of virtual and augmented reality. But when ACM A.M. Turing Award recipients Ed Catmull and Pat Hanrahan began working together at Pixar, the goal of being able to produce truly realistic images on a computer seemed remote. Here, Ed and Pat discuss what it took to overcome the challenge—and how their work on Pixar’s pioneering image system, RenderMan, came about.

Pixar was formed in 1986, when Steve Jobs helped spin off the computer division from Lucasfilm, where Ed had been working since 1979. Pat, you joined shortly thereafter.

HANRAHAN: Within days, I think.

Your work on what became RenderMan traces back to the Lucasfilm days. Ed, can you talk about that?

CATMULL: George Lucas hired me to bring technology into the film industry, which included digital audio and video editing and, later, games. While we were working on modeling and animation, we had to solve a number of problems to do with complexity and motion blur. Rob Cook, Loren Carpenter, and I also set a goal of being able to create an image containing 80 million micropolygons. It was a ridiculously high goal—the state of the art for polygons at the time was about 40,000—but we wanted to force ourselves to think in a different way. In the end, a suggestion was made by an engineer named Rodney Stock. Rob Cook ran with it, and it



Ed Catmull and Pat Hanrahan

became stochastic sampling, which today is the way the entire industry works.

This led to the creation of Lucasfilm’s rendering software, Reyes, which was RenderMan’s predecessor and stood for “renders everything you ever saw.”

CATMULL: Around the same time as we launched Pixar, Silicon Graphics was becoming a significant player in making chips and computers, and we and several other companies decided to come up with an interface that would give us a common way of describing objects,

lighting, and shading. Pat was asked to be the architect because he’s so brilliant and because he understood languages. He took some of the concepts of shading and turned them into a shading language. He made all the final decisions on the software design, and once it was done, we renamed it RenderMan.

HANRAHAN: My thinking was motivated by a language called PostScript, which was the foundation of the original laser printer and which had revolutionized desktop publishing. We thought that if we could [CONTINUED ON P. 95]

Today's Research Driving Tomorrow's Technology

The ACM Digital Library (DL) is the most comprehensive research platform available for computing and information technology and includes the ongoing contributions of the field's most renowned researchers and practitioners.

Each year, roughly 20,000 newly published articles from ACM journals, magazines, technical newsletters and annual conference volumes are added to the DL's complete full text contents of more than 550,000 articles.

The DL also features the fully integrated and comprehensive bibliographic index, *The Guide to Computing Literature*—a continually updated index featuring millions of publication records from over 5,000 publishers worldwide.

For more information, please visit

<https://libraries.acm.org/>

or contact ACM at

dl-info@hq.acm.org

ACM

DL

DIGITAL
LIBRARY



SIGGRAPH THINK
2020 S2020.SIGGRAPH.ORG BEYOND

THINK BEYOND

[S2020.SIGGRAPH.ORG](https://s2020.siggraph.org)

SIGGRAPH 2020 offers inspiration, putting the latest in art and tech at your fingertips. Discover inspiring content that demonstrates the latest advancements in computer graphics and interactive techniques research, education, applications and entertainment.

The 47th International Conference & Exhibition
on Computer Graphics and Interactive Techniques



Sponsored by ACMSIGGRAPH