

# COMMUNICATIONS

CACM.ACM.ORG

OF THE

# ACM

02/2020 VOL.63 NO.02

## **Toward ML-Centric Cloud Platforms**

Tracking Shoppers

Professional Social  
Matching Systems

Fuzzing: Hack, Art, and Science

Association for  
Computing Machinery

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

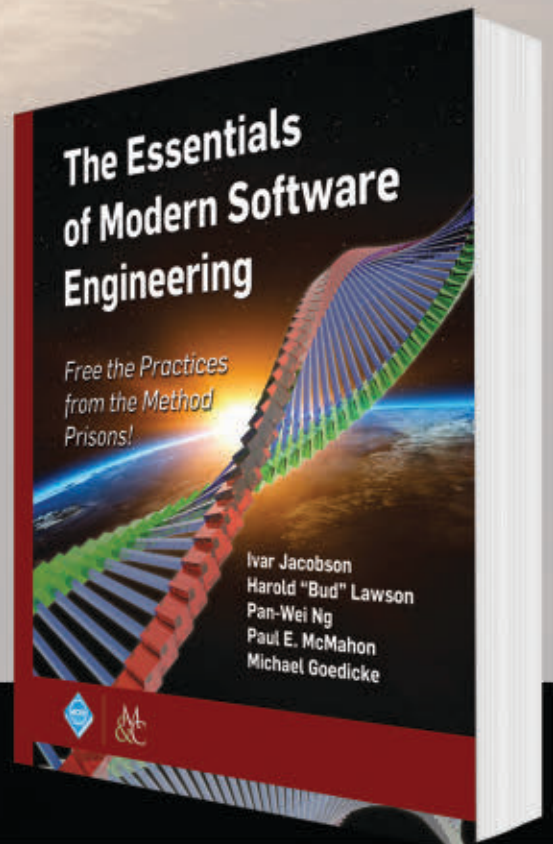
**Ivar Jacobson, Harold “Bud” Lawson,  
Pan-Wei Ng, Paul E. McMahon,  
Michael Goedicke**

ISBN: 978-1-947487-24-6

DOI: 10.1145/3277669

<http://books.acm.org>

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



**ACM BOOKS**  
Collection I

July 26–July 30, 2020 | Portland, Oregon

# PEARC20

PRACTICE AND EXPERIENCE IN ADVANCED RESEARCH COMPUTING

PEARC20 brings together community thought leaders, CI professionals, and students to learn, share ideas, and craft the infrastructure of the future. This year's theme, "Catch the Wave," embodies the spirit of the community's drive to stay on pace and in front of the new waves in technology, analytics, and a globally connected and diverse workforce.

## Technical Tracks

- Advanced research computing environments - Systems and System Software
- Application software, support, and outcomes
- People enabling research computing - workforce development, diversity, and professionalization
- Trending now - Machine learning and artificial intelligence

Join us at PEARC20 as we discuss the community's challenges, opportunities, and solutions in research computing.

Learn more at: [pearc.acm.org/pearc20](https://pearc.acm.org/pearc20)

## Become an Exhibitor

PEARC20 provides a global forum for students and professionals to discuss challenges, opportunities, and solutions in the fields of advanced research and data-intensive computing. Become a PEARC20 exhibitor to get in front of this audience.

Learn more by visiting:  
[pearc.acm.org/pearc20/exhibitors/](https://pearc.acm.org/pearc20/exhibitors/)



Association for  
Computing Machinery

*Advancing Computing as a Science & Profession*

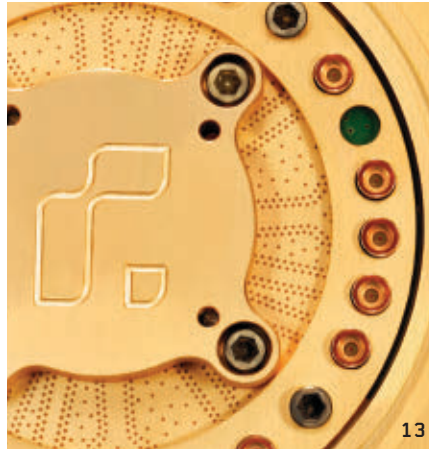
## Departments

- 5 **From the President**  
**Start Small, Then Achieve Big Impact**  
*By Cherri M. Pancake*
- 
- 7 **Cerf's Up**  
**On Durability**  
*By Vinton G. Cerf*
- 
- 10 **BLOG@CACM**  
**Sizing the U.S. Student Cohort for Computer Science**  
Mark Guzdial considers indicators that U.S. K–12 students will study computer science.
- 
- 33 **Calendar**
- 
- 91 **Careers**

## Last Byte

- 96 **Q&A**  
**'Everything Fails All the Time'**  
Werner Vogels, an expert on ultra-scalable systems, talks about listening to customers, reconceptualizing the stack, and building a product-centered culture.  
*By Leah Hoffmann*

## News



- 13 **Learning to Trust Quantum Computers**  
They need to show us they can solve the biggest problems.  
*By Chris Edwards*
- 
- 16 **Dark Web's Doppelgängers Aim to Dupe Antifraud Systems**  
Digital doppelgängers that fool online payment fraud detection systems are a threat to your bank balance.  
*By Paul Marks*
- 
- 19 **Tracking Shoppers**  
Retailers of all stripes are using technology to follow consumers through their brick-and-mortar stores in order to develop detailed profiles of their shopping habits and preferences.  
*By Keith Kirkpatrick*

## Viewpoints

- 22 **Global Computing**  
**Are We Losing Momentum?**  
Estimating when the second half of the world will come online.  
*By Carlos Iglesias, Dhanaraj Thakur, and Michael L. Best*
- 
- 25 **Inside Risks**  
**Are You Sure Your Software Will Not Kill Anyone?**  
Using software to control potentially unsafe systems requires the use of new software and system engineering approaches.  
*By Nancy Leveson*
- 
- 29 **Kode Vicious**  
**Numbers Are for Computers, Strings Are for Humans**  
How and where software should translate data into a human-readable form.  
*By George V. Neville-Neil*
- 
- 31 **Viewpoint**  
**When Human-Computer Interaction Meets Community Citizen Science**  
Empowering communities through citizen science.  
*By Yen-Chia Hsu and Illah Nourbakhsh*
- 
- 35 **Viewpoint**  
**Guiding Students to Develop Essential Skills**  
Students should interact with one another to practice skills and construct their own understanding, with assistance from a teacher acting as a coach and guide—not a lecturer.  
*By Clif Kussmaul*



Practice



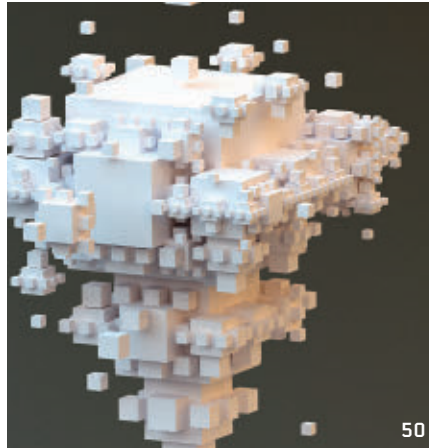
38

38 **Opening Up the Baseboard Management Controller**  
If the CPU is the brain of the board, the BMC is the brain stem.  
*By Jessie Frazelle*

41 **Optimizations in C++ Compilers**  
A practical journey.  
*By Matt Godbolt*

**Q** Articles' development led by **acmqueue**  
[queue.acm.org](https://queue.acm.org)

Contributed Articles



50

50 **Toward ML-Centric Cloud Platforms**  
Exploring the opportunities to use ML, the possible designs, and our experience with Microsoft Azure.  
*By Ricardo Bianchini, Marcus Fontoura, Eli Cortez, Anand Bonde, Alexandre Muzio, Ana-Maria Constantin, Thomas Moscibroda, Gabriel Magalhaes, Gırish Bablani, and Mark Russinovich*



Watch the authors discuss this work in the exclusive *Communications* video.  
<https://cacm.acm.org/videos/ml-centric-cloud-platforms>

Review Articles



70

60 **Directions for Professional Social Matching Systems**  
Future PSM systems will require diversity-enhancing yet contextually sensitive designs.  
*By Thomas Olsson, Jukka Huhtamäki, and Hannu Kärkkäinen*

70 **Fuzzing: Hack, Art, and Science**  
Reviewing software testing techniques for finding security vulnerabilities.  
*By Patrice Godefroid*



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

Research Highlights

80 **Technical Perspective**  
**Lighting the Way to Visual Privacy**  
*By Marco Gruteser*

81 **Automating Visual Privacy Protection Using a Smart LED**  
*By Shilin Zhu, Chi Zhang, and Xinyu Zhang*



**About the Cover:**  
This month's cover story traces an exercise in integrating machine learning technology into Microsoft's Azure cloud platform. This firsthand experience, as detailed by Microsoft researchers, explores future opportunities in and the technical challenges of building ML-centric cloud platforms. Cover artwork "Sky Ball" by Andi Mucke.





DOI:10.1145/3377932

Cherri M. Pancake

# Start Small, Then Achieve Big Impact

**A**CM HAS AN amazing cadre of volunteers around the globe, and they deserve the credit for what ACM accomplishes. It's not just that volunteers are the ones who come up with innovative ideas. They also invest their personal time to make those ideas come to life. ACM owes its success to volunteers' collective efforts to "start small, but with a big vision in mind." Let's look at some recent examples.

**Taking a stand on harassment.** It's hard to work in computing these days and not be aware that people who are "different" feel excluded from many technical conversations and opportunities. Studies have shown that's one reason young women and people from underrepresented groups don't choose computing as a career. ACM is not directly involved in hiring or educating the future workforce, but as a professional society we can and should speak out about the importance of encouraging representation from all backgrounds, perspectives, and ideas. This is reflected in the ACM Code of Ethics and Professional Conduct ([ethics.acm.org](http://ethics.acm.org)), particularly sections 1.4 and 1.5.

What I'd like to recognize here is how the work of many volunteers contributed to what has become an ACM-wide stance on what constitutes acceptable behavior. Although we have no authority over what

happens in the workplace or classroom, we *can* address behavior at all ACM events, from conferences to committee meetings to sponsored speaker events. ACM members noted that sometimes a speaker or attendee engages in behavior that is demeaning or offensive to other participants. Concerned volunteers began trying to codify what behavior is/isn't acceptable and what the consequences of poor behavior should be. What started as statements by specific conferences culminated in our formal Policy Against Harassment at ACM Activities (<http://bit.ly/2PXiz98>). This was followed by a Policy on Coercion and Abuse in the ACM Publications Process (<http://bit.ly/2PWYvby>). There is more work to be done, but these policies address the most common ACM contexts.

**Recognizing the need for reproducible computing.** Technology has become essential to everyday life, which means society collectively places a lot of trust in technology. Since software and data reliability is key to that trust, it must be possible to independently verify the results of algorithms and software. Recently, volunteers working with a couple of ACM journals and conferences began to encourage authors to include their data, code, intermediate results, and so on as part of the publication process. These efforts



gained momentum, with some conferences even awarding prizes for cases where independent teams succeeded in reproducing the results. The ACM Digital Library now has a series of "badges" reflecting what supporting materials are available and the extent to which they have been evaluated or replicated by others ([www.acm.org/publications/artifacts](http://www.acm.org/publications/artifacts)). The badges are indicators of skill and quality, recognizing the extra effort required to make computing processes more transparent.

**Expanding the reach of computing.** Some of the most exciting work happens at the intersection of computing with other disciplines, or in areas still emerging within mainstream computer science. Each year, ACM's Special Interest Groups and regional councils host summer schools where students and researchers explore new challenges and experiment with new techniques. The table shows some of the summer/winter schools around the globe in 2019. These events create opportunities to start building a community that may result in a future publication, conference, or SIG.

These are just some of the many ways small groups within ACM share ideas that end up taking on a life of their own. Special thanks to all of you who contributed to them. And I challenge other members to reach out to your colleagues and consider developing your own grassroots effort. What's small today could be a global initiative within just a few years. □

*Cherri M. Pancake*, ACM PRESIDENT

## A sampling of the summer/winter schools offered by ACM groups in 2019.

Topic	Location	ACM Group
Detection and Analysis of Malware	Pune, India	ACM India Council
HPC for AI and Dedicated Applications	Barcelona, Spain	ACM Europe Council
Data Science	Athens, Greece	SIGMOD+SIGIR+SIGKDD
Cybersecurity and Data Analytics	Delhi, India	ACM India Council
Projects Based on UN's Sustainability Goals	Prague, Czech Rep.	SIGEVO
Hybrid Cloud	Bangalore, India	ACM India Council
Machine Learning for Data Mining and Search	Cape Town, S. Africa	SIGIR+SIGKDD

# Inviting Young Scientists



HEIDELBERG  
LAUREATE  
FORUM



Association for  
Computing Machinery

## Meet Great Minds in Computer Science and Mathematics

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

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

The 8th Heidelberg Laureate Forum will take place **September 20–25, 2020** in Heidelberg, Germany.

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

### Who can participate?

Students of computer science and mathematics (or closely related fields) at the Undergraduate/Pre-Master, Graduate Ph.D. and Postdoc levels can apply for the Heidelberg Laureate Forum. Postdocs include young researchers in classical postdoc positions as well as those who have recently completed (within five years) their Ph.D. and are still strongly interested in scientific matters, even if currently working in a non-scientific environment.

### How to apply:

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

### What is the schedule?

The application deadline is **February 14, 2020**.

Successful applicants will be notified by **mid-late April 2020**.

*More information available on*  
<https://www.heidelberg-laureate-forum.org/young-researchers/faq.html>







Vinton G. Cerf

DOI:10.1145/3377424

# On Durability

I'm no economist, but I have become convinced that our consuming societies, at least in the economically "developed" world, have become sources of harm to

ourselves and our planet. One has only to read about single-use products (for example, plastic bags and bottles, packaging material, straws, paper cups, gift wrapping) or short-use products (for example, mobile phones and other electronics) to recognize this is a serious and huge problem. This consumption drives a significant part of the economy. It isn't just relatively modest per-item costs either. We probably don't use automobiles as long as we could (and should) before we want the latest and greatest. There *is* a category of goods called *durable goods* that typically have longer usage. Washing machines, refrigerators, ovens, stove tops, fireplaces, utensils, and dishes fall into this category. This line of reasoning makes me wonder whether we could shift the pollution needle toward longer use products as substitutes.

While it is not a solution to all short-use goods, software can contribute new functionality without requiring replacement of the underlying hardware. A good example of this is found with the Tesla electric cars. They are heavily dependent on software for their operation and upgrades with new functionality are frequent and unobtrusive. One wonders whether this concept could increase the useful lifetime of at least some products. Of course, the idea is not useful for nonprogrammable goods, like plastic bottles and paper towels!

However, it is worth thinking about the design of products to maximize ease of repair, durability, and perhaps

also repurposing. For example, imagine glass jars designed to be useful as cups as well as being reused as containers. Remember Mason jars for preserving fruits and vegetables for the winter? Or the glass bottles that soft drinks came in that were redeemed for two cents and reused by the bottling companies? What about reversible clothing? I remember saving animal fat and turning it into soap (OK, I suppose that would be a bit extreme today). None of these are new ideas but they make me wonder about increasing the value of durability and decreasing the attraction of throwaway products.

I guess that changing the value system is a form of social engineering and not so easy. One would look for incentives for consumers and producers to induce such a change. It's also important to understand how such a trend would affect the economy. Consumption drives production and expenditure. Would the "velocity of money" change significantly in a society that values durability? How might that affect economic indicators such as gross domestic product (GDP)? Would this change income and wealth inequality? Would it affect the way in which companies are valued? Profit growth as a metric of corporate value may induce behaviors that are not conducive to societal well-being if the side effects are pollution, waste, and externalizing costs (for example, shipping trash out of the country). I'm not qualified to

answer any of these questions, but they all seem to me worthy of asking.

Returning to the possible role of software and computing, one wonders whether these tools could be applied to the design of durable and reusable products? I am thinking not only of programmable goods but of other products that would benefit from flexible designs. It used to be the case that repair of products and inventory of spare parts added value to the products. Products today seem to be designed not for repair but replacement. Could we apply our machine learning and other software tools to make products more repairable rather than less? It is possible that modularity of design could play a role although it might increase the cost of production but perhaps reduce the lifetime cost of use. We have tried to make products recyclable but this seems to have been only partially effective. Perhaps a return to durability is worthy of consideration. Of course, I may have been influenced by the fact that I live in a house full of antiques, some of which are over 200 years old—not counting me. At some point, the value of used furniture goes from zero to ridiculously expensive. □

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

Copyright held by author.

# ACM ON A MISSION TO SOLVE TOMORROW.

Dear Colleague,

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

For over 70 years, ACM has helped computing professionals be their most creative, connect to peers, and see what's next, and inspired them to advance the profession and make a positive impact.

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

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

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

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

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

Sincerely,



Cherri M. Pancake  
President  
Association for Computing Machinery



Association for  
Computing Machinery

*Advancing Computing as a Science & Profession*

# SHAPE THE FUTURE OF COMPUTING. JOIN ACM TODAY.

[www.acm.org/join/CAPP](http://www.acm.org/join/CAPP)

## SELECT ONE MEMBERSHIP OPTION

### ACM PROFESSIONAL MEMBERSHIP:

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

### ACM STUDENT MEMBERSHIP:

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

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

## PAYMENT INFORMATION

Name

Mailing Address

City/State/Province

ZIP/Postal Code/Country

- Please do not release my postal address to third parties

Email Address

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

- AMEX  VISA/MasterCard  Check/money order

Credit Card #

Exp. Date

Signature

### Purposes of ACM

ACM is dedicated to:

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

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

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

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

# BE CREATIVE. STAY CONNECTED. KEEP INVENTING.



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

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

Fax: 212-944-1318  
[acmhelp@acm.org](mailto:acmhelp@acm.org)  
[www.acm.org/join/CAPP](http://www.acm.org/join/CAPP)

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:10.1145/3374764

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

## Sizing the U.S. Student Cohort for Computer Science

*Mark Guzdial considers indicators that U.S. K-12 students will study computer science.*



### Mark Guzdial The Size of Computing Education Today, By The Numbers

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

November 3, 2019

Alan Kay, Cathie Norris, Elliot Soloway, and I had an article in the September 2019 issue of *Communications* called “Computational Thinking Should Just Be Good Thinking” (access the article at <http://bit.ly/2P7RYEV>). Our argument is that “computational thinking” is already here — students use computing every day, and that computing is undoubtedly influencing their thinking. What we really care about is effective, critical, “expanded” thinking, where computing helps us think. To do that, we need better computing.

Ken Kahn engaged with our article in the comments section (thank you, Ken!), and he made a provocative comment:

There are have been many successful attempts to add program-

ming to games: Rocky’s Boots (1982), Robot Odyssey (1984), RoboSport (1991), Minecraft (multiple extensions), and probably many more. But these efforts excite a small fraction relative to those who are excited about using general-purpose programming systems such as Logo, Scratch, Squeak, ToonTalk, or Snap! for their own projects.

That got me wondering: Is general-purpose programming more popular than specialized programming? That’s a hard question to answer, but there are other related questions that are answerable. What fraction of people use Scratch and other general-purpose programming systems, versus something like Minecraft? What fraction of students learn any kind of programming at all? How many students learn general-purpose programming today, compared to using other computing environments or learning other science, technology, engineering and mathematics (STEM) subjects?

Here is what I found. I decided to be U.S.-centric, so that I could compare to known overall populations.

### Overall Populations

Number of people in the U.S. (World Population Review, <http://bit.ly/2Pg79Mu>): 329 million.

Number of school children in the U.S. (National Center for Educational Statistics, <http://bit.ly/2OKFhAK>): 56.6 million

► About 50.8 million in public schools, about 5.8 million in private schools. Only about 1.5 million children are homeschooled.

► 35.5 million are in pre-K to grade 8, 15.3 million in high school.

### Learners of Computing vs. Users of Computing Tools

Number of Scratch users in U.S. (Scratch community statistics at a glance, <http://bit.ly/37YWrCl>): 19.2 million.

We don’t know ages (and we know that we can’t trust the age data in Scratch; see “Youth Computational Participa-

tion in the Wild: Understanding Experience and Equity in Participating and Programming in the Online Scratch Community,” <http://bit.ly/2ONsq0E>). Roughly one-third of all U.S. schoolchildren have tried Scratch. While Logo, Snap!, Squeak, and other tools are popular, they are nowhere near as popular as Scratch. Scratch is likely our best possible scenario.

Number of Code.org students in U.S. (2018, Computing Education Research Blog, <http://bit.ly/2DJiz5W>): 1.2 million.

Number of Minecraft users (from “‘Minecraft’ has been quietly dominating for over 10 years, and now has 112 million players every month,” *Business Insider*, September 14, 2019, <http://bit.ly/2OISLNv>): 112 million.

Obviously, that’s more than the U.S., and not just school age, since that’s even school children or a third of everyone in the U.S. I was unable to find a number just in the U.S.

I found this estimate for the number of children playing Minecraft (in “Minecraft teaches kids about tech, but there’s a gender imbalance at play,” *The Conversation*, January 16, 2018, <http://bit.ly/2OISLNv>):

The results show that 53% of children aged 6 to 8, and 68% of children aged 9 to 12, are actively playing Minecraft. More than half of those play more than once per week.

Number of Fortnite users (from Fall Skirmish Fortnite Series, Fall Skirmish Details, September 20, 2018): 78.3 million.

Again, more than in the U.S., and not limited to school children.

The best estimate I could find is that 61% of U.S. teens play Fortnite (from “Exclusive: ‘Fortnite’ survey shows kids are playing in class. So what can parents do?” *USA Today*, December 6, 2018, <http://bit.ly/363N02V>), so perhaps 9.3 million.

**Bottom line:** There is no computing education tool that comes anywhere close to the number of children who play Minecraft or Fortnite.

## Learners of Computing vs. Other Learners of STEM

One way of getting a sense of access to computing vs. other subjects in the U.S. is to look at access to Advanced Placement, which is a program for accessing advanced subjects (for college credit or placement) in high school. It is generally considered elitist: not every student gets access to it (though there is some argument that it’s no longer elitist.) But if we’re going to compare between subjects, it’s elitist to all subjects equally.

Number of students who took AP Calculus AB (2018, <http://bit.ly/2sHB1cU>): 308,537

Number of students who took AP Calculus BC (2018, <http://bit.ly/2sHB1cU>): 139,376  
Total: 347,913

Compared to the 15.3-million high schools students in the U.S., this is about 2%. It’s an upper bound on the number of students that one might get in an AP STEM class.

Number of students who took AP Physics 1 (2018, <http://bit.ly/2LhaXMg>): 170,653

Number of students who took AP Physics 2 (2018, <http://bit.ly/2LgcvpC>): 24,985

Total: 195,638

Number of students who took AP Computer Science A (2018, <http://bit.ly/360EMbs>): 65,133

Number of students who took AP Computer Science Principles (2018, <http://bit.ly/2ONsq0E>): 72,187

So, combined, AP exams in computer science have about 137,320 exam takers, which is about 70% of AP Physics. That’s about 40% of AP Calculus, or about 40% of 2% of all U.S. high school students.

**The Bottom Line:** Computing education is not even close to the mainstream of STEM.

**Conclusion: We should try *everything*:** In general, we have not yet created popular computing education. We reach very few students.

To respond to Ken, we only have reached a small fraction of students with *all* our tools combined. We should be programming extensions to games *and* general-purpose programming languages *and* everything else we can think of that might help kids to gain access to computing education.

There’s so little reaching students now that it doesn’t make sense to build only on what’s been most successful in the past. All past successes may be local maxima. We may have to try something radically different to reach many students.

---

## Comments

*Interesting numbers. It would be great to get a worldwide census. I think the numbers are better in the U.K. for 5-to-16-year-olds, and not as good for 16 to-18-year-olds. I've heard about countries with much better numbers (Costa Rica, Lithuania, ??).*

*Twelve years ago, I wrote “Should Logo Keep Going Forward 1?” (<http://bit.ly/2Lz2gx1>) for EuroLogo 2007. I tried to argue against incremental progress; instead, we should move towards “something radically different” as you suggest. I came up with a list of 12 ideas that looking at it now should be a much longer list. One thing I would add today is artificial intelligence playing multiple roles including being a guide/coach and providing new kinds of computational building blocks for students to use in their projects.*

*One topic that Seymour Papert brought up when I spoke to him about this just before his accident is, whose aesthetics should count: mathematicians or engineers? A mathematically aesthetic design would be based upon a very small set of very expressive building blocks. Nearly everything we build for children has been engineered to have a large number of nice features, instead of a beautiful kernel.*

*Ken Kahn  
November 4, 2019*

---

**Mark Guzdial** is a professor in the Computer Science & Engineering Division of the University of Michigan, USA.

© 2020 ACM 0001-0782/20/2 \$15.00

# ACM Transactions on Quantum Computing (TQC)

Open for  
Submissions

**Publishes high-impact, original research papers and select surveys on topics in quantum computing and quantum information science**



Recent advances in quantum computing have moved this new field of study closer toward realization and provided new opportunities to apply the principles of computer science. A worldwide effort is leveraging prior art as well as new insights to address the critical science and engineering challenges that face the design, development, and demonstration of quantum computing. Alongside studies in physics and engineering, the field of quantum computer science now provides a focal point for discussing the theory and practice of quantum computing.

*ACM Transactions on Quantum Computing (TQC)* publishes high-impact, original research papers and select surveys on topics in quantum computing and quantum information science. The journal targets the quantum computer science community with a focus on the theory and practice of quantum computing including but not limited to: quantum algorithms and complexity, models of quantum computing, quantum computing architecture, principles and methods of fault-tolerant quantum computation, design automation for quantum computing, issues surrounding compilers for quantum hardware and NISQ implementation, quantum programming languages and systems, distributed quantum computing, quantum networking, quantum security and privacy, and applications (e.g. in machine learning and AI) of quantum computing.

For more  
information  
and to submit  
your work,  
please visit:

[tqc.acm.org](http://tqc.acm.org)



Association for  
Computing Machinery

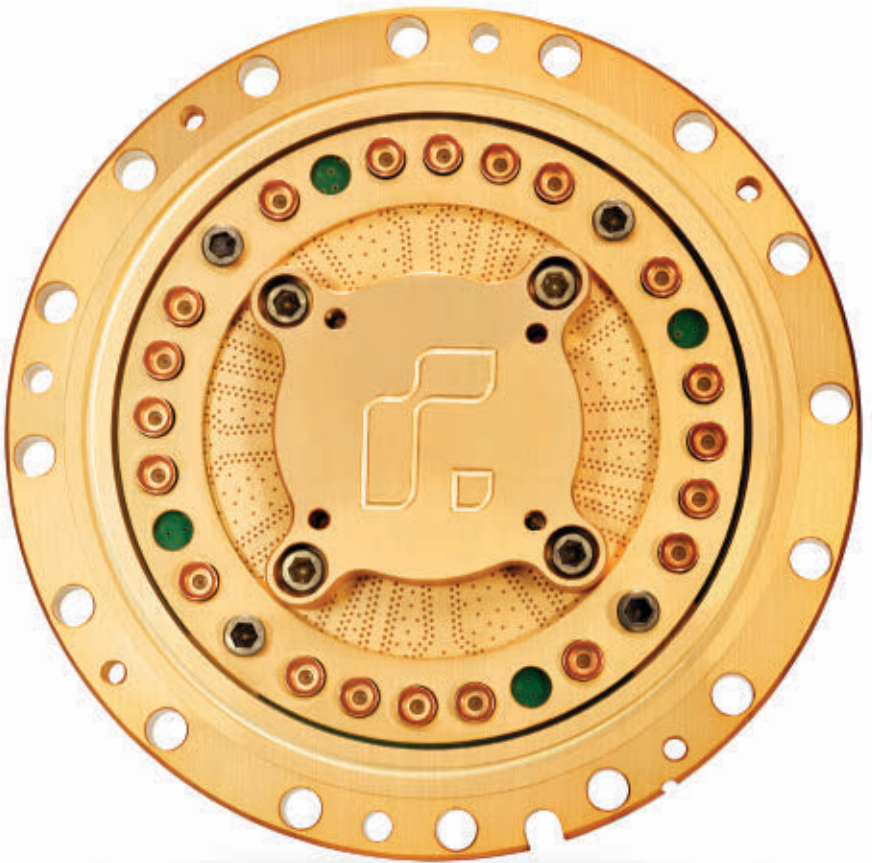
## Learning to Trust Quantum Computers

*They need to show us they can solve the biggest problems.*

**O**NE OF THE core beliefs behind the push to build quantum computers is that they will power a massive expansion in computing capability. However, how much capability could the technology really bring and, even if we can harness all that power, how can we be sure quantum computing will provide accurate answers when there is no way to run the same algorithms on conventional computers for verification?

A paper on the use of quantum entanglement in verifying the solutions to problems published in the spring of 2019 by California Institute of Technology (Caltech) postdoctoral researchers Anand Natarajan and John Wright has shown how quantum computers can prove their results are legitimate. The expansion in what is provable is likely to lead to a situation where the ability of quantum computers to demonstrate the correctness of their calculations far outstrips their ability to compute the results in the first place.

The key to checking the work of highly powerful computers lies in a result published in 1988 by Michael Ben-Or and Avi Wigderson of Hebrew University, working together with Shafi Goldwasser and Joe Kilian at the Massachusetts



**A Rigetti Computing quantum processor based on 32-qubit superconducting chip technology.**

Institute of Technology (MIT). Their original aim was to find a new way to construct authentication systems that did not rely on cryptographic functions that are assumed to be hard for computers to break. To do so, they employed the idea of the zero-knowledge proof, which lets two remote systems (known as provers) demonstrate to a third party, the verifier, that they hold a secret, without revealing the secret itself.

The provers are assumed to be able to solve any problem. To prevent the provers from cheating, the verifier uses randomly generated queries in an interactive protocol designed to catch attempts by either prover to conspire with the other to deliver a false answer. The only way they could lie reliably is to work together. To avoid this possibility, classical implementations of zero-knowledge proofs rely on setting up the tests so the two provers cannot communicate with each other directly.

The work on multiprover interactive proof (MIP) systems later expanded into delegated computing: to let systems offload tasks to remote servers. In a paper presented at the 2008 ACM Symposium of the Theory of Computing in British Columbia, Canada, Goldwasser and colleagues referenced the Harry Potter series, whose seventh and final volume was published that year, with the claim that a “muggle” machine (in this context, a conventional computer) could check the work much more capable computers claim to be able to perform. The question that challenged theoreticians was how much more those magical systems could do, and still let non-wizardly computers check whether their work is valid.

A few years after the original MIP work appeared, László Babai, Lance Fortnow, and Carsten Lund, working at the University of Chicago, showed a user with access to a machine able only

to handle problems in polynomial time could verify the work on problems that, for a Turing machine that can perform multiple operations in each computational step, require exponential time to solve. This is a complexity class known as NEXP. The open question before last year was what difference quantum entanglement between the provers would make, an arrangement that mathematicians call MIP\*.

“When MIP\* was first introduced, it wasn’t clear whether it was more powerful than MIP,” Wright says.

Researchers believed MIP\* could deliver greater power by letting the provers share states through entanglement. But it carried with it the threat of collusion. Separate provers could use their shared knowledge to collude with each other in a way that classical multiprover systems do not. This would, in turn, reduce the theoretical power of multiprover systems that use quantum-capable provers.

## Milestones

# ACM Recognizes 2019 Fellows

ACM has named 58 members ACM Fellows for their wide-ranging, fundamental contributions in areas including artificial intelligence, cloud computing, combating cybercrime, quantum computing, and wireless networking.

“Computing technology has had a tremendous impact in shaping how we live and work today,” said ACM President Cheri M. Pancake. “In highlighting the accomplishments of the ACM Fellows, we hope to give credit where it is due, while also educating the public about the extraordinary array of areas in which computing professionals work.”

The 2019 Fellows hail from universities, companies and research centers in Australia, Canada, China, Egypt, France, Germany, Israel, Italy, Switzerland, and the United States.

The 2019 ACM Fellows are: Scott J. Aaronson, *University of Texas*; Tarek F. Abdelzaher, *University of Illinois at Urbana-Champaign*; Saman Amarasinghe, *Massachusetts Institute of Technology*; Kavita Bala, *Cornell University*; Magdalena Balazinska, *University of Washington*; Paul Beame, *University of*

### Washington

Emery D. Berger, *University of Massachusetts Amherst*  
 Ronald F. Boisvert, *National Institute of Standards and Technology*  
 Christian Cachin, *University of Bern*  
 Brad Calder, *Google*  
 Diego Calvanese, *Free University of Bozen-Bolzano*  
 Srdjan Capkun, *Swiss Federal Polytechnic, Zurich*  
 Claire Cardie, *Cornell University*  
 Timothy M. Chan, *University of Illinois at Urbana-Champaign*  
 Kavianthra Mani Chandy, *California Institute of Technology*  
 Xilin Chen, *Institute of Computing Technology, Chinese Academy of Sciences*  
 Elizabeth F. Churchill, *Google*  
 Philip R. Cohen, *Monash University*  
 Vincent Conitzer, *Duke University*  
 Noshir Contractor, *Northwestern University*  
 Matthew B. Dwyer, *University of Virginia*  
 Elena Ferrari, *University of Insubria*  
 Michael J. Freedman, *Princeton University*  
 Deborah Frincke, *U.S. National Security Agency*

Lise Getoor, *University of California, Santa Cruz*  
 Maria L. Gini, *University of Minnesota*  
 Subbarao Kambhampati, *Arizona State University*  
 Tamara G. Kolda, *Sandia National Laboratories*  
 Songwu Lu, *University of California, Los Angeles*  
 Wendy Elizabeth Mackay, *Inria*  
 Diana Marculescu, *University of Texas at Austin*  
 Sheila McIlraith, *University of Toronto*  
 Rada Mihalcea, *University of Michigan*  
 Robin R. Murphy, *Texas A&M University*  
 Marc Najork, *Google*  
 Jason Nieh, *Columbia University*  
 Hanspeter Pfister, *Harvard University*  
 Timothy M. Pinkston, *University of Southern California*  
 Mihai Pop, *University of Maryland, College Park*  
 Andreas Reuter, *Heidelberg University/Heidelberg Laureate Forum Foundation*  
 Jeffrey S. Rosenschein, *Hebrew University*  
 Srinivasan Seshan, *Carnegie Mellon University*  
 Prashant J. Shenoy, *University of Massachusetts Amherst*

Peter W. Shor, *Massachusetts Institute of Technology*  
 Mona Singh, *Princeton University*  
 Ramesh K. Sitaraman, *University of Massachusetts Amherst*  
 Dawn Song, *University of California, Berkeley*  
 Salvatore J. Stolfo, *Columbia University*  
 Dacheng Tao, *The University of Sydney*  
 Moshe Tennenholtz, *Technion*  
 Giovanni Vigna, *University of California, Santa Barbara*  
 Nisheeth K. Vishnoi, *Yale University*  
 Darrell Whitley, *Colorado State University*  
 Yuan Xie, *University of California, Santa Barbara*  
 Moustafa Amin Youssef, *Alexandria University*  
 Carlo A. Zaniolo, *University of California, Los Angeles*  
 Lidong Zhou, *Microsoft Research Asia*

ACM will formally recognize its 2019 Fellows at the annual Awards Banquet in San Francisco on June 20, 2020. Additional information about the 2019 ACM Fellows, as well as previously named ACM Fellows, is available on the ACM Fellows site at <https://awards.acm.org/fellows>.



Working with Caltech professor of computing and mathematical sciences Thomas Vidick, Natarajan demonstrated in 2016 it was possible to stop cheating by entangled provers. The protocol, known as the Pauli braiding test, exploits the Heisenberg Uncertainty Principle. If one system attempts to measure a variable held by a quantum bit or qubit, information on the other properties the entangled qubit holds is destroyed. Over a series of questions, the verifier switches between the provers, asking them different things in a way that the answers can be checked for consistency, but destroying data that would let them collude.

By this point, mathematicians in the field considered the quantum versions of multiprover systems at least as powerful as their entirely classical counterparts. What remains unclear to this day is the upper bound. Natarajan and Wright were able to expand the known upper bound to systems exponentially larger than NEXP: a class called NEEEXP.

A major hurdle in developing the proof was the mismatch between the size of messages a verifier can send to the provers and the space occupied by a problem with an NEEEXP-level of complexity. Simply expressing the problem can exhaust the capacity of the verifier. “The graph becomes so large that to even give a name to a specific vertex requires an exponential number of bits,” Wright says.

If a verifier, for example, cannot identify a specific vertex in the complete graph, how does it get provers to deliver believable answers? The answer, for Natarajan and Wright, was to use what they call “introspection.”

Says Henry Yuen, assistant professor of computer science and mathematics at the University of Toronto, “The idea can be traced back to a paper of Zhengfeng Ji, although he didn’t call it introspection.”

Ji’s work led to a result superficially similar to Natarajan and Wright’s, but with a key difference: there would be a credibility gap in what the provers can demonstrate to the verifier that expands as problems become bigger. Ji also showed a carefully chosen protocol can make provers ask questions of themselves in a controlled manner and use the answers they derive to look into the much, much larger answer space. In effect, introspection acts as a form of

compression that makes it possible for classical machines to handle spaces far beyond their capabilities.

The problem space that a MIP\* system can handle could be far bigger than that and introspection may provide the way forward. “You’d hope to use successive layers of introspection,” Wright says, with each round of introspection producing a smaller protocol until the messages are small enough to be exchanged with a purely classical computer. Here again, Ji’s work provides inspiration for this nested use of introspection.

In a follow-up paper with Yuen, Vidick, and Singapore University of Technology and Design associate professor Joseph Fitzsimons, Ji showed how it was possible to use the technique recursively to arbitrarily deep levels of exponentiation. As long as the verifier could tolerate some level of doubt, they could check the work of provers on problem spaces far larger than the number of atoms in the universe. For mathematicians specializing in the field, Natarajan and Wright’s result put MIP\* on much firmer footing that may expand the power of quantum-enabled provers to the edges of complexity theory.

“This result, I think, has dramatically shifted the attitude of the community towards MIP\*: while before I think many of us would’ve hedged our bets on the complexity of MIP\*, now it seems much more conceivable—likely, even—that the complexity of MIP\* won’t just stop at NEEEXP, but could keep on going to arbitrarily large complexity classes such as NEEEXP or NEEEEEEEXP, or potentially even undecidable problems,” Yuen says. “The Natarajan-Wright result is giving us really compelling evidence of the tremendous complexity of MIP\*.”

Wright says work continues on recursively using introspection, though it is not yet clear whether it will work. The implications would be pretty big, he says, if undecidable problems are also shown to fit into MIP\*. “If two people told you that a Turing machine halts on a given input, how would you check that without just running it until it halts, which could take an unbounded amount of time?” he asks. Quantum multiprover, in this scenario, would provide the way to determine whether those people were telling the truth.

How they knew would likely remain a mystery as there is no prospect of anyone being able to construct a computer that can provide the answer.

The result also circles back to the zero-knowledge proofs of cryptography research that kicked off work on multiprover systems. Says Yuen, “One of my recent papers shows that the class MIP\* is equal to the class zero-knowledge MIP\*. In other words, every interactive proof conducted with quantum-entangled provers can be transformed into an equivalent zero-knowledge protocol.”

If undecidable problems do fall into MIP\*, the verifier would have proof there is an answer to a problem that computer science today considers unknowable, but not what the answer is. “While I can’t imagine that there could be practical applications of this fact, it would be an amusing fact if nothing else.”

Yuen says there are potential practical benefits for much smaller problems: “One could envision people inventing protocols for verifying extremely large computations where the verifier could be extremely succinct in its interrogation of the provers.”

The research continues, although much remains uncertain as to how far mathematicians can push the upper bounds of theoretical capability. What has been shown so far is that quantum entanglement could underpin a massive expansion in what is computable. ■

#### Further Reading

Natarajan, A. & Wright J.

NEEXP  $\subseteq$  MIP\*

arXiv preprint (2019). arXiv: 1904.05870

Goldwasser, S., Kalai, Y.T., and Rothblum, G.N.

Delegating Computation:

Interactive Proofs for Muggles

Proceedings of the 40<sup>th</sup> Annual Symposium

on the Theory of Computing (STOC), May

2008, DOI: 10.1145/1374376.1374396

Ji, Z.

Compression of Quantum

Multiprover Interactive Proofs

arXiv preprint (2016). arXiv: 1610.03133

Grilo, A.B, Slofstra, W., and Yuen, H.

Perfect Zero Knowledge for Quantum

Multiprover Interactive Proofs

arXiv preprint (2019). arXiv: 1905.11280

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

# Dark Web's Doppelgängers Aim to Dupe Antifraud Systems

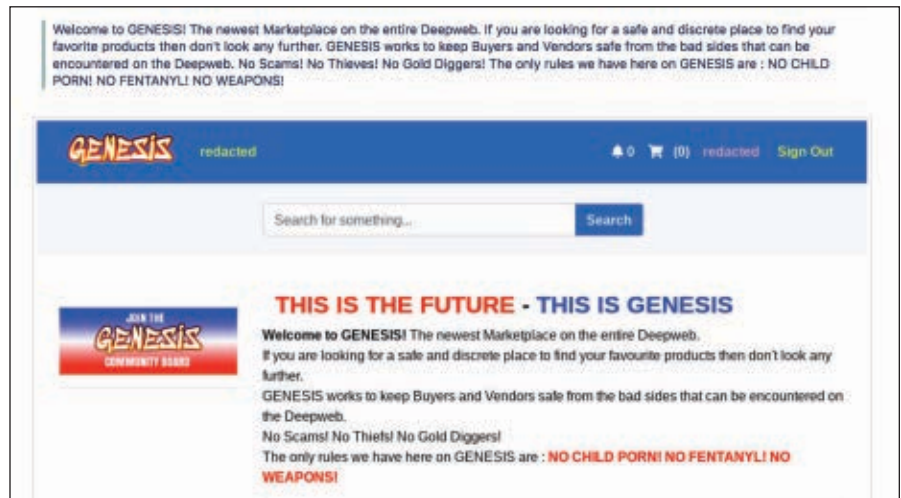
*Digital doppelgängers that fool online payment fraud detection systems are a threat to your bank balance.*

**D**EEP WITHIN THE encrypted bowels of the dark Web, beyond the reach of regular search engines, hackers and cybercriminals are brazenly trading a new breed of digital fakes. Yet unlike AI-generated deepfake audio and video—which embarrass the likes of politicians and celebrities by making them appear to say or do things they never would—this new breed of imitators is aimed squarely at relieving us of our hard-earned cash.

Comprising highly detailed fake user profiles known as digital doppelgängers, these entities convincingly mimic numerous facets of our digital device IDs, alongside many of our tell-tale online behaviors when conducting transactions and e-shopping. The result: credit card fraudsters can use these doppelgängers to attempt to evade the machine-learning-based anomaly-detecting antifraud measures upon which banks and payments service providers have come to rely.

It is proving to be big criminal business: many tens of thousands of doppelgängers are now being sold on the dark Web. With corporate data breaches fueling further construction of what market analyst Juniper Research calls “synthetic identities,” Juniper estimates online payment fraud losses will jump to \$48 billion by 2023, more than double the \$22 billion lost in 2018.

The existence of a doppelgänger dark market was first discovered in February 2019 by security researcher Sergey Lozhkin and his colleagues at Kaspersky Lab, the Moscow-based security software house. His team was carrying out their regular threat analyses on several underground dark forums, “when we discovered a private forum where Russian cybercriminals were hosting information about something called the Genesis Store,” Lozhkin says.



**The home page of Genesis Market, a referral market focused on scam prevention for both vendors and buyers. As the site says, “Our mission is to create a market where scams are not be tolerated from neither vendors nor buyers. Period.”**

## Fraud-on-Demand

When the security researchers gained access to it, Genesis turned out to be an invitation-only, crime-as-a-service, identity-theft e-shop containing sophisticated doppelgänger datasets mimicking 60,000 people, including, in many cases, their stolen logins and passwords for online shops and payment service providers. Each identity was for sale, at prices varying from \$5 to \$200 (depending on the amount of useful credit-card-hacking data each contained.) Once launched in a browser, each doppelgänger could then be used for fraud.

What is actually going on here, it turns out, is the turning of one of the major pillars of latter-day antifraud technology against itself. To detect fraudulent transactions in real time, credit card companies, banks, and payment processors use commercial machine learning (ML) anomaly-detection software, which determines whether the dataset covering the devices and behaviors of the user attempting to make a transaction are

close enough to that of a previously authenticated template that digitally describes legitimate users.

This authenticated template is known in the payments industry as the user’s “digital mask.” Such masks have two major components: a device fingerprint (which includes device fingerprinting factors like your commonly used IP addresses, firmware version, installed plugins, time zone, screen resolution, preferred window sizes, GPU type, OS version, and browser cookies), and a behavioral profile based on factors such as how ‘clicky’ you are when using a mouse or touch-screen online, the amount of time you typically spend at an e-store, your usual items of interest, the amount you usually spend, and whether you tend to buy digital or actual goods.

However, here’s the thing: hackers can penetrate payment systems and steal copies of those masks. Alternatively, and more likely, they can use malware planted on poorly defended computers and smartphones to transmit device and behavioral data to them

over a botnet, so they can then create their own competing masks from scratch. Then they just launch them through a customized browser the Genesis gangsters call Tenebris, and connect using an IP-address-mimicking proxy to make transactions that fool the fraud detection systems.

Lozhkin says the Genesis dark market has effectively turned the decades-old practice of credit card fraud (also known as ‘carding’) into a new, highly targeted, industrial-scale criminal activity. It’s one slick high-tech operation: for instance, the Kaspersky team found the fraudsters had written algorithms that automatically price each doppelgänger, based on its fraudulent earnings potential.

Because the Genesis Store uses botnets to harvest data to construct its own fake masks, criminals can target victims directly. “A search panel lets users search for specific bots, website logins and passwords, the victim’s country, operating system, date the profile first appeared on the market—everything is searchable,” Kaspersky Lab says in an analysis of the offerings on Genesis that it has posted on its Securelist blog.

It appears to be popular. “We are seeing monthly increases in the numbers of [data-stealing] bots that are being sold on the market, and also in the numbers of cybercriminals that want to buy stolen information, so it’s still growing,” says Lozhkin.

### Combating the Threat

On the front line of the antifraud industry, doppelgängers are indeed being seen as a latter-day adversary. “Our customers continue to see fraud attacks of many different types, including those using doppelgängers, and the landscape of these attacks changes daily,” says David Excell, founder of Featurespace, one of the leading antifraud anomaly detection technology providers, with bases in Cambridge, U.K. and Atlanta, GA.

To fight digital payment fraud, Featurespace has developed a probabilistic machine learning platform that alerts finance firms to fraud attempts. Called the Adaptive Real-time Change Identifier (ARIC), it lets companies “construct their own doppelgänger for each consumer, in the form of an individual behavioral profile,” says Excell. “Using these profiles, we’re able

to identify if the interaction for a customer is normal, based on how they’ve behaved in the past. Typically, a fraudster is revealed when they attempt to monetize their attack, because at that point, their behaviors aren’t matching the ones we expect to see from the actual customer.”

Like its rivals in the anomaly detection arena, Featurespace does not reveal how its proprietary algorithms spot that mismatch. However, Excell says, defending against digital fraud is about far more than the ‘secret sauce’ behind their ever-changing algorithms. “Protecting a financial institution from fraud is no easy task and relies on a combination of data, technology and processes,” he says.

One measure finance firms can take in the war against fraud is to eschew the use of standalone disconnected businesses. “Financial institutions tend to operate multiple business units in silos, making it difficult to join systems together. This is one of the weaknesses that fraudsters often try to exploit,” says Excell.

By linking data sources in such units together and having oversight of customer interactions across many channels, Excell says, “financial institutions can build individual behavioral profiles in real time to spot the unusual or incorrectly mimicked behavior of the fraudster.” As an example, he points to how the channels a bank would need to integrate would include traffic on its online service, its mobile app, transactions undertaken in the branch, and also those on the phone.

It’s the sunk costs in older technology, such as the kit and code in those silos, that are leaving some firms behind in the cyber arms race against fraudsters, says Ian Thornton-Trump, an IT security analyst at cybersecurity insurer and underwriter AmTrust Financial Services in New York City. “The current cybersecurity problem has little to do with security controls or their effectiveness: the arch nemesis of cybersecurity is network complexity and technological debt,” he says.

The problem, he says, is that while physical network complexity has changed little, logical network complexity has gone through the roof with the addition of low-cost, off-premise, cloud-hosted software-as-a-service

# ACM Member News

## PRODUCING LEADERS FOR THE NEXT CS GENERATION



“I was lucky,” says Huan Liu, a professor of computer science (CS) and engineering at Arizona State

University (ASU), when reflecting on the trajectory of his career. “I picked computer science in college, I picked AI (artificial intelligence) when doing my Ph.D. research, I picked machine learning after I graduated, and then I moved into data mining.”

Liu earned master’s and Ph.D. degrees in computer science from the University of Southern California, following an undergraduate degree in computer science and electrical engineering from China’s Shanghai Jiao Tong University.

Before joining the faculty at ASU, Liu worked in the Research Labs of Telecom Australia Research Labs, and was affiliated with the faculty of the National University of Singapore.

Liu’s research interests are in data mining, machine learning, social computing, and artificial intelligence. He investigates interdisciplinary problems that arise in many real-world, data-intensive applications with high-dimensional data of disparate forms, such as social media.

His current research focus is on causal learning with data, detecting fake news and bots, and also how to preserve privacy in social media. He works to discover actionable patterns or insights from data, particularly social media.

Liu is highly invested in his students. His senior doctoral students guide junior members and help them to succeed, and through this process they naturally become leaders. “The key for me is to produce the next generation of top computer scientists,” he says.

Liu takes pride in his graduates, pointing out that many co-authors of his published research are former students who have now moved on. “I follow my students to success.”

—John Delaney

(SaaS), platform-as-a-service (PaaS) and infrastructure-as-a-service (IaaS) facilities to those physical networks. Understanding that complexity, adopting a roadmap that reduces it, and removing the least secure parts (probably the old legacy systems) is one key to enhancing security, Thornton-Trump says. “If you know what ‘normal’ is, and what systems you need to protect, this would make anomaly detection far more effective.”

### Bounty Hunters vs. Fraudsters

Still another way to defeat doppelgänger fraudsters is to harness the skills of crowdsourced ethical hackers, says Laurie Mercer, a security engineer with London-based HackerOne, a bug bounty firm that offers cash rewards to the more than 400,000 ethical hackers it has registered on its books. By probing corporate systems, they can help firms find the kind of vulnerabilities that let criminals plant their data-stealing bots.

“Online payment systems and e-commerce brands have sophisticated technology in place to prevent fraud, but humans can often outwit these technical controls. And, it takes a human to outwit a human,” Mercer says, suggesting his organization’s hacker army is the best way to unleash human intelligence against the fraudsters. He’s not alone: Goldman Sachs, the U.K. challenger bank Starling, and the Germany-based direct bank N26 are all financial organizations working with white-hat hackers to secure their digital assets. “Anyone who finds a security vulnerability on these companies’ assets can report them to the company, potentially earning a bounty,” Mercer says.

The human factor matters at another level, too, says Amanda Widdowson, cybersecurity champion at the Chartered Institute of Ergonomics & Human Factors in London, noting that the Genesis Store’s doppelgängers are more of a threat if they include login/password pairs, something people have the power to limit spilling to fraudsters.

“The temptation is to use the same, easy-to-remember and so easy-to-guess password for multiple applications. The danger is, if one application is compromised, our other applications

## Goldman Sachs, the U.K. challenger bank Starling, and Germany’s direct bank N26 are all working with white-hat hackers to secure their digital assets.

are also at risk. What’s needed is more investment in alternative methods of user validation such as biometrics—face, iris, and fingerprint recognition—to reduce reliance on limited human memory capacity,” Widdowson says.

Kaspersky Lab agrees, urging that people use strong passwords, biometrics, and multi-factor authentication to keep the bots out. “People should be practicing safe cybersecurity habits, implementing the same methods they would to prevent any malware infection on their personal devices,” says Lozhkin.

Most importantly, he says, since the doppelgängers were constructed via botnets, “strong cooperation and quick information sharing between cybersecurity vendors and law enforcement agencies around the world will be key to a fast shutdown of such services.”

For the future, however, the banking and finance industry’s move to voice-based account management services—following on the success of voice assistants like Amazon’s Alexa, Google Home, and Apple’s Siri—may end up making them vulnerable to hack attacks via deepfake audio. That might add a voiceprint component to digital doppelgängers, with heavy ramifications for services.

Indeed, the threat that fake AI-generated voices pose to humans, rather than voice assistants, is already more than apparent. In September, it emerged that the CEO of a U.K.-based energy company had been convinced that he was talking on the phone to his boss in Germany, who asked him

to make a \$220,000 cash transfer to Hungary, which he did. However, his supposed superior was actually a criminal using very convincing voice synthesizer software that had been trained to mimic every aspect of the target’s voice, from his tonality to his German accent. The money is lost.

Such attacks will force changes on the payments industry. “Since the attack surface of deepfakes is primarily banking information and money transfer functions, I believe we will see mandatory holds for cash amounts over certain levels, with additional authorizations required to complete money transfers,” says Thornton-Trump.

“Although this may impede business agility, the risk of being victimized by a multi-thousand- or multi-million-dollar fraud exceeds the inconvenience.” □

### Further Reading

Losses from Online Payment Fraud to More than Double by 2023, Reaching \$48 Billion Annually, Juniper Research, Nov. 20, 2018 <https://www.juniperresearch.com/press/press-releases/losses-from-online-payment-fraud>

Digital Doppelgangers: Cybercriminals cash out money using stolen digital identities, Kasperky Lab *Securelist blog*, April 9, 2019 <https://securelist.com/digital-doppelgangers/90378/>

Cimpanu, C. Cybercrime market selling full digital fingerprints of over 60,000 users, *ZDNet*, April 9, 2019 <https://www.zdnet.com/article/cybercrime-market-selling-full-digital-fingerprints-of-over-60000-users/>

Providing a unique behavioral analytics approach to prevent fraud attacks Featurespace, 50 Smartest Companies of the Year 2017, *The Silicon Review* <http://bit.ly/2lcD8IE>

Marks, P. Bounties Mount For Bugs, *ACM News*, August 23 2018, <https://cacm.acm.org/news/230582-bounties-mount-for-bugs/fulltext>

Statt, N. Thieves are now using AI deepfakes to trick companies into sending them money, *The Verge*, Sept. 5, 2019 <http://bit.ly/2mKod80>

Paul Marks is a technology journalist, writer, and editor based in London, U.K.

# Tracking Shoppers

*Retailers of all stripes are using technology to follow consumers through their brick-and-mortar stores in order to develop detailed profiles of their shopping habits and preferences.*

**M**OST PEOPLE ARE AWARE that if they use the Internet, social media sites, or electronic payment systems, their activity is tracked, thereby creating a digital footprint and roadmap that can reveal a significant amount of personal data and activity patterns. Online retailers and marketers are leveraging (some may say exploiting) this treasure trove of data, helping them craft advertisements and marketing messages that specifically target consumers in the hopes of driving sales, or creating a deeper level of engagement with customers.

It is no surprise physical brick-and-mortar retailers are trying to replicate the engagement strategies that have worked well for their online counterparts (or which they themselves have used online). Using a combination of technology and social engineering, retailers are tracking shoppers in and around their stores to leverage shopping and behavioral data to increase sales, build greater customer loyalty and, in some cases, deter or thwart theft or fraud.

Indeed, according to Retail Systems Research, a research company that tracks technology use by predominantly North American retailers, 61% of those surveyed in 2018 said they see potential value in technology that tracks shoppers as they walk through the store, up from 51% the year before. Similarly, the use of in-store cameras and video analytics for customer experience purposes hit an interest level of 49% in 2018, up from 43% in 2017.

Not surprisingly, the use of video surveillance technology appears to be more pervasive in other parts of the world, such as Asia, where consumer privacy protections are weaker and shoppers appear to be more receptive to the use of new technology, provided they receive tangible benefits.



**Thermal after-images of shoppers can be tracked as a “heat map,” to inform retailers of shopper traffic patterns in their stores.**

“Especially in the APAC (Asia-Pacific) region, you see a lot of companies that will use surveillance footage that will capture the images of customers, and then match that to their loyalty programs,” explains Auria Moore, director of solution strategy at Stibo Systems, a provider of master data management (MDM) solutions that allow retailers to maintain a single view of all customer data. “Some companies are doing it better than others.”

Cosmetics seller Sephora is an example of a retailer that has successfully deployed video surveillance and analytics to provide a more granular view of store traffic patterns, while also providing more individualized marketing and incentives to customers. Working with AllGoVision, a provider of video analytics solutions, Sephora deployed 54 Axis cameras at store entrances and near various product sections across 10 Sephora retail outlets located in Malaysia.

The video solution can count the number of people entering and moving about the store, and feeds this data into the analytics platform. The data is used to monitor and address checkout line wait times, and to generate store traffic flow maps and heat

maps; the data is then fed into a model that can be used to optimize the retail space. The solution also incorporates facial detection and recognition technology, which allows Sephora to better measure product engagement with individual users, create demographic and marketing profiles for products, and can be used to aid in shoplifter detection and prevention.

Camera data that includes customer positioning, daypart engagement information, and foot traffic can be combined with data analytics “to help marketing make decisions about how you place a promotion, what time of day you have more people passing by, and how you set up your retail space, all based on the data you’ve collected,” says Aji Anirudhan, global sales and marketing manager for AllGoVision, Inc.

However, the use of cameras feels invasive to some shoppers, which is why some retailers choose a different method of tracking shoppers. Beacons—small devices that send signals to Bluetooth-equipped devices such as smartphones—can be used to detect the presence of shoppers, recognize them individually, and interact with each one by providing an offer (such as

a coupon) to further strengthen the store/customer relationship.

Retail chains with a presence in the U.S., such as Target, Walmart, Urban Outfitters, and Neiman Marcus, have deployed such Bluetooth-based wireless technology to connect and interact with customers over the past few years. Unlike Wi-Fi, beacons work well indoors, can track customer location within stores with accuracy ranging from within a foot to a few yards, and are energy-efficient, making them ideal for the retail environment.

Target, one of the notable large retailers to embrace beacons a few years ago, had rolled out the technology to most of its stores by last year. Shoppers who download the Target app (which is equipped with software that interacts with the beacons) will display their location on the app's map as they move throughout the store.

Mike McNamara, Target's CIO, explained in a video discussing the technology that the store-tracking technology is "a bit like driving your car with GPS. Just click on an item from your list, and the app will indicate on a store map the precise aisle where you will find your item." McNamara noted that the app will even tell you if the product is on sale.

Because beacons are designed to tie offline behavior (shopping in a physical store) to online behavior (which is often tracked by the use of a store app or via browsing history on a mobile or desktop device), retailers can capture, analyze, and make predictions on a substantial amount of personal data. A retailer then can use this data to increase both online and offline sales by offering discounts, loyalty points, or other incentives in order to further engage customers and drive sales. That means consumers' private habits are quickly becoming public knowledge, which is then used for commercial purposes.

"I still see that people are willing to trade the benefit of giving up their personal information," says Norman Shaw, an associate professor at Ryerson University in Toronto, Canada, who focuses on digital privacy issues. "They want to give it up because they get the convenience."

In a retail environment, a beacon placed adjacent to a product would be alerted to a shopper's proximity to the

## The tide may be changing, as there appears to be growing interest in the use of beacons among retailers.

beacon by the app on the shopper's phone. Once the smartphone app recognizes the beacon, it sends the data back to a company's server, and the app then can be prompted to display ads for products the customer is likely to buy, based on data contained in that shopper's profile. Alternatively, a store can use a strategy called "retargeting," which refers to the practice of sending a follow-up offer, such as a coupon, after the shopper leaves the store, to entice the shopper to return.

Despite their potential benefits to both retailers and customers seeking deals or personalized offers, the adoption rate of beacons has not matched the hype. Indeed, much of the coverage of beacons being deployed at large retailers such as Walmart, Target, and other retailers occurred in the 2013–2017 timeframe; since then, few success stories have been popping up in the popular or trade press. "Beacons are valuable, [but] they blend the lines of scariness at the same time," Moore says. "I don't think retailers are adopting them as quickly as they thought."

This could be due to growing consumer concerns about privacy. Most large, reputable retailers notify customers they may be tracked if they download their mobile application, although these disclosures are often buried within their terms and conditions, rather than being highlighted within the application itself. However, a large number of third-party location-marketing firms generate beacon tracking code and bundle it into a tool kit that developers can insert into benign-looking weather or newsfeed apps. A retailer would then be able to track where a customer spends the most time in a supermarket, even if

that user wasn't aware he or she was being explicitly tracked.

That said, the tide may be changing, as there seems to be increasing interest in beacons among retailers, according to data from Retail Systems Research. The use of beacons for in-store marketing applications saw an 11% year-over-year increase in interest among retailers between 2017 and 2018, from 38% to 49%. Similarly, the use of beacons for store perimeter marketing saw an interest level of 44% in 2018, up from 37% in 2017.

"The interest [in beacons]" is there, Rowen says, noting that he expects this year interest in beacons for in-store marketing is likely to reach nearly 60% of all U.S. retailers, based on the established historical trajectory of retailer interest in beacons. From the retailers' perspective, the biggest challenge is that there are so many potential technological innovations that could be used, that picking a winner or devising a strategy has become a massive challenge.

Indeed, Rowen says many retailers have been afflicted with the 'if I buy something this year, who's to say it's still going to be valuable next year?' conundrum, particularly because store margins are so thin that any mistake likely will have a huge, potentially devastating financial impact.

"There are these exciting technologies that are available," Rowen says. "But if [they're] not guaranteed that they will increase sales, reduce shrink (theft), increase footfall (the number of people visiting a store), drive conversion (from shopping to buying), or prompt ancillary sales," stores are unlikely to commit operational expenditures to deploying a new technological solution.

Tracking shoppers, however, is not solely about marketing. Retailers are also keen to use technology to track shoppers to identify potential shoplifters. Shrinkage, in industry parlance, remains a major problem for retailers, and due to the legal and safety issues surrounding the apprehension of thieves after they have taken merchandise or cash, there is a push to identify and thwart theft prior to it occurring.

A combination of video surveillance and machine learning can be used to not only identify people who populate the BOLO ("be on the lookout for") lists

of suspected or known thieves, but also those who may display body-language cues (nervous tics, eyes scanning the area rapidly, etc.) that could indicate they may be looking to steal.

Vendors including FaceFirst, Vaak, and Third Eye have conducted extensive public relations campaigns highlighting the effectiveness of their solutions; FaceFirst, for example, claimed its software is in use at “hundreds” of retail locations. However, it is difficult to accurately quantify what percentage of retailers are actively using facial recognition or video analytics to deter or prevent fraud and theft, because few retailers choose to publicize their use of the technology, lest they incur the ire and scorn of privacy-minded shoppers.

Anirudhan says retailers outside the U.S. may be somewhat more comfortable discussing how they use video technology, possibly because privacy is not expected in some markets, such as Asia. Using video analytics for marketing purposes, such as making the store easier to navigate, or by offering shoppers discounts or other benefits, is likely to be acceptable to shoppers in those regions, but using the technology to deter theft may not be as welcome as there is no direct benefit to shoppers.

Further, facial recognition technology, as well as gesture or body-language analysis technologies, are not yet accurate or reliable enough for some retailers to use, mainly because there isn't a large-enough repository of real-world video data available that can be used to train recognition systems so they are accurate. Particularly in privacy-conscious markets such as the U.S., these types of systems need to be accurate nearly 100% of the time in order to gain the confidence of retailers.

Currently, regulation of video data varies by jurisdiction. In more autocratic jurisdictions, such as China, it's unlikely that meaningful regulation on the use of video will be enacted. However, in privacy-focused Europe, the General Data Protection Regulation (GDPR) appears to apply to not just text-based data, but also video. In October 2018, the Austrian regulator fined a company 4,800 euros for capturing detailed images of passers-by and not having adequate

signage to alert them their images were being captured.

In Canada, the Information and Privacy Commissioner of Ontario released in 2015 updated guidelines governing the lawful use of video surveillance, which laid out a number of collection, use, access, and retention requirements.

While there is no law in the U.S. currently requiring retailers to disclose their customers are being tracked or recorded on video, that may change, according to Moore.

“I think you're going to see digital compliance tightness spill over into the retail store, and I don't know what that looks like yet,” Moore says. “It could be as simple as a sign that hangs in front of every Walmart store that says, ‘video surveillance is in use and you agree to be captured for [marketing or security purposes].’ But in order to get shoppers to provide this consent, retailers will need to provide a worthwhile trade-off for the use of shoppers' private data,” Moore says.

“My data is my gold, so if I'm giving up my gold, what am I getting back?” Moore says. “I think that the experience has to match up for consumers. You look at things like the Fitbit and Apple Watch, people are okay with giving up their data if it's going to help them monitor their heart. You're okay with giving up data if it's going to help them in their day-to-day lives. [Giving up privacy] for a coupon or promo code while I'm in the store is not good enough.” ■

#### Further Reading

*Kilcourse, B. and Rowen, S.*  
Location Intelligence in Retail:  
The Value of 'Where', Retail Systems  
Research, February 27, 2019  
[https://www.rsresearch.com/research/  
location-intelligence-in-retail-  
the-value-of-where](https://www.rsresearch.com/research/location-intelligence-in-retail-the-value-of-where)

*Lang, E.*  
Store People Counting and Gesture  
Recognition, December 20, 2018  
[https://www.youtube.com/  
watch?v=cryEd6XD41k](https://www.youtube.com/watch?v=cryEd6XD41k)

There's More in Store With the Target App,  
A Bullseye View, September 20, 2017  
[https://corporate.target.com/  
article/2017/09/target-app-mike-mcnamara](https://corporate.target.com/article/2017/09/target-app-mike-mcnamara)

**Keith Kirkpatrick** is principal of 4K Research & Consulting, LLC, based in New York.

© 2020 ACM 0001-0782/20/2 \$15.00

#### Milestones

## Gordon Bell Prize To ETH Zurich Team

ACM named a six-member team from the Swiss Federal Institute of Technology (ETH) Zurich recipients of the 2019 ACM Gordon Bell Prize for their project, “A Data-Centric Approach to Extreme-Scale Ab initio Dissipative Quantum Transport Simulations.”

The ETH Zurich team introduced DaCe OMEN, a new framework for simulating the transport of electrical signals through nanoscale materials (like the silicon atoms in transistors). The team simulated how electricity would flow through a two-dimensional slice of transistor consisting of 10,000 atoms, 14 times faster than an earlier framework used for a 1,000-atom system.

The ACM Gordon Bell Prize, which tracks the progress of parallel computing and rewards innovation in applying high-performance computing to challenges in science, engineering, and large-scale data analytics, was presented to the researchers by ACM president Cherri M. Pancake and Arndt Bode, chair of the 2019 Gordon Bell Prize Award Committee, during the International Conference for High Performance Computing, Networking, Storage and Analysis (SC19) in Denver, CO.

The ETH Zurich team used their simulation to develop a map of where heat is produced in a single transistor, how it is generated, and how it is evacuated, in the hope a deeper understanding of thermal characteristics could inform development of semiconductors with heat-evacuating properties.

The team also built a graphical interface for the framework that includes a visualization of dataflow in lieu of a simple textual description. Anyone running the code can use the image representation to interact with the data directly.

Winning team members include Alexandros Nikolaos Ziogas, Tal Ben-Nun, Timo Schneider, and Torsten Hoeffler from ETH Zurich's Scalable Parallel Computing Laboratory, as well as Guillermo Indalecio Fernández and Mathieu Luisier from ETH Zurich's Integrated Systems Laboratory.

# V viewpoints

DOI:10.1145/3376125

Carlos Iglesias, Dhanaraj Thakur, and Michael L. Best

## Global Computing Are We Losing Momentum?

*Estimating when the second half of the world will come online.*

**T**HE INTERNATIONAL TELECOMMUNICATIONS (ITU) estimated that more than half (51.2%) of the world's population was using the Internet (defined as using any Internet-based application from any location and device over the last three months) by the end of 2018. The outcome of a stunning pace of global growth, this marks the first time a majority of the world is using the Internet. However, it also highlights the fact that approximately half the world is not yet connected to the Internet. While the threats to Internet users are many—from misinformation or cyberstalking to an increasing loss of control over our personal data and privacy—to be offline today means to be excluded from opportunities to learn and earn, to access valuable services, and to participate in democratic debate. Although a discussion on the value of connectivity is essential, and we recognize a need to define just what sort of Internet we want to build for the future—this column focuses on those who remain unconnected and when they may come online.



In 2016, the United Nations (U.N.) declared Internet access a human right.<sup>a</sup> If we accept this U.N. position then those who remain unconnected are not merely inconvenienced, they are being denied

a fundamental human right. The U.N. has developed connectivity targets including the U.N. Broadband Commission 2025 target (75% of the world using broadband Internet, 35% in least-developed countries or LDCs) and the U.N. Sustainable Development Goal 9c target for 2020 (universal and affordable

<sup>a</sup> 32<sup>nd</sup> session of the Human Rights Council; <http://bit.ly/2sh7Flx>



access in LDCs). Based on the models presented in this column we are unlikely to meet either of these goals; instead our results indicate a general slowdown in growth in Internet use.

### A Model for Estimating Global and Regional Internet Use

To predict future Internet connectivity levels we studied historical data provided by the International Telecommunications Union (ITU).<sup>b,c</sup> Limitations exist with this data, including the relatively short time series available, year-on-year intervals, and the different age ranges used to identify Internet populations in different countries. Also, the data is based on several assumptions including linear growth of the overall population, no substantial reduction of existing Internet users, and no significant new mass Internet connectivity projects.

Finally, we note the ITU could be more transparent by publishing the methodology used to develop these estimates. Publishing the methodology would help ensure the estimates are informed by best practice<sup>2</sup> and consider correlating predictor variables such as population size, life expectancy, economic growth, literacy and education rates, and the policy environment.

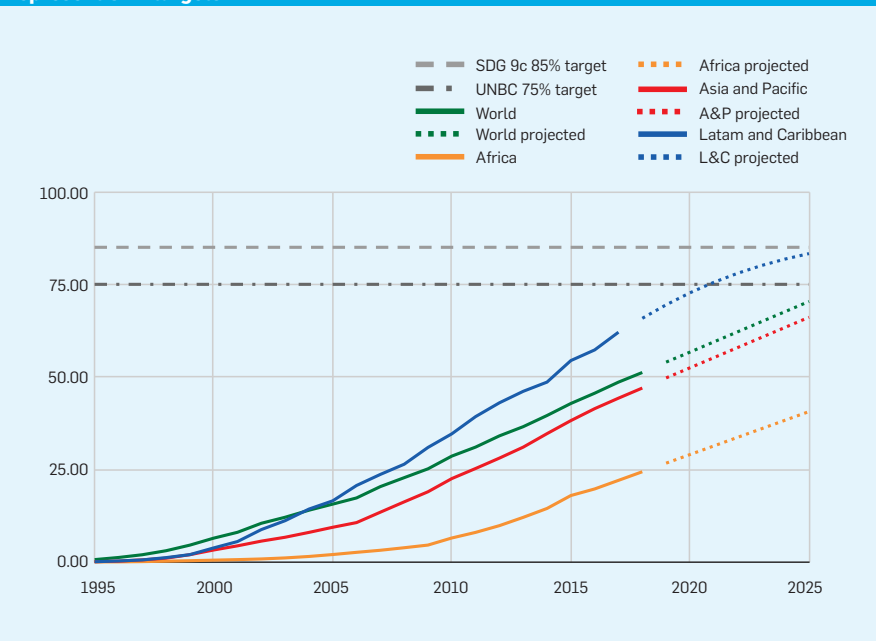
### Data Projections and Results Discussion

We have proposed a projection model of the time series data and use it to

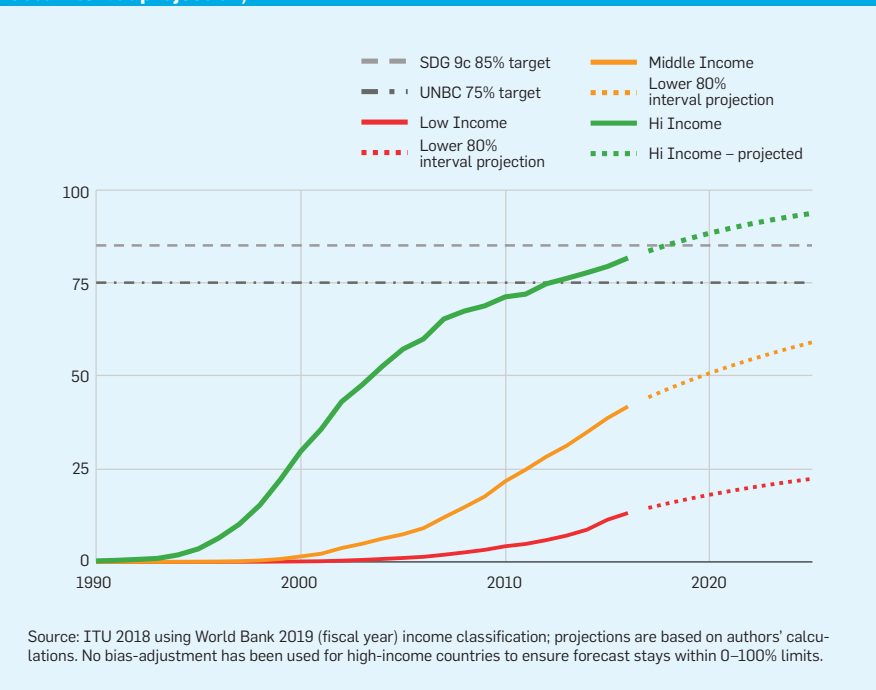
b This is part of the ITU's World Telecommunications/ICT Indicators (WTI) database (June 2019 update). Other secondary data sources we consulted include: ITU ICT Statistics: Key ICT indicators for developed and developing countries and the world (totals and penetration rates: 2005–2018)—see <https://www.itu.int/en/itu-d/statistics/documents/>; ITU ICT Statistics: Percentage of individuals using the internet (2000–2017)—see <https://www.itu.int/en/itu-d/statistics/documents/statistics/2018/>; and World Bank: Individuals using the internet (% of population: 1989–2017)—based on ITU data—see <https://data.worldbank.org/indicator/it.net.user.zs>

c Our predictions are based upon exponential smoothing and ARIMA models for non-stationary time-series forecasting with 95% confidence intervals. We use the point averages as reference and the 80% lower confidence interval values as an alternative scenario—following the traditional diffusion of innovation theory S-curve pattern as described in this column. Calculations are available at the Web Foundation github repository; see <https://www.github.com/TheWebFoundation/Internet-Users>

**Figure 1. Percentage of population using the Internet (2005–2017/2018, solid lines) and global and regional forecasts (2018/2019–2025, dashed lines). Grey horizontal lines represent U.N. targets.**



**Figure 2. Individuals (%) using the Internet in high, middle, and low-income countries (1990–2016 reported data, solid lines, and 2017–2025 projections, dashed lines—lower 80% interval projection).**



Source: ITU 2018 using World Bank 2019 (fiscal year) income classification; projections are based on authors' calculations. No bias-adjustment has been used for high-income countries to ensure forecast stays within 0–100% limits.

**Forecast (rounded point average) of Internet users globally and in the least-developed countries (2020 UN SDG 9c target and 2025 UN Broadband Commission target).**

Forecast Year	Estimated Internet Use - Global	Estimated Internet Use - LDCs
2020 - UN SDG 9c (universal LDCs access target)	57%	23%
2025 - UN Broadband Commission (75% world broadband target, 35% in LDCs)	70%	31%

forecast by simply continuing the model dynamics into the future. This allows us to get an estimation of how close (or far) we are to achieving the aforementioned U.N. SDG 2020 and Broadband Commission 2025 goals.

We focus on the three regions where the next 50% of users are most likely to come from: Latin America and the Caribbean; Asia and the Pacific; and Africa. Figure 1 provides the historical data, between 1995 and 2017/2018, of average Internet use rates for these regions, as well as globally. We also added projections for each region and the world up to 2025; the table in this column includes our estimates of Internet use rates for those target years.

Based on these models, we will meet neither the U.N. SDG 9c 2020 nor Broadband Commission 2025 targets at current rates. The 2020 target calls for universal access in LDCs. “Universal access” does not necessarily imply that 100% of the population is online. Rather, we take an 85% adoption rate as *universal access*.<sup>d</sup> Clearly, we are a long way from even that milestone globally, and particularly for LDCs. The 2025 target calls for a 75% global broadband Internet use rate (and 35% in LDCs). Instead, by then we project 70% for global Internet use (broadly defined as mentioned earlier and may not include actual broadband use) and 31% in LDCs.

In attempting to understand these trends we draw on diffusion of innovation theory (for example, Rogers<sup>3</sup>) and specifically its application to Internet use (see for example Wolcott, et al.<sup>4</sup>). That research looks at many aspects of technology adoption including types of adopters and factors that drive overall adoption. This, in turn, has been applied to studies on trends in global and regional Internet use.<sup>1</sup> According to the diffusion of innovation framework, adoption over time is typically de-

scribed with an S-curve. That is, we will observe slow growth at first (indicating adoption among early users), followed by rapid growth (as the technology becomes mainstream), and ending with slow growth again (close to universal access as late adopters trickle on board).

This S-curve is present in Figure 2 when we look at high-income countries as defined by the World Bank and depicted by the green line.<sup>1</sup> This curve shows an initial slow-growth phase for the first five years followed by 12 years of rapid growth between 1995 and 2007, followed again by slow growth from 2008 to present. This final slow-growth period of late adopters started with more than 65% of the entire population already online.

Turning to middle and low-income countries, we do not observe the same S-curve pattern compared to the high-income data. In particular, for middle and low-income countries, we see a slow-growth late adopter regime well before we hit universal access. While the final slow-growth phase in high-income countries started at approximately 65% penetration and continues above 85%, the projected slow-growth phase in middle and low-income economies could start as soon as 42% and 13% penetration rates respectively, without being preceded by an extended period of rapid exponential growth. In other words, while Internet use rates in high-income countries have begun to plateau around the universal access level, in low and middle-income countries the plateau may occur much earlier. Although this model, and its underlying datasets, does not allow us to robustly predict penetration rates decades into the future (and in any case, technology adoption prediction of this sort is error-prone), based upon this model if these regions are ever going to reach universal access it will not be for a very long time.

We are already experiencing slowing Internet access growth rates, particularly among low and middle-income countries. This is particularly salient given that it is in these countries that the next 50% reside, and points to the need for everyone—policymakers, engineers, companies, and civil-society—to carefully assess current strategies if we wish to reach universal access. These offline groups are more likely to be women or older adults, with lower levels of education and employment,

who live in rural communities—so, specific policies to address not only these countries but also particular groups within them will be necessary to speed up adoption.

## Conclusion

This column’s model suggests we will fall short of the 2020 and 2025 U.N. targets. As shown in our projections, low and middle-income countries have yet to enter a rapid growth phase and may be at risk of moving into a slow growth phase well before approaching a point of universal access. This is worrying since 50% of the world that is still offline is mostly from these countries. If we adopt the U.N. position that the Internet is a human right, then this access gap is profoundly significant.

Reaching a world of broad universal access will require a combination of innovative policies, technologies, and new business models. Policies will need to address the specific reasons why some groups (for instance, women) are less likely to use the Internet. Service providers will need to adopt business models that can accommodate the needs of low-income and rural populations. And while innovative new technologies (such as global low-orbit satellites) look promising, we need to make sure that the right regulations are in place to ensure the greatest benefit for those who are not yet connected. Finally, we need improved and more transparent data sources and methodologies on Internet access and use, which can help overcome today’s access gap and enable the realization of policy goals. ■

## References

1. Andrés, L. et al. The diffusion of the Internet: A cross-country analysis. *Telecommunications Policy* 34, 5-6 (2010), 323-340.
2. Hargittai, E. Weaving the Western Web: Explaining differences in Internet connectivity among OECD countries. *Telecommunications Policy* 23, 1999, 701-718.
3. Rogers, E.M. *Diffusion of Innovations*, 5<sup>th</sup> Edition. Free Press, New York, 2003.
4. Wolcott, P. et al. A framework for assessing the global diffusion of the internet. *Journal of the Association for Information Systems* 2, 1 (2001).

**Carlos Iglesias** (carlos.iglesias@webfoundation.org) is a Senior Research Manager, Web Foundation, in Oviedo, AS, Spain.

**Dhanaraj Thakur** (dhanaraj.thakur@webfoundation.org) is Research Director, Web Foundation, in Washington, DC, USA.

**Michael L. Best** (mikeb@gatech.edu) is an Associate Professor of interactive computing and international affairs at the Georgia Institute of Technology, Atlanta, GA, USA.

Copyright held by author.

<sup>d</sup> Defining universal access is part of a complex ongoing discussion that is out of the scope of this column. For this analysis we take the population of five years and above as potential Internet users (based on UN population data; see <http://bit.ly/348ypBS>). Moreover, the ITU data suggests that even in high-income countries Internet penetration rates have plateaued between 80%-90% (which is understandable as some groups in the population may not be able to or interested in access). Thus, we take 85% as a reasonable and conservative reference for universal access.

► Peter G. Neumann, Column Editor

## Inside Risks

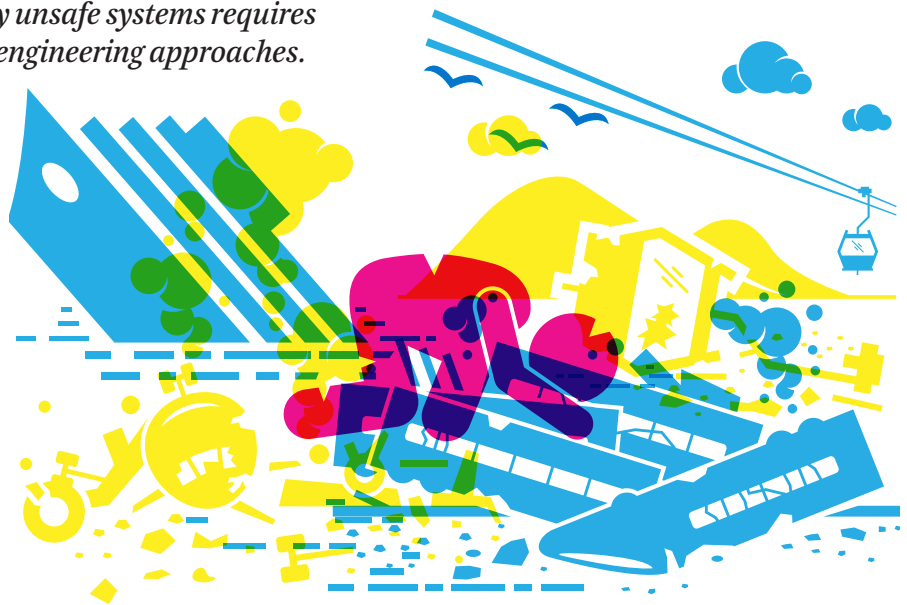
# Are You Sure Your Software Will Not Kill Anyone?

*Using software to control potentially unsafe systems requires the use of new software and system engineering approaches.*

**F**ROM WHAT I have seen, heard, and read, confusion and misinformation about software and safety. I have worked in this area for nearly 40 years, starting around the time when computers were beginning to be introduced into the control of safety-critical systems. I want to share what I have learned. Too many incorrect beliefs are being promoted, which are inhibiting progress and, in some cases, unnecessarily costing lives. This column clarifies this topic so that the solutions we propose are more likely to have a significant impact on safety.

With only a few exceptions, software was not used to directly control safety-critical systems until approximately 1980, although it was used to provide computational power for complex systems, such as spacecraft. Direct control was very limited, but the hesitation has now almost completely disappeared and software is used to control most systems, including physical systems that could involve potentially large and even catastrophic losses.

Originally, “embedded software” was used to denote these new control roles for software, but more recently the term “cyber-physical systems” has come into vogue. The figure here shows a standard cyber-physical control loop. Note that, for some reason, cyber-physical systems usually forget that control can be, and often is, provided by humans. In a little more realistic model



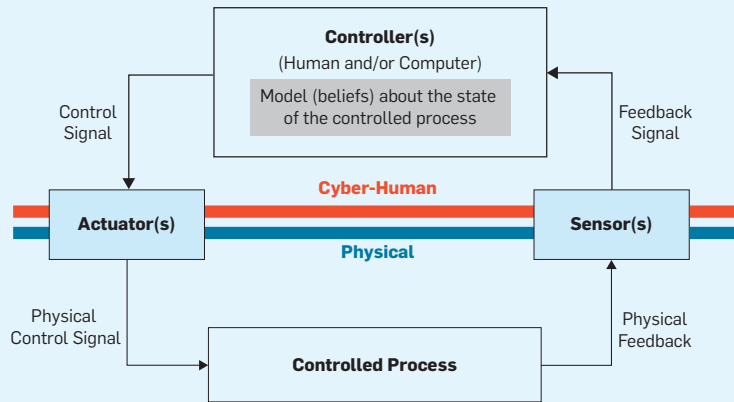
(but more complicated than necessary here), there would be two controllers where a human controller is providing control signals to a computer controller. To cover more than the unusual case where there are no human controllers, we should actually talk about “cyber-human-physical” systems. Even so-called “unmanned” air vehicles, for example, usually have a human controller on the ground. A more realistic and complete model is provided in Appendix G of the *STPA Handbook*.<sup>4</sup>

As illustrated in the figure here, a controller (or controllers, which may be human, automated or both) compares the current state of the controlled process with the control goals and sends control signals to an actuator, which in turn may be automated or human. The actuators translate the control signals

into physical actions on the controlled process. Sensors provide feedback about the state of the controlled process to the controller so it can determine the state of the controlled system and decide whether further control signals are needed. The actuators and sensors may be software, hardware, physical devices, or humans.

In order to decide on what control actions to provide in order to satisfy its goals (requirements), the controller must have a model (often called a mental model when the controller is human) of the current state of the controlled process. The most common cause of accidents stemming from unsafe controller action is that the model of the controlled process is incorrect: the pilot thinks the aircraft is not in a stall when it is and does not issue a

A cyber-human-physical control loop.



required control action to escape from the stall, the driver does not see the pedestrian and does not brake in time to prevent a collision, the weapon controller thinks that friendly troops are the enemy and initiates friendly fire. The pilot, driver, and weapon controller can be human or computerized, or a combination of both.

Accidents involving computers (and humans) most often occur when their models of the current state of the controller do not match the actual state of the controlled process; the controller issues a control action that is appropriate for a different state but not the one that currently exists. As an example, the software controller thinks the aircraft is in a stall when it is not and issues a control action to escape the non-existent stall only to inadvertently put the aircraft into a dangerous state.

Starting from this foundation, let's consider some of the most common misconceptions with respect to software and safety.

#### **Misconception 1: Software Itself Can Be Unsafe**

Software cannot catch on fire or explode; it is an abstraction. Only physical entities can inflict damage to life and property: physical energy is usually required to inflict physical harm. In the figure in this column, software sends control signals to a physical process, which may have physical effects. Nuclear power plants can release radiation, chemical plants can release toxins, weapon systems can explode or inadvertently target a friendly object, for example. One old model of an accident describes it as uncontrolled energy.

Software does not release energy; it simply releases bits, which can be used to send a control signal.

To avoid misconceptions that arise from the term “software safety,” sometimes safety engineers speak of “software system safety,” to denote the contribution of software behavior to a dangerous process. An alternative conception is to speak of the contribution of software to *system safety*. Either way, by considering software in isolation, without including the controlled physical process, it is not possible to assure anything about the safety of the system the software is controlling.

The Ariane 4 software Inertial Reference System was perfectly safe in that launcher. However, when reused in the Ariane 5, it led to an explosion and loss of a satellite. Many accidents involve reused software.<sup>3</sup> It is not the software that is unsafe, but the entire system controlled by the software.

#### **Misconception 2: Reliable Systems Are Safe; That Is, Reliability and Safety Are Essentially the Same Thing. Reliability Assessment Can Therefore Act as a Proxy for Safety**

Reliability and safety are different system properties and sometimes even conflicting. This is true also with respect to the contribution of software to accidents. System components (including software) can operate 100% reliably and accidents may still result, usually from unsafe interactions among the system components. In addition, the larger environment (including social policies and decision making) beyond the system boundaries is important.

As a simple, real-world example, consider going out to the middle of a large deserted area, pointing a gun away from oneself, and firing. If there is nobody or nothing in the vicinity, the gun could be considered to be both reliable and safe. Consider, however, doing the same thing in a crowded mall. The gun has not changed, the gun's reliability has not changed, and the action (pulling the trigger) has not changed. But the safety certainly has.

The accompanying sidebar highlights three examples out of hundreds of similar losses.<sup>4</sup> Considering reliability only at the system level (instead of the component level) does not help. Complex systems almost always have many requirements (or goals) while there are constraints on how those goals can be achieved. As an example, a chemical plant may very reliably produce chemicals (the goal or mission of the plant) while at the same time polluting the environment around the plant. The plant may be highly reliable in producing chemicals but not safe. Most safety-critical systems have both mission (non-safety) requirements and safety constraints on how the mission or goals can be achieved. A “system failure” or inability to satisfy its requirements is not equivalent to a hazard or an accident. One exception is if safety is the only goal of the system; however, even for systems such as air traffic control, there are usually non-safety goals such as optimizing throughput in addition to the safety goals.

A common approach to assessing safety is to use probabilistic risk assessment to assess the reliability of the components and then to combine these values to obtain the system reliability. Besides the fact that this assessment ignores accidents that are caused by the interactions of “unfailed” components (see Misconception 3), most of these assessments include only random hardware failures and assume independence between the failures. Therefore, they provide anything close to a real safety assessment when the systems are just hardware and relatively simple. Such systems existed 50+ years ago when these probabilistic risk methods were developed; virtually all systems today (particularly complex ones) contain non-stochastic components including

software logic and humans making cognitively complex decisions.

We need to stop pretending that these probabilistic estimates of safety have anything to do with reality, and not base our confidence about safety on them. I have examined hundreds of accident reports in my 40 years in system safety engineering. Virtually every accident involved a system with a probabilistic risk assessment that showed the accident could/would not occur, usually exactly in the way it did happen.

**Misconception 3:  
The Safety of Components in a Complex System Is a Useful Concept; That Is, We Can Model or Analyze the Safety of Software in Isolation from the Entire System Design**

While the components of a more complex system can have hazards (states that can lead to some type of loss), these are usually not of great interest when the component is not the entire system of interest. For example, the valve in a car or an aircraft can have sharp edges that could potentially lead to abrasions or cuts to those handling it. But the more interesting hazards are always at the system level—the sharp corners on the valve do not impact the hazards involved in the role of the valve in the inadvertent release of nuclear radiation from a nuclear power plant or the release of noxious chemicals from a chemical plant (for example).

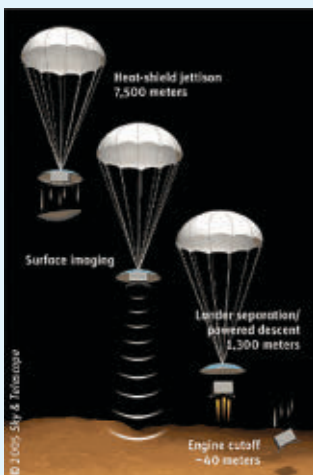
In other words, safety is primarily a system property and the hazards of interest are system-level hazards. The component's behavior can, of course, contribute to system hazards, but its contribution cannot be determined without considering the behavior of all the system components as a whole. Potentially effective approaches to safety engineering involve identifying the system hazards and then eliminating or, if that is not possible, preventing or mitigating them at the system level. The system hazards can usually be traced down to behavior of the system components, but the reverse is not true. One cannot show that each component is safe in isolation and then use that analysis to conclude the system as a whole will be safe.

Another way of saying this is that a system component failure is not

## Three Examples of Accidents Due to Unsafe Interactions between Systems Components



Some Navy aircraft were ferrying missiles from one point to another. One pilot executed a planned test by aiming at the aircraft in front (as he had been told to do) and firing a dummy missile. Apparently, nobody knew that the “smart” software was designed to substitute a different missile if the one that was commanded to be fired was not in a good position. In this case, there was an antenna between the dummy missile and the target, so the software decided to fire a live missile located in a different (better) position instead. What aircraft component(s) failed here?



This loss involved the Mars Polar Lander. It is necessary to slow the spacecraft down to land safely. Ways to do this include using the Martian atmosphere, a parachute and descent engines (controlled by software). As soon as the spacecraft lands, the software must immediately shut down the descent engines to avoid damage to the spacecraft. Some very sensitive sensors on the landing legs provide this information. But it turned out that noise (sensor signals) is generated when the legs are deployed. This expected behavior was not in the software requirements. Perhaps it was not included because the software was not supposed to be operating at this time, but the software engineers decided to start early to even out the load on the processor. The software thought the spacecraft had landed and shut down the descent engines while the spacecraft was still 40 meters about the planet surface. Which spacecraft components failed here?



It is dangerous for an aircraft's thrust reversers (which are used to slow the aircraft after it has touched down) to be activated when the aircraft is still in the air. Protection is designed into the software to prevent a human pilot from erroneously activating the thrust reversers when the aircraft is not on the ground. Without going into the details, some of the clues for the software to determine the plane has landed are weight on wheels and wheel spinning rate, which for a variety of reasons did not hold in this case. For example, the runway was very wet and the wheels hydroplaned. As a result, the pilots could not activate the thrust reversers and the aircraft ran off the end of the runway into a small hill. What aircraft components failed here?

equivalent to a hazard. Component failures can lead to system hazards, but a component failure is not necessary for a hazard to occur. In addition, even if a component failure occurs, it may not be able to contribute to a system hazard. This is simply another way of clarifying misconception #2 concerning the difference between reliability and safety.

**Misconception #4:  
Software Can Be Shown to Be Safe by Testing, Simulation, or Standard Formal Verification**

**Testing:** Exhaustive testing of software is impossible. The problem can be explained by examining what “exhaustive” might mean in the domain of software testing:

- Inputs: The domain of possible

inputs to a software system includes both valid and invalid inputs, potential time validity of inputs (an input may be valid at a certain time but not at other times), and all the possible sequences of inputs when the design includes history (which is almost all software). This domain is too large to cover any but a very small fraction of the possible inputs in a realistic timeframe.

► **System states:** Like the number of potential inputs, the number of states in these systems is enormous. For example, TCAS—an aircraft collision avoidance system—was estimated to have  $10^{40}$  possible states.<sup>5</sup> Note that collision avoidance is only one small part of the automation that will be required to implement autonomous (and even non-autonomous) vehicles.

► **Coverage of the software design:** Taking a simple measure of coverage like “all the paths through the software have been executed at least once during testing” involves enormous and impractical amounts of testing time and does not even guarantee correctness, let alone safety.

► **Execution environments:** In addition to the problems listed so far, the execution environment becomes significant when the software outputs are related to real-world states (the controlled process *and* its environment) that may change frequently, such as weather, temperature, altitude, pressure, and so on. The environment includes the social policies under which the system is used.

In addition, as seen in the much-repeated Dijkstra quote, testing can show only the presence of errors, not their absence.

Finally, and perhaps most important, even if we could exhaustively test the software, virtually all accidents involving software stem from unsafe requirements.<sup>2,6</sup> Testing can show only the consistency of the software with the requirements, not whether the requirements are flawed. While testing is important for any system, including software, it cannot be used as a measure or validation of acceptable safety. Moving this consistency analysis to a higher level (validation) only shifts the problem, but does not solve it.

**Simulation:** All simulation depends on assumptions about the environment in which the system will execute. Autonomous cars have now been sub-

## System and software requirements development are necessarily a system engineering problem, not a software engineering problem.

jected to billions of cases in simulators, and have still been involved in accidents as soon as they are used on real roads. The problems described for testing apply here, but the larger problem is that accidents occur when the assumptions used in development and in the simulation do not hold. Another way of saying this is that accidents occur because of what engineers call “unknown unknowns” in engineering design. We have no way to determine what the unknown unknowns are. Therefore, simulation can show only that we have handled the things we thought of, not the ones we did not think about, assumed were impossible, or unintentionally left out of the simulation environment.


**Formal verification:** Virtually all accidents involving software stem from unsafe requirements, not implementation errors. Of course, it is possible that errors in the implementation of safe requirements could lead to an accident; however, in the hundreds of software-related accidents I have seen over 40 years, none have involved erroneous implementation of correct, complete, and safe requirements. When I look at accidents where it is claimed the implemented software logic has led to the loss, I always find the software logic flaws stem from a lack of adequate requirements. Of the three examples shown in the sidebar in this column, none of these accidents (nor the hundreds of others that I have seen) would have been prevented using formal verification methods. Formal verification (or even formal validation) can show only the consistency of two formal models. Complete discrete mathematical models do not exist of

complex physical systems, that is, the controlled process shown in the figure in this column.

### Conclusion

All of this leads to the conclusion that the most effective approach to dealing with safety of computer-controlled systems is to focus on creating the safety-related requirements. System and software requirements development are necessarily a system engineering problem, not a software engineering problem. The solution is definitely not in building a software architecture (design) and generating the requirements later, as has been surprisingly suggested by some computer scientists.<sup>7</sup>

Some features of a potential solution can be described. It will likely involve using a model or definition of the system. Standard physical or logical connection models will not help. For most such models, analysis can identify only component failures. In some, it might be possible to identify component failures leading to hazards, but this is the easy part of the problem and omits software and humans. Also, to be most effective, the model should include controllers that are humans and organizations along with social controls. Most interesting systems today are sociotechnical.

Using a functional control model, analysis tools can be developed to analyze the safety of complex systems. Information on an approach that is being used successfully on the most complex systems being developed today can be found in *Engineering a Safer World*<sup>1</sup> and on the related website <http://psas.scripts.mit.edu/home/>. 

### References

1. Leveson, N.G. *Engineering a Safer World*. MIT Press, 2012.
2. Leveson, N.G. *Safeware: System Safety and Computers*. Addison-Wesley, 1995.
3. Leveson, N.G. The role of software in spacecraft accidents. *AIAA Journal of Spacecraft and Rockets* 41, 4 (July 2004).
4. Leveson, N.G. and Thomas, J.P. *STPA Handbook* (2018); <http://psas.scripts.mit.edu/home/materials/>
5. Leveson, N.G. et al. Requirements specification for process-control systems. *IEEE Transactions on Software Engineering SE-20*, 9 (Sept. 1994).
6. Lutz, R. Analyzing software requirements errors in safety-critical, embedded systems. In *Proceedings of the International Conference on Software Requirements*. IEEE (Jan. 1992).
7. National Research Council. *Software for Dependable Systems*, 2007.

**Nancy Leveson** (leveson@mit.edu) is a professor of Aeronautics and Astronautics at the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA.

Copyright held by author.



# Kode Vicious

## Numbers Are for Computers, Strings Are for Humans

*How and where software should translate data into a human-readable form.*

### Dear KV,

While debugging a set of networked systems that seemed to be executing operations out of the expected order, I discovered an interesting feature of the protocol used by the application. The system is fairly old, and I was not involved in its creation, but I was asked to figure out why about 10% of the transactions were flagged as being in the wrong order. All the application communication happens using TCP. And since TCP guarantees the ordering of messages, I was confused as to how transactions between two systems could arrive out of order. What I found—by using Wireshark—was that the TCP stream was, as expected, in order, but the application protocol used on top of TCP had some rather odd properties.

In particular, all of the information, including time, was communicated as strings. The bug turned out to be an incorrect conversion of the time from a string to a value that could easily be compared. Although the messages arrived in the correct order, the system, reading the time, thought they were out of order and complained loudly. When I finally tracked down the developer who wrote the code, he said that he had used strings to make the protocol easier to debug and to make it easier for people looking at the log file to know what was happening in the system. My



feeling is that he got this concept the wrong way around, and I am wondering how you might feel about this, as you have written about in the past.

### Stung by Strings

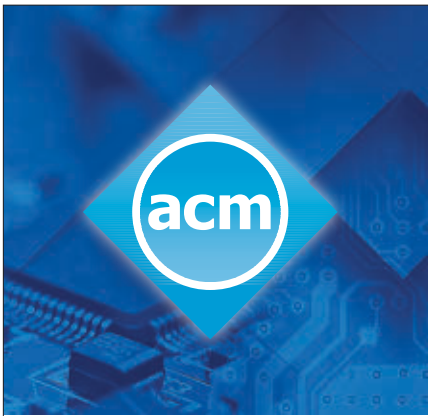
### Dear Stung,

How does this make me feel? Well, like nearly all questions around software and technology, it makes me angry, but then, what does not make me angry? When you play only one note, you might

as well play it really well, even if your instrument is a hammer and anvil.

As you point out, the developer definitely has this the wrong way around for at least two reasons: The first has to do with how one communicates a value as important as time; and the other has to do with how and where software should translate data into a human-readable form.

Computer systems generally, and networked systems specifically, often depend on time and time stamps to



## Advertise with ACM!

Reach the innovators and thought leaders working at the cutting edge of computing and information technology through ACM's magazines, websites and newsletters.



Request a media kit with specifications and pricing:

**Ilia Rodriguez**  
+1 212-626-0686  
acmm mediasales@acm.org



maintain or reconstruct the ordering of events, and this ordering often must be maintained so that the system as a whole can function properly. Some systems can operate without needing to know the time of day; instead, they depend on a total ordering of events, as established by Leslie Lamport—who ACM honored with a Turing Award for bothering to figure out that little problem (see [https://amturing.acm.org/award\\_winners/lamport\\_1205376.cfm](https://amturing.acm.org/award_winners/lamport_1205376.cfm)).

There remains a class of systems that actually do care about the very human time of day, such as credit-card processing. If your payment to the bank is not timestamped with the appropriate time of day (for example, midnight on the first day of the month), then you are going to be subject to interest and penalty payments that will likely make you even angrier than KV. Time also plays an important role in many non-banking security protocols. Get a comparison backward, or let it roll over to 0 or to a negative value, and the security of the system is broken. The number of attacks on systems based on time fills a large number of papers and books on computer security.

Computers, it should be pointed out, like to work with numbers. Strings are for humans, and so the idea of storing something as integral as time in a string is ridiculous on the face of it. The only way to know if X came before Y with a string-formatted time is to convert the times into something easily comparable (that is, integers) and then to have the computer do the comparison by math, which computers are very good at.

The number of times that a human will be looking at any of this data to make the same comparison is minuscule, which is the whole reason that pretty much all computer languages have a time data structure that can be both easily compared and that, hopefully, is resistant to misinterpretation. Not that we cannot get time wrong as an integer; we can, but it is far less likely than getting it wrong when stored as a string and converted before the math happens. Find your language's provided time type and use it, and check for errors on every comparison.

The second fallacy under which your software was written has become more common as computers have become

more powerful, and that fallacy is that compute power should always be used to give the human the best experience. When computers were slow and expensive, programmers were able to avoid dealing with the human question by pointing out that storing data in a compact numerical form resulted in better efficiency and use of an expensive resource. As computers became cheap and pervasive, many people pointed out that these efficiencies were no longer as strictly necessary as they used to be.

Those of us who continue to program near bare metal have never really let go of this cherished orthodoxy, but KV must, grudgingly, very grudgingly, admit that the other camp might have a point here. A few bytes here, a few instructions there, they do add up, but often saving them is not worth the effort, unless it leads either to incorrect results or to increased complexity, which usually leads to incorrect results (see the previous section of this column).

Unless what you are processing, storing, or transmitting are, quite literally, strings that come from and are meant to be shown to humans, you should avoid processing, storing, or transmitting that data as strings. Remember, numbers are for computers, strings are for humans. Let the computer do the work of presenting your data to the humans in a form they might find palatable. That is where those extra bytes and instructions should be spent, not doing the inverse.

**KV**

### Related articles on [queue.acm.org](https://queue.acm.org)

#### Time Is an Illusion

George Neville-Neil

<https://queue.acm.org/detail.cfm?id=2878574>

#### Time, but Faster

Theo Schlossnagle, Circonus

<https://queue.acm.org/detail.cfm?id=3036398>

#### There Is No Now

Justin Sheehy

<https://queue.acm.org/detail.cfm?id=2745385>

**George V. Neville-Neil** ([kv@acm.org](mailto: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.



## Viewpoint

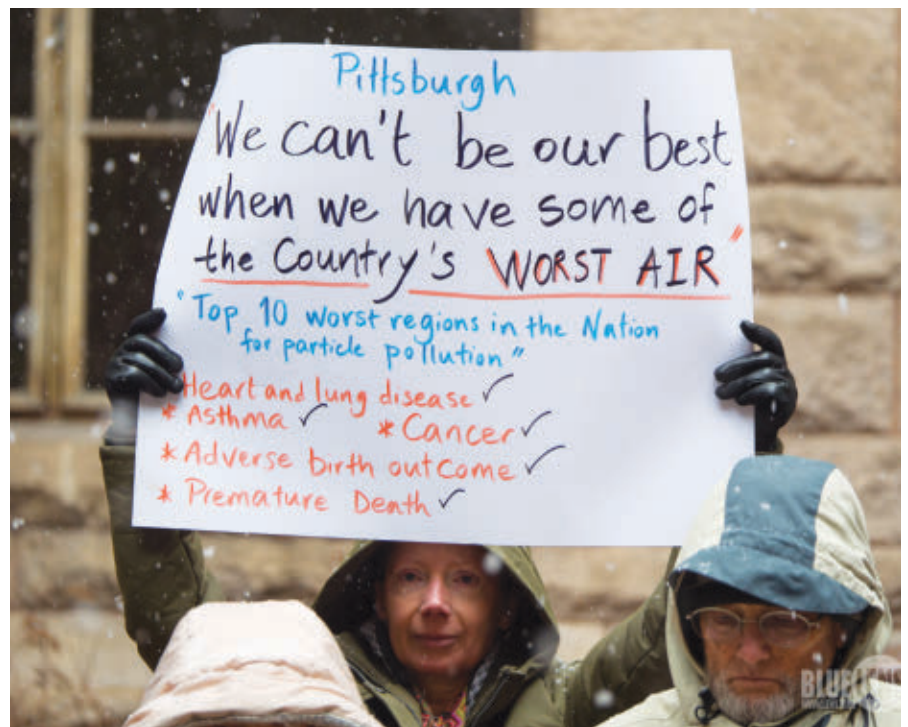
# When Human-Computer Interaction Meets Community Citizen Science

*Empowering communities through citizen science.*

**H**UMAN-COMPUTER INTERACTION (HCI) studies the design and use of interfaces and interactive systems. HCI has been adopted successfully in modern commercial products. Recently, its use for promoting social good and pursuing sustainability—known as sustainable HCI—has begun to receive wide attention.<sup>4</sup> Conventionally, scientists and decision-makers apply top-down approaches to lead research activities that engage lay people in facilitating sustainability, such as saving energy. We introduce an alternative framework, Community Citizen Science (CCS), to closely connect research and social issues by empowering communities to produce scientific knowledge, represent their needs, address their concerns, and advocate for impact. CCS advances the current science-oriented concept to a deeper level that aims to sustain community engagement when researchers are no longer involved after the intervention of interactive systems.

### Defining Community Citizen Science

Citizen science refers to the framework that empowers amateurs and professionals to form partnerships and produce scientific knowledge jointly. A major science-oriented strand aims to facilitate scientific research and address large-scale problems that are in-



Participants in a Clean Air Rally in Pittsburgh, PA, in 2018.

feasible for experts to tackle alone.<sup>11</sup> In this strand, professionals lead projects, define the goals, and encourage the public to participate in scientific research. One example is Galaxy Zoo (see <https://galaxyzoo.org>), which uses the knowledge collected from volunteers to classify a large number of galaxies online. Another example, eBird (see <https://ebird.org>), is an online crowdsourcing platform that engages

birdwatchers to contribute bird data collaboratively. Projects in this strand, such as Galaxy Zoo and eBird, are typically designed to answer scientific research questions and increase public understanding of science.

Community Citizen Science is a particular case of citizen science that embraces participatory democracy, community co-design, and power rebalance. These characteristics corre-

The left image (a) shows the user interface of our prior work, a community-empowered air-quality monitoring system.<sup>6</sup> The right image (b) shows the user interface of our other prior work, Smell Pittsburgh.<sup>7</sup>



(a)



(b)

spond to three different issues: core value, participation model, and governance structure. Depending on who defines the research question, citizen science projects can have different scientific, educational, social, environmental, and political values.<sup>11</sup> These projects can make use of diverse participation models between scientists and citizens, ranging from crowdsourcing to co-creation.<sup>5</sup> Citizen science can also apply different governance structures to connect stakeholders, ranging from top-down to bottom-up.<sup>2</sup>

### Participatory Democracy

CCS embraces participatory democracy to influence policymaking and address local concerns that community members wish to advocate for themselves. This community-oriented strand seeks to generate scientific evidence from community data to support exploration, understanding, and dissemination of local concerns.<sup>8</sup> In CCS, community members frame the main research question, and scientists engage in local issues that are raised by communities. For example, the Bucket Brigades project, pioneered by Global Community Monitor, (see <https://gcmmonitor.org>) provides low-cost devices that enable affected residents to collect air samples, send these samples to laboratories for analysis, measure the impact of local industrial pollution, and launch advocacy efforts. When connected to CCS, sustainable HCI extends scientific research into community empowerment, exploring how to use technology to strengthen the link between science and civil society.

### Community Co-Design

CCS embraces community co-design to develop interactive systems with advocacy groups, who are deeply grounded in local cultures and can bring diverse expertise to inform the design and use of computational tools.<sup>3</sup> In this way, CCS intends to rebalance technological privilege and develop a shared understanding of how technology is embodied in context, which brings community members and scientists into parity. Previous work has shown that a strong partnership among scientists and citizens has great potential to prompt decision making and produce policy changes.<sup>11</sup> For instance, The Community-Driven Environmental Project co-designed its technology platform, NatureNet, with naturalists and community members to successfully support local watershed management, such as engaging residents in installing rain barrels.<sup>9</sup> When connected to CCS, sustainable HCI further explores how researchers take on the role of supporters that facilitate utilizing and disseminating technology, instead of supervisors that oversee and control the entire community engagement procedure.

### Power Rebalance

CCS aims to rebalance power by using a bottom-up and multiparty structure, where local communities play significant roles in initiating grassroots movements, providing organizational networking, and disseminating critical findings to influence policymaking. CCS is especially beneficial when

lay perspectives contradict professional ones, and thus activism is needed to inform decision-makers about the perceptions of community concerns. In this way, CCS promotes ongoing political discourse around local concerns to improve the conditions of society. For instance, in a study of childhood leukemia cases that were clustered near contaminated water wells in Woburn, MA, residents recruited epidemiologists to show the relationship between the risk of childhood leukemia and the hazardous chemicals in their drinking water.<sup>1</sup> When connected to CCS, sustainable HCI is extended to exploring how technology can empower citizens to produce scientific evidence and rebalance power relationships among stakeholders.

### Architecting Interactive Systems as Technology Infrastructure

Designing interactive systems to support CCS suffers from the dilemma of Wicked Problems.<sup>10</sup> These problems have no precise definition, cannot be fully observed at the beginning, are unique and depend on context, have no opportunities for trial and error, and have no optimal or provably correct solutions. While researchers intend to enable citizens to generate scientific evidence and express their concerns with interactive systems, they are unable to accurately predict if citizens will contribute sufficient data to draw meaningful insights. It is also difficult to determine the critical amount of human effort, time, and the geographical scale required

for extracting reliable knowledge. Moreover, there are various methods of collecting, presenting, analyzing, and using the data. It is not feasible to explore and evaluate all possible methods without deploying the system in a real-world context. Furthermore, each context requires customized community outreach strategies due to different power relationships among stakeholders. These challenges, combined with other social conditions, make it difficult to integrate modern sensing devices and computational tools to support sustainability and community empowerment.

To tackle Wicked Problems, we propose an approach inspired by the design process used in architecture and urban planning. When approaching Wicked Problems on community or city scales, architects and urban planners design physical infrastructure based on prior empirical experiences to sustain human activities without their continuous supervision. This mind-set treats interactive systems as an ongoing technical infrastructure that sustains communities over time, even when the researchers are no longer present. For instance, in Civic Technoscience, citizens use open source tools to conduct scientific research, raise awareness of local issues, and influence policymaking.<sup>12</sup> The Balloon Mapping project, pioneered by Public Lab (see <https://publiclab.org>) provides low-cost technology for communities to create high-resolution landscape imagery with various applications, such as documenting protest events and evaluating the effectiveness of bioswales in absorbing pollutants. In this case, the Balloon Mapping tool became the technology infrastructure that supports community activism without extensive ongoing expert assistance following deployment.

### Evaluating the Impact of Interactive Systems

When evaluating interactive systems after deployment, we believe it is more beneficial to ask “Is the system influential?” instead of “Is the system useful?” Merely focusing on usability testing metrics, such as the time of completing tasks, may restrict the perspective of system design. Metrics

**CSS is by nature not replicable since each community has distinct characteristics and power relationships.**

of impacts, such as changes of attitude and behavior, can be useful proxies for evaluating the effectiveness of technology interventions.

However, due to the dilemma of Wicked Problems, it is extremely challenging to statistically verify whether the interactive system truly empowers communities and causes attitude or behavior changes. Unlike observational studies, CCS applies technology to simultaneously produce scientific knowledge and influence community members. If researchers frame the question of identifying the causal relationships as an observational study, it is difficult to track and control confounding factors that may influence their behaviors and attitudes, such as the effect of news and social media.

Alternatively, evaluating the intervention of technology with a randomized experiment can suffer from scientific and ethical concerns. It is not practical to randomly sample a control group with sufficient size from affected residents, since the information can spread among communities. Even if there is a way to prevent the control group from accessing the information about the deployed system, it is not appropriately ethical and contradicts the value of democratizing scientific knowledge. One may further consider an ethical way to compare the changes in the targeting community with another independent one that shares similar concerns but does not have access to the tool at the beginning. Nevertheless, CCS is by nature not replicable since each community has distinct characteristics and power relationships. The results obtained by conducting a ran-

## Calendar of Events

**Feb. 7–8**

AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA  
Contact: Toby Walsh,  
Email: [toby.walsh@nicta.com.au](mailto:toby.walsh@nicta.com.au)

**Feb. 9–12**

TEI '20: 14<sup>th</sup> International Conference on Tangible, Embedded, and Embodied Interaction, Sydney, Australia  
Sponsored: ACM/SIG,  
Contact: Lian Loke,  
Email: [lian.loke@sydney.edu.au](mailto:lian.loke@sydney.edu.au)

**Feb. 22–23**

CC '20: International Conference on Compiler Construction, San Diego, CA,  
Sponsored: ACM/SIG,  
Contact: Louis-Noel Pouchet,  
Email: [pouchet@cse.ohio-state.edu](mailto:pouchet@cse.ohio-state.edu)

**Feb. 22–26**

CGO '20: 18<sup>th</sup> Annual IEEE/ACM International Symposium on Code Generation and Optimization, San Diego, CA, USA  
Sponsored: ACM/SIG,  
Contact: Lingjia Tang,  
Email: [lingjia@umich.edu](mailto:lingjia@umich.edu)

**Feb. 22–26**

PPoPP '20: 25<sup>th</sup> ACM SIGPLAN Symposium on Principles and Practice on Parallel Programming, San Diego, CA, USA  
Sponsored: ACM/SIG,  
Contact: Rajiv Gupta,  
Email: [gupta@cs.ucr.edu](mailto:gupta@cs.ucr.edu)

**Feb. 23–25**

FPGA '20: The 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Seaside, CA, USA  
Sponsored: ACM/SIG,  
Contact: Stephen Neuendorffer,  
Email: [stephen@neuendorffer.name](mailto:stephen@neuendorffer.name)

**March**

**Mar. 3–4**

HotMobile '20: The 21<sup>st</sup> International Workshop on Mobil Computing Systems and Applications, Austin, USA,  
Sponsored: ACM/SIG,  
Contact: Padmanabhan Pillai,  
Email: [pillai@umich.edu](mailto:pillai@umich.edu)

domized experiment on two independent communities can be misleading.

Although it is difficult to statistically and rigorously validate the impact of systems on communities, understanding “How can the system be influential?” and “Does the community think that the system is influential?” can be beneficial. Through qualitative and quantitative analysis, the evaluation of impact can provide valuable insights to inform system design for the HCI community. For instance, our work, a community-empowered air-quality monitoring system (see the left image in the figure) enabled affected residents to present evidence of local air pollution for social activism, including smell reports, sensor data, and videos from monitoring cameras.<sup>6</sup> It studied how community members used animated smoke images and found that both manual and automatic approaches for generating images are essential during the engagement life cycle. The data provided by the system, combined with personal stories from affected residents, urged regulators to respond to the air pollution problem during a public community meeting.<sup>a</sup> Despite the small sample size in the analysis of self-efficacy and sense of community, the survey study found that the capability of using data-driven evidence from multiple perspectives is an important reason that the communities felt more confident after interacting with the system.

Our other work—Smell Pittsburgh (see <https://smellpgh.org>)—is a mobile application that enables residents to report pollution odors and track where these odors are frequently concentrated.<sup>7</sup> From our previous work, we found that smell experiences were valuable in representing how local air pollution affected the living quality of communities. In this Viewpoint, we translate this lesson from a hyperlocal to a citywide scale and provide insight into how smartphones, sensors, and statistical methods can be used to support CCS. The data analysis showed that events of poor smell were related to a joint

## Lessons that are learned from hyperlocal-scale projects may be translated to large-scale ones.

effect of wind information and hydrogen sulfide. A survey study found that motivations for community members to use Smell Pittsburgh came mainly from internal factors, including the desire to contribute data-driven evidence, concern about the welfare of others, and the ability to validate personal experiences using the visualization. This result was reinforced by the quantitative analysis of system usage, which identified a moderate association between contributing data and interacting with the visualization. The reports collected via Smell Pittsburgh were printed and presented by the community at the Board of Health meeting with the local health department, which urged the regulator to enact rigorous rules for petroleum coke plants.<sup>b</sup>

### Next Steps

Community Citizen Science aims to empower everyday citizens and scientists to represent their voices, reveal local concerns, and shape more equitable power relationships. Lessons that are learned from hyperlocal scale projects may be translated to large scale ones. However, when conducting CCS research, it is essential to acknowledge that replicating successful experiences and “parachuting” technology intervention without thoughtful consideration can cause irreversible harm to communities. Developing interactive systems to support CCS requires training of multidisciplinary skills, including system development, interaction design, data ana-

lytics, community engagement, and research methods. To connect science tightly to local concerns, we call for establishing CCS as a formal research field and integrating both computational and design thinking skills into curricula, which involves understanding local concerns through community fieldwork, forming the research question, co-designing technology infrastructure with communities, and evaluating the social impact after system deployment. In this way, we go beyond the mind-set of “citizens as scientists” to “scientists as citizens.” We envision that CCS can drive sustainable HCI toward citizen empowerment at a time when community concerns, sustainability issues, and technological ethics are at the forefront of global social discourse. ■

### References

1. Brown, P. Popular epidemiology and toxic waste contamination: Lay and professional ways of knowing. *Journal of Health and Social Behavior* (1992), 267–281.
2. Conrad, C.C. and Hilchey, K.G. A review of citizen science and community-based environmental monitoring: issues and opportunities. *Environmental Monitoring and Assessment* 176 (1–4), (2011), 273–291.
3. David, S., et al. Co-design with communities. A reflection on the literature. In *Proceedings of the 7th International Development Informatics Association Conference (IDIA)* (2013), 152–166.
4. DiSalvo, C., et al. Mapping the landscape of sustainable HCI. In *Proceedings of the 2010 CHI Conference on Human Factors in Computing Systems* (ACM, 2010), 1975–1984.
5. Haklay, M. Citizen science and volunteered geographic information: Overview and typology of participation. In *Crowdsourcing Geographic Knowledge*. Springer, Dordrecht 2013, 105–122.
6. Hsu, Y.C., et al. Community-empowered air quality monitoring system. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, (2017).
7. Hsu, Y.C., et al. Smell Pittsburgh: Community-empowered mobile smell reporting system. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. ACM 2019.
8. Irwin, A. *Citizen Science: A Study of People, Expertise and Sustainable Development*. Routledge, 2002.
9. Preece, J. et al. Interaction design of community-driven environmental projects (CDEPs): A case study from the Anacostia Watershed. In *Proceedings of the National Academy of Sciences* 116, 6 (June 2019), 1886–1893.
10. Rittel, H.W. and Webber, M.M. Dilemmas in a general theory of planning. *Policy Sciences* 4, 2 (Feb. 1973), 155–169.
11. Shirik, J.L. et al. Public participation in scientific research: A framework for deliberate design. *Ecology and Society* 17, 2 (Feb. 2012).
12. Wylie, S.A. et al. Institutions for civic technoscience: How critical making is transforming environmental research. *The Information Society* 30, 2 (2014), 116–126.

**Yen-Chia Hsu** (yenchiah@andrew.cmu.edu) is a Project Scientist at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA.

**Iliah Nourbakhsh** (illah@andrew.cmu.edu) is K&L Gates Professor of Ethics and Computational Technologies at The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA.

a Regulators reviewing Shenango Coke Works’ compliance with 2012 consent decree: <http://bit.ly/34TWZ9L>

b Allegheny County Health Department defends air quality efforts, plans stricter coke plant rules: <http://bit.ly/361wKQt>

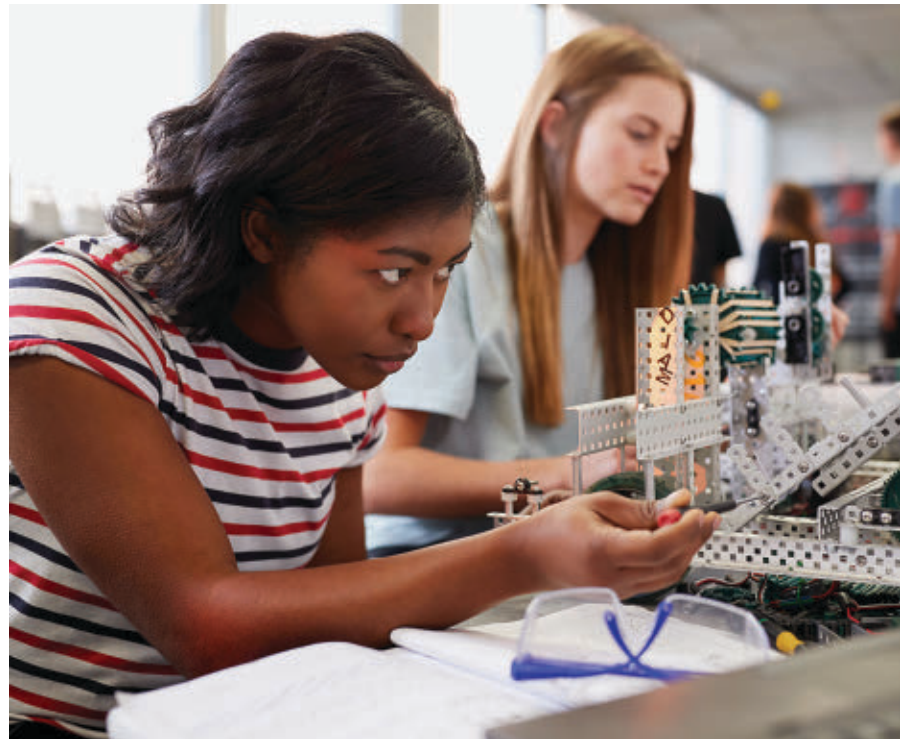
## Viewpoint

# Guiding Students to Develop Essential Skills

*Students should interact with one another to practice skills and construct their own understanding, with assistance from a teacher acting as a coach and guide—not a lecturer.*

**R**ECENT SURVEYS<sup>4,10</sup> INDICATE that employers seek candidates with broad “professional” skills in communication, teamwork, information processing, critical thinking, and problem solving. At the same time, faculty often feel pressure to “cover” more course content and feel there is not enough time for students to learn key concepts and also develop these skills. Faculty also feel students are disengaged, do not ask or answer questions, and struggle to think critically and solve problems. What more could faculty do to help students engage in the classroom, master content, and develop important skills?

Imagine a classroom where student teams actively collaborate to process information; think critically about problems; identify and evaluate possible solutions; and share insights and questions with each other. This starts on the first day of my first CS course, when students with no programming experience use a number-guessing game to develop key ideas in algorithm complexity analysis (see description in Simon et al.<sup>11</sup>). This continues through advanced courses such as artificial intelligence and programming languages. A graduate noted this approach “actually prepared me with hands-on experience in performing tasks that I was immediately required to complete when I began my job. The practice ... gave me a noticeable edge during my internship that led to me being hired.”



The ICAP (Interactive, Constructive, Active, and Passive) framework<sup>1</sup> describes how learning outcomes improve as students progress from passive to active to constructive (generate understanding) to interactive (discuss with peers). In recent decades, faculty have developed instructional approaches that build on educational research and help students to interact, construct understanding, and develop important skills. These approaches include Case-Based Learning, Pair Programming, Peer

Instruction, Peer-Led Team Learning, Problem-Based Learning, Project-Based Learning, and Process Oriented Guided Inquiry Learning (POGIL) (for overviews, see Eberlein et al.<sup>2</sup> and Simon et al.<sup>11</sup>).

POGIL focuses on how students learn by emphasizing skills and the *process* of learning (the “process oriented” in POGIL) as well as specific concepts and content.<sup>9,12</sup> In a POGIL classroom, students interact in teams (typically 3–5) to work through an activity together, discuss each question, and ensure that team members

**Figure 1. Excerpt of Process Oriented Guided Inquiry Learning (POGIL) activity on Python style.**

*Learning Objectives:* After completing this activity, learners should be able to:

- Describe good conventions for variable names, parentheses, function names, spacing, indentation, etc.
- Explain the value and benefits of good programming style.

1. For each row in the table below, write X or Y in the last column to indicate which option is better. *Sample answers are shown in blue italics.*

Option X	Option Y	X/Y
<code>s1=i1*c1+i2*c2;</code>	<code>s1 = i1*c1 + i2*c2;</code>	<i>Y</i>
<code>s1=(i1*c1)+(i2*c2);</code>	<code>s1=i1*c1+i2*c2;</code>	<i>X</i>
<code>s1 = c1*i1 + i2*c2;</code>	<code>s1 = i1*c1 + i2*c2;</code>	<i>Y</i>
<code>total = nCD*sCD + (nCD*cCD + nMP3*cMP3) * (1+rateTax);</code>	<code>cost = nCD *cCD + nMP3*cMP3; ship = nCD*sCD; tax = cost * rateTax; total = cost + tax + ship;</code>	<i>Y</i>

2. Based on your answers above, summarize advice for writing expressions. (Write complete sentences, and be ready to report your answers to the class.)

*Use spacing & parentheses to be readable and clarify precedence.*

*Use consistent ordering of the terms in subexpressions.*

*Use subexpressions and extra variables instead of long complicated expressions.*

3. Code that can be read and understood without separate documentation is called **self-documenting code**. How can we make expressions self-documenting?

*same as above*

```
# calculate trip cost from 2-way flights, hotels, & meals
fc=500; ch=150; mc=30; nn=5 # costs & number nights
c = 2*fc+ch*n+nn*3*mc
```

4. Use your answers to previous questions to rewrite the code above.

understand and agree on each answer. Each team member has a role that focuses on specific skills, and the roles rotate each class to help each student practice all of the skills. For example, the manager keeps track of time and encourages all team members to participate, the recorder takes notes for the team on key concepts and insights, and the presenter shares the team's conclusions with other teams and the teacher. The instructor is not a lecturer, but an active facilitator, who observes teams, prompts members to focus on their roles, responds to questions that a team cannot resolve, and leads classroom discussion, usually based on key questions in the activity.

Each POGIL activity is specifically designed with models (for example, code, diagrams, graphs, tables), each followed by a sequence of questions to guide student to construct understand-

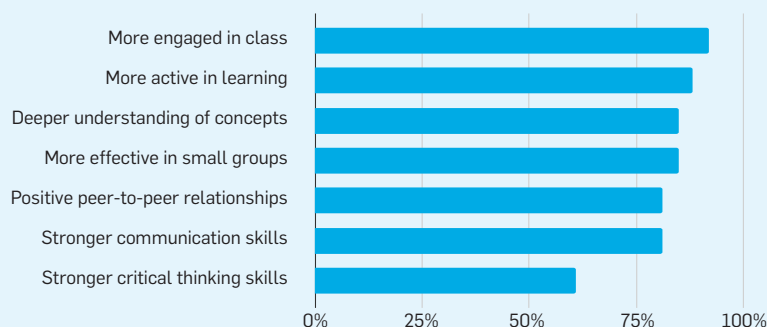
ing (the "guided inquiry" in POGIL). For example, Figure 1 shows part A of a CS1 activity on coding style in the Python language. (Sample answers are shown in blue italics.) Question 1 prompts the team to explore two code snippets (X and Y), notice differences, discuss, and decide which is better. For many students these answers are not obvious and so at first they might disagree, but after discussion most teams reach consensus. Question 2 prompts them to discuss and invent general insights and advice. Thus, each team will consider their answers to Q1, develop insights, and express them clearly. Question 3 introduces a new term ("self-documenting code") and prompts teams to connect it to their insights in Q2. Question 4 prompts teams to apply these insights to a new, longer code snippet. Again, team members might disagree at first, but will usually reach

consensus. Thus, as teams work on the activity, they process information, think critically and solve problems, and communicate with their team, the instructor, and the entire class.

Most POGIL activities have several of these Explore-Invent-Apply (EIA) learning cycles. For example, in the Python style activity, the EIA cycle repeats in part B with code snippets that involve function names, and in part C with comments. Similar structures are used for other syntax elements, and more complex examples, depending on the course and level of student experience. Other activities use longer code samples, and guide students to identify and correct defects, refactor and extend the code, or analyze algorithmic complexity. Activities can use many other models: a table of storage device size and access time (memory hierarchy); a recipe to bake cookies (project scheduling); normal magic squares and eight tile puzzles (search strategies); and equations (neural networks and genetic algorithms). As students gain skills and content knowledge, they are able to explore more complex models with less guidance, invent more complex ideas, and apply their insights to larger and more abstract problems.

While the teams work, the instructor walks around the classroom to observe, and ask questions such as "What differences do you see between the two snippets?", "Does everyone on your team agree with this answer?", or "Could you rewrite your insight as a complete sentence?". At the same time, the instructor learns what their students know (and don't know), and how to revise the activity to better guide student learning. After most teams have answered Q2, the instructor might ask a few teams to report their answers to the class, so all teams can check their work. Students who hesitate to share their own answers are often eager to share their team's answers. The instructor might ask a few teams to write their answers to Q4 on the board or share them electronically, and then moderate a short discussion of any differences in answers. After teams complete the activity, the instructor spends a few minutes to emphasize key ideas, and prompts teams to reflect on what they learned, how they worked as a

Figure 2. CS faculty (n=26) perceptions of POGIL effectiveness.



team, and how they might improve in the future. A homework assignment might prompt students (individually, in pairs, or in teams) to apply good coding style to their own code, or to an example provided by the instructor.

Research on POGIL in other disciplines (for example, Farrell<sup>3</sup>, Moog<sup>8</sup>) generally finds that, compared to traditional approaches, student attrition is lower, content mastery is greater, and students have more positive attitudes. In one study,<sup>13</sup> on the first day of Organic Chemistry II, students took an unannounced quiz on content from Organic I. Students from the POGIL section of Organic I did dramatically better than those from the lecture section, although the quiz was written by the lecture instructor. There are fewer studies of POGIL in computer science, but results are encouraging. After converting a CS1 course to POGIL, pass rates increased for females but not males.<sup>6</sup> In a software project course, POGIL activities helped students to understand the importance of communication in real software projects.<sup>7</sup> In a survey (summarized in Figure 2), CS POGIL instructors reported that students were more active and engaged, understood concepts better, and were better at teamwork, communication, and critical thinking.<sup>5</sup>

Thus, POGIL can both improve student learning outcomes and develop key skills. POGIL has been used across STEM disciplines in secondary and post-secondary settings. The POGIL Project (see <http://pogil.org>) reviews and endorses POGIL activities, and offers an extensive series of workshops on classroom facilitation, activity authoring, and related topics. The U.S. National Science Foundation has funded numerous projects to support POGIL,

including several focused on computing disciplines. To learn more about POGIL activities for a range of computing topics, see <http://cspogil.org>. □

#### References

- Chi, M.T.H. and Wylie, R. The ICAP framework: Linking cognitive engagement to active learning outcomes. *Educational Psychologist* 49, 4 (Oct. 2014), 219–243. DOI:<https://doi.org/10.1080/00461520.2014.965823>
- Eberlein, T. et al. Pedagogies of engagement in science: A comparison of PBL, POGIL, and PLTL. *Biochemistry and Molecular Biology Education* 36, 4 (2008) 262–273. <http://doi.org/10.1002/bmb.20204>
- Farrell, J.J. et al. A guided-inquiry general chemistry course. *Journal of Chemical Education* 76, 4 (1999), 570–574.
- Hart Research Associates. *Falling Short? College Learning and Career Success*. 2015.
- Hu, H.H. et al. Results from a survey of faculty adoption of Process Oriented Guided Inquiry Learning (POGIL) in Computer Science. In *Proceedings of the ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE)* (Arequipa, Peru, July 11–13, 2016) ACM. DOI:<https://doi.org/10.1145/2899415.2899471>
- Hu, H.H. and Shepherd, T.D. Using POGIL to help students learn to program. *ACM Transactions on Computing Education (ToCE)* 13, 3 (Aug. 2013), DOI:<https://doi.org/10.1145/2499947.2499950>
- Kumar, S. and Wallace, C. 2014. Instruction in software project communication through guided inquiry and reflection. In *Proceedings of the IEEE Frontiers in Education (FIE) Conference* (Madrid, Spain, Oct. 22–25, 2014). ASEE/IEEE. DOI:<https://doi.org/10.1109/FIE.2014.7044167>
- Moog, R.S. et al. POGIL: Process oriented guided inquiry learning. In *Chemist's Guide to Effective Teaching: Vol II*, N.J. Pienta, M.M. Cooper, and T.J. Greenbowe (Eds.). Prentice Hall, 90–107.
- Moog, R.S. and Spencer, J.N. POGIL: An overview. In *Process-Oriented Guided Inquiry Learning: ACS Symposium Series 994*, R.S. Moog and J.N. Spencer, Eds., American Chemical Society, 2008, 1–13.
- NACE. *Job Outlook 2018*. National Association of Colleges and Employers, Bethlehem, PA, 2018.
- Simon, B. et al. Students as teachers and communicators. In *The Cambridge Handbook of Computing Education Research*, S.A. Fincher and A.V. Robins, Eds. Cambridge University Press, 2019, 827–858.
- Simonson, S., Ed. *POGIL: An Introduction to Process Oriented Guided Inquiry Learning for Those Who Wish to Empower Learners*. Stylus Publishing, 2019.
- Straumanis, A. and Simons, E.A. A multi-institutional assessment of the use of POGIL in Organic Chemistry. In *Process Oriented Guided Inquiry Learning (POGIL)*, ACS Symposium Series 994, R.S. Moog and J.N. Spencer, Eds., American Chemical Society, (2008), 226–239.

Clif Kussmaul (clif@kussmaul.org) is Principal Consultant at Green Mango Associates, LLC, in Bethlehem, PA, USA.

Copyright held by author.

## Coming Next Month in COMMUNICATIONS

**Crowdsourcing Moral Machines**

**Toward Model-Driven Sustainability Evaluation**

**The BBC micro:bit—From the U.K. to the World**

**Spotify Guilds**

**Editing Self-Image**

**Securing the Boot Process**

**Above the Line, Below the Line**

**Conferences in the Era of Expensive Carbon**

**Unsafe at Any Level**

**Four Internets**

**Pivot Tracing: Dynamic Causal Monitoring for Distributed Systems**

**Plus the latest news about nanosheet transistors, how algorithms improve renewable energy, and translation devices.**

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

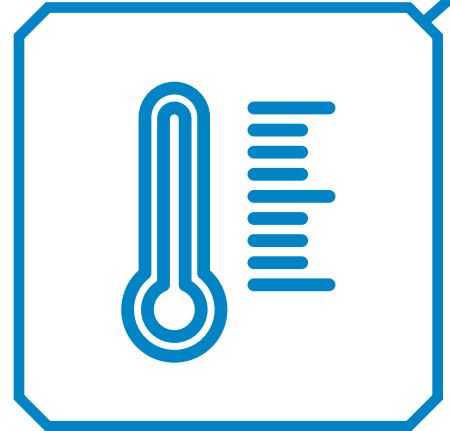
**If the CPU is the brain of the board,  
the BMC is the brain stem.**

BY JESSIE FRAZELLE

# Opening Up the Baseboard Management Controller

IN 2011, FACEBOOK announced the Open Compute Project to form a community around open source designs and specifications for datacenter hardware. Facebook shared its hardware specs, which resulted in 38% less energy consumption and 24% cost savings compared with its existing datacenters.<sup>6</sup> What Facebook and other hyperscalers (Google, Microsoft, et al.) donate to the Open Compute Project are their solutions to the agonizing problems that come with running datacenters at scale.

Since then, the project has expanded to all aspects of the open datacenter: baseboard management controllers (BMCs), network interface controllers (NICs), rack designs, power busbars, servers, storage, firmware, and security. This column focuses on the BMC. This is an introduction to a complicated topic;



some sections just touch the surface, but the intention is to provide a full picture of the world of the open source BMC ecosystem, starting with a brief overview of the BMC's role in a system, touching on security concerns around the BMC, and then diving into some of the projects that have developed in the open source ecosystem.

If the CPU is the brain of the board, the BMC is the brain stem. It monitors and manages the physical state of a computer or hardware device. This state includes temperature, humidity, power supply voltage, fan speeds, remote access, and operating system functions. The BMC has historically been a SuperH or ARM-based system on a chip (SoC) with common functionality including but not limited to:

- ▶ RMII (reduced media-independent interface) and RGMII (reduced gigabit media-independent interface) for ethernet.
- ▶ A boot flash with an SPI (serial peripheral interface) NOR. (NOR and NAND are types of nonvolatile flash memory; the difference is in the type of logic gate used.)
- ▶ PCI (peripheral component interconnect) express.
- ▶ An LPC (low pin count) bus for communicating with the host. Intel's successor to LPC is the eSPI (Enhanced Serial Peripheral Interface bus).

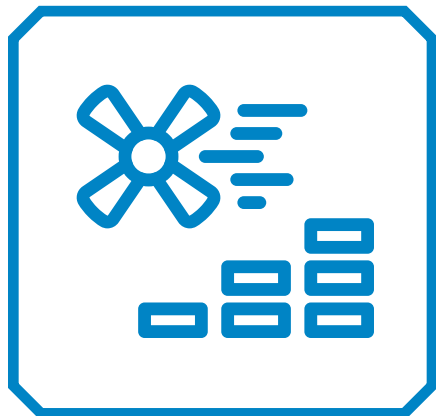
The BMC usually communicates to the outside world (or the datacenter control network) using the intelligent platform management interface (IPMI),



a message-based, hardware-level interface specification for managing and operating computer systems. It operates independently of the operating system, the server's CPU, and the firmware that allows admins to manage a system without an operating system or any system management software. Admins can also take advantage of IPMI's local network to get a console on a remote computer that is otherwise inaccessible.

### BMC Security Concerns

The IPMI stack was not designed with security in mind (the IPMI spec requires making the hash of a user's password available over the stack). The assumption was the datacenter control networks would be segregated and trusted, which is why IPMI is notorious for security vulnerabilities.<sup>2</sup> Exploits in the IPMI stack and the BMC are devastating because of the many privileged operations for which they are responsible. Improving IPMI security has historically been neglected, as most IPMI software is proprietary.



The BMC has its own problems with largely proprietary software and vulnerabilities. The most recent notable BMC vulnerability is USBAnywhere,<sup>3</sup> discovered by Rick Altherr, principal engineer at Eclipsium. On Supermicro servers, an attacker can use USBAnywhere to connect remotely to a server and virtually mount any USB device to the server. As a result, an attacker could load a new operating system image or implant a firmware backdoor to facili-

tate ongoing remote access. At the time of the disclosure, 47,000 vulnerable systems were found to be exposed to the public Internet. Another fun vulnerability is Pantsdown,<sup>1</sup> which allows read and write access to the BMC's address space from the host. Pantsdown is an example of a requested feature causing a vulnerability.

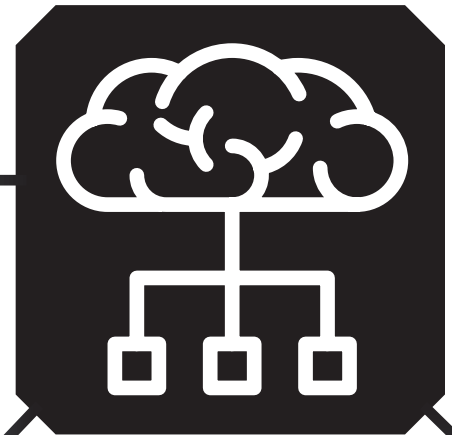
But wait, it gets worse. As Trammell Hudson pointed out in his Modchips of the State talk at the 35<sup>th</sup> Chaos Communication Congress in 2018,<sup>7</sup> the BMC often has access to the host firmware via serial peripheral interface (SPI) and to host memory through direct memory access (DMA). The BMC gets DMA access because it is on the peripheral component interconnect express (PCIe) bus as a device. This means it can inject code into the host's firmware. Much BMC firmware also lacks the notion of a secure boot. This makes the BMC a prime target for hackers. Here, I emphasize a point I made in a previous article on open source firmware:<sup>5</sup> It's an alarming problem that the code running with the most privilege has the least visibility and inspectability.

### The BMC Becomes Open Source

The trend toward open sourcing the datacenter has led to a number of innovative BMC projects.

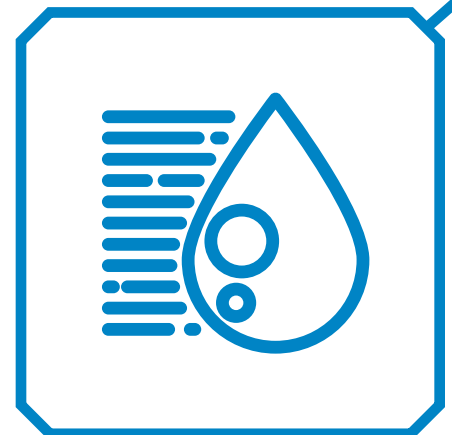
**OpenBMC.** In 2014, Facebook decided to solve the problems with proprietary BMC software by starting an open source BMC software project.<sup>4</sup> In 2015, IBM and Rackspace collaborated on solving the same problems with their own project.<sup>9</sup> Both projects were called OpenBMC and ended up merging into the OpenBMC project the firmware community is familiar with today (<https://github.com/openbmc/openbmc>). The founding organizations of the OpenBMC project, post-merger, were Microsoft, Intel, IBM, Google, and Facebook. OpenBMC has the widest range of support for various BMCs.

The OpenBMC project encompasses u-boot, an open source boot-



loader that boots a Linux kernel with a minimal root file system containing all the tools and binaries needed to run OpenBMC. OpenBMC is designed with a service-oriented approach. Services are started and maintained by systemd and communicate with each other over dbus. Designing for services makes sense as an easy way for multiple collaborators and vendors to contribute to a single BMC implementation. This allows each vendor contributing to the codebase to have separate daemons it can turn on to ship in its specific distribution of OpenBMC; however, it also makes the BMC software more complex to debug, audit, and put into production.

**U-bmc.** After OpenBMC came u-bmc (<https://github.com/u-root/u-bmc>), a software project started by Christian Svensson of Google. Written in Go, u-bmc aims for a more minimal BMC software architecture, challenging the status quo by replacing IPMI with gRPC. Removing IPMI makes u-bmc provocative from a security perspective since the attack surface area is reduced. Unlike OpenBMC, u-bmc boots a Linux kernel directly from the ASPEED startup code




after DRAM initialization, thus removing the need for a bootloader such as u-boot. As of the publication of this article, u-bmc supports BMCs based on the ASPEED AST2400 and AST2500, but plans to support more in the future and always welcomes contributions. If you have a Supermicro X11SSH board that supports coreboot, it is possible to use u-bmc as your BMC software.

**RunBMC.** Not only has software around the BMC been open sourced, but the hardware has as well. Eric Shobe and Jared Mednick of Dropbox analyzed all the BMC system topologies and their differences on a platform-by-platform basis. The result was RunBMC, a standard hardware interface for BMCs. Dropbox donated version 1 of the RunBMC hardware specs, along with two reference boards for the Nuvoton NPCM750R and ASPEED 2500 RunBMC modules, to the Open Compute Project in August 2019.<sup>8</sup>


The RunBMC design allows for swapping out BMCs separate from the rest of the board, isolating and locking down the BMC subsystem. Previous to this, the BMC was soldered onto the board. This is compelling from a security perspective since focus is shifted to a single, swappable BMC card, which can easily be replaced if broken, updated with a different version, or integrated with other security features. For example, a root of trust, the trusted source that verifies system software before execution, can secure I/O between the BMC card and the rest of the board. This also allows users to switch easily between the common BMC manufacturers, ASPEED, and Nuvoton. Interesting fact: Sun also had a BMC interconnect with its Integrated Lights Out Manager (iLOM), as did Dell with Dell Remote Access Controller (DRAC), HP with Integrated Lights-out (iLO), and IBM and Lenovo with integrated management module (IMM)—however, most do not ship this way today.

### Open Source Moving the Ecosystem Further

OpenBMC set the stage for BMC firmware and hardware to be open sourced. This spawned a series of other innovations being open sourced,




**OpenBMC set the stage for BMC firmware and hardware to be open sourced. This spawned a series of other innovations being open sourced, and more can be expected.**



and more can be expected. This space is turning out many awesome projects, and I am lucky to be able to shine a light on the amazing work being done. Open sourcing the software at the lowest levels of the stack provides visibility into the code running with the most privileges on systems. We can only hope this will lead to more eyes vetting the code, encourage more minimal architectures, and lessen the risk of systems being caught with their “Pantsdown” in the future.

### Acknowledgments

Thanks to the individuals in the open source ecosystem for helping me learn about their projects: Rick Alther, Chris Koch, Christian Svensson, Ron Minnich, Trammell Hudson, Eric Shobe, and Jared Mednick. If you are interested in helping with any of the projects mentioned here, check out GitHub. 

### Related articles on queue.acm.org

#### Security for the Modern Age

Jessie Frazelle

<https://queue.acm.org/detail.cfm?id=3301253>

#### Commercializing Open Source Software

Michael J. Karels

<https://queue.acm.org/detail.cfm?id=945125>

#### GNL is Not Linux

Kode Vicious

<https://queue.acm.org/detail.cfm?id=2909572>

### References

1. Common Vulnerabilities and Exposures. CVE-2019-6260; <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-6260>.
2. Common Vulnerabilities and Exposures. Intelligent Platform Management Interface; <https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=ipmi>.
3. Eclipsium. Virtual media vulnerability in BMC opens servers to remote attack; <http://bit.ly/2NSdX2b>
4. Fang, T. Introducing ‘OpenBMC.’ An open software framework for next-generation system management. Facebook Engineering, 2015; <http://bit.ly/2PHS73T>
5. Frazelle, J. Open-source firmware. *acmqueue* 17, 3 (2019); <https://queue.acm.org/detail.cfm?id=3349301>.
6. Heiliger, J. Building efficient data centers with the Open Compute Project. Facebook, 2011; <http://bit.ly/2NBDwEm>
7. Hudson, T. 2019. Modchips of the State; <https://trmm.net/Modchips#Defenses>.
8. Shobe, E., Mednick, J. RunBMC: OCP hardware spec solves data center BMC pain points; <https://blogs.dropbox.com/tech/2019/08/runbmc-ocp-hardware-spec-solves-data-center-bmc-pain-points/>
9. Sullivan, A. OpenPOWER & Open Compute: Full speed ahead with Barreleye, 2015; <https://blog.rackspace.com/openpower-open-compute-barreleye>.

Jessie Frazelle is an independent consultant. She previously worked on the Docker Core Team, Google, and Microsoft.

Copyright held by author/owner.

---

**A practical journey.**

---

**BY MATT GODBOLT**

---

# Optimizations in C++ Compilers

COMPILERS ARE A necessary technology to turn high-level, easier-to-write code into efficient machine code for computers to execute. Their sophistication at doing this is often overlooked. You may spend a lot of time carefully considering algorithms and fighting error messages but perhaps not enough time looking at what compilers are capable of doing.

This article introduces some compiler and code generation concepts, and then shines a torch over a few of the very impressive feats of transformation your compilers are doing for you, with some practical demonstrations of my favorite optimizations. I hope you will gain an appreciation for what kinds of optimizations you can expect your compiler to do for you, and how you might explore the subject further. Most of all, you may learn to love looking at the assembly output and may learn to respect the quality of the engineering in your compilers.

The examples shown here are in C or C++, which are the languages I have had the most experience with, but

many of these optimizations are also available in other compiled languages. Indeed, the advent of front-end-agnostic compiler toolkits such as LLVM<sup>3</sup> means most of these optimizations work in the exact same way in languages such as Rust, Swift, and D.

I have always been fascinated by what compilers are capable of. I spent a decade making video games where every CPU cycle counted in the war to get more sprites, explosions, or complicated scenes on the screen than our competitors. Writing custom assembly, and reading the compiler output to see what it was capable of, was par for the course.

Fast-forward five years and I was at a trading company, having switched out

Many optimizations fall under the umbrella of *strength reduction*: taking expensive operations and transforming them to use less expensive ones.

sprites and polygons for fast processing of financial data. Just as before, knowing what the compiler was doing with code helped inform the way we wrote the code.

Obviously, nicely written, testable code is extremely important—especially if that code has the potential to make thousands of financial transactions per second. Being fastest is great, but not having bugs is even more important.

In 2012, we were debating which of the new C++11 features could be adopted as part of the canon of acceptable coding practices. When every nanosecond counts, you want to be able to give advice to programmers about how best to write their code without being antagonistic to performance. While experimenting with how code uses new features such as `auto`, lambdas, and range-based `for`, I wrote a shell script (a) to run the compiler continuously and show its filtered output. This proved so useful in answering all these “what if?” questions that I went home that evening and created Compiler Explorer.<sup>1</sup>

Over the years I have been constantly amazed by the lengths to which compilers go in order to take our code and turn it into a work of assembly code art. I encourage all compiled language programmers to learn a little assembly in order to appreciate what their compilers are doing for them. Even if you cannot write it yourself, being able to read it is a useful skill.

All the assembly code shown here is for 64-bit x86 processors, as that is the CPU I’m most familiar with and is one of the most common server architectures. Some of the examples shown here are x86-specific, but in reality, many types of optimizations

apply similarly on other architectures. Additionally, I cover only the GCC and Clang compilers, but equally clever optimizations show up on compilers from Microsoft and Intel.

### Optimization 101

This is far from a deep dive into compiler optimizations, but some concepts used by compilers are useful to know. In these pages, you will note a running column of examples of scripts and instructions for the processes and operations discussed. All are linked by the corresponding (letter).

Many optimizations fall under the umbrella of *strength reduction*: taking expensive operations and transforming them to use less expensive ones. A very simple example of strength reduction would be taking a loop with a multiplication involving the loop counter, as shown in (b). Even on today’s CPUs, multiplication is a little slower than simpler arithmetic, so the compiler will rewrite that loop to be something like (c).

Here, strength reduction took a loop involving multiplication and turned it into a sequence of equivalent operations using only addition. There are many forms of strength reduction, more of which show up in the practical examples given later.

Another key optimization is *inlining*, in which the compiler replaces a call to a function with the body of that function. This removes the overhead of the call and often unlocks further optimizations, as the compiler can optimize the combined code as a single unit. You will see plenty of examples of this later.

Other optimization categories include:

► *Constant folding*. The compiler takes expressions whose values can be

```
$ g++ /tmp/test.cc -O2 -c -S -o - -masm=intel \
  | c++filt \
  | grep -vE '\s+\.'
```

(a)

```
for (int i = 0; i < 100; ++i)
{
    func(i * 1234);
}
```

(b)

```
for (int iTimes1234 = 0; iTimes1234 < 100 * iTimes1234 += 1234)
{
    func(iTimes1234);
}
```

(c)

calculated at compile time and replaces them with the result of the calculation directly.

► *Constant propagation.* The compiler tracks the provenance of values and takes advantage of knowing that certain values are constant for all possible executions.

► *Common subexpression elimination.* Duplicated calculations are rewritten to calculate once and duplicate the result.

► *Dead code removal.* After many of the other optimizations, there may be areas of the code that have no effect on the output, and these can be removed. This includes loads and stores whose values are unused, as well as entire functions and expressions.

► *Instruction selection.* This is not an optimization as such, but as the compiler takes its internal representation of the program and generates CPU instructions, it usually has a large set of equivalent instruction sequences from which to choose. Making the right choice requires the compiler to know a lot about the architecture of the processor it's targeting.

► *Loop invariant code movement.* The compiler recognizes that some expressions within a loop are constant for the duration of that loop and moves them outside of the loop. On top of this, the compiler is able to move a loop invariant conditional check out of a loop, and then duplicate the loop body twice: once if the condition is true, and once if it is false. This can lead to further optimizations.

► *Peephole optimizations.* The compiler takes short sequences of instructions and looks for local optimizations between those instructions.

► *Tail call removal.* A recursive function that ends in a call to itself can often be rewritten as a loop, reducing call overhead and reducing the chance of stack overflow.

The golden rule for helping the compiler optimize is to ensure it has as much information as possible to make the right optimization decisions. One source of information is your code: If the compiler can see more of your code, it's able to make better decisions. Another source of information is the compiler flags you use: telling your compiler the exact CPU architecture you are targeting

```
int count(const vector<int> &vec)
{
    int numPassed = 0;
    for (size_t i = 0; i < vec.size(); ++i)
    {
        if (testFunc(vec[i]))
            numPassed++;
    }
    return numPassed;
}
(d)
```

```
.L4:
mov edi, DWORD PTR [rdx+rbx*4] ; read rbx'th element of vec
; (inlined vector::operator [])

call testFunc(int) ; call test function
mov rdx, QWORD PTR [rbp+0] ; reread vector base pointer
cmp al, 1 ; was the result of test true?
mov rax, QWORD PTR [rbp+8] ; reread the vector end pointer
sbb r12d, -1 ; add 1 if true, 0 if false
inc rbx ; increment loop counter
sub rax, rdx ; subtract end from begin...
sar rax, 2 ; and divide by 4 to get size()
; (inlined vector::size())

cmp rbx, rax ; does loop counter equal size()?
jnb .L4 ; loop if not
(e)
```

```
template<typename T> struct _Vector_impl {
    T * _M_start;
    T * _M_finish;
    T * _M_end_of_storage;
};
(f)
```

can make a big difference. Of course, the more information a compiler has, the longer it could take to run, so there is a balance to be struck here.

Let's take a look at an example (d), counting the number of elements of a vector that pass some test (compiled with GCC, optimization level 3; [https://godbolt.org/z/acm19\\_count1](https://godbolt.org/z/acm19_count1)). If the compiler has no information about testFunc, it will generate an inner loop like (e).

To understand this code, it's useful to know that a std::vector<> contains some pointers: one to the beginning of the data; one to the end of the data; and one to the end of the storage currently allocated (f). The size of the vector is not directly stored, it's implied in the difference between the begin() and end() pointers. Note that the calls to vector<>::size() and vector<>::operator[] have been inlined completely.

In the assembly code (e), ebp points to the vector object, and the begin() and end() pointers are therefore QWORD PTR [rbp+0] and QWORD PTR [rbp+8], respectively.

Another neat trick the compiler has done is to remove any branching: you might reasonably expect if (testFunc(...)) would turn into a comparison and branch. Here the compiler

does the comparison cmp al, 1, which sets the processor carry flag if testFunc() returned false, otherwise it clears it. The sbb r12d, -1 instruction then subtracts -1 with borrow, the subtract equivalent of carrying, which also uses the carry flag. This has the desired side effect: If the carry is clear (testFunc() returned true), it subtracts -1, which is the same as adding 1. If the carry is set, it subtracts -1 + 1, which has no effect on the value. Avoiding branches can be advantageous in some cases if the branch is not easily predictable by the processor.

It may seem surprising that the compiler reloads the begin() and end() pointers each loop iteration, and indeed it rederives size() each time too. However, the compiler is forced to do so: it has no idea what testFunc() does and must assume the worst. That is, it must assume calls to testFunc() may cause the vec to be modified. The const reference here doesn't allow any additional optimizations for a couple of reasons: testFunc() may have a non-const reference to vec (perhaps through a global variable), or testFunc() might cast away const.

If, however, the compiler can see the body of testFunc(), and from this know that it does not in fact modify vec (g), the story is very different

```

.L6:
mov edi, DWORD PTR [rdx] ; read next value
call testFunc(int)      ; call testFunc with it
cmp al, 1               ; check return code
sbb r8d, -1             ; add 1 if true, 0 otherwise
add rdx, 4              ; move to next element
cmp rcx, rdx            ; have we hit the end?
jne .L6                 ; loop if not

```

(g)

```

for (auto val : vec)
{
    if (testFunc(val))
        numPassed++;
}

```

(h)

```

{
    auto __begin = begin(vec);
    auto __end = end(vec);
    for (auto __it = __begin; __it != __end; ++__it)
    {
        if (testFunc(*__it))
            numPassed++;
    }
}

```

(i)

([https://godbolt.org/z/acm19\\_count2](https://godbolt.org/z/acm19_count2)).

In this case the compiler has realized that the `vector`'s `begin()` and `end()` are constant during the operation of the loop. As such it has been able to realize that the call to `size()` is also a constant. Armed with this knowledge, it hoists these constants out of the loop, and then rewrites the index operation (`vec[i]`) to be a pointer walk, starting at `begin()` and walking up one `int` at a time to `end()`. This vastly simplifies the generated assembly.

In this example, I gave the compiler a body to `testFunc()` but marked it as non-inlineable (a GNU extension) to demonstrate this optimization in isolation. In a more realistic codebase, the compiler could inline `testFunc()` if it believed it beneficial.

Another way to enable this optimization without exposing the body of the function to the compiler is to mark it as `[[gnu::pure]]` (another language extension). This promises the compiler that the function is pure—entirely a function of its arguments with no side effects.

Interestingly, using `range-for` in the initial example yields optimal assembly, even without knowing that `testFunc()` does not modify `vec` ([https://godbolt.org/z/acm19\\_count3](https://godbolt.org/z/acm19_count3)). This is because `range-for` is defined as a source code transformation that puts `begin()` and `end()` into local variables as shown in (h) and is interpreted as (i).

All things considered, if you need to

use a “raw” loop, the modern `range-for` style is preferred: it's optimal even if the compiler cannot see the body of called functions, and it is clearer to the reader. Arguably better still is to use the STL's `count_if` function to do all the work for you: the compiler still generates optimal code ([https://godbolt.org/z/acm19\\_count4](https://godbolt.org/z/acm19_count4)).

In the traditional single-translation-unit-at-a-time compilation model, function bodies are often hidden from call sites, as the compiler has seen only their declaration. Link time optimization (LTO, also known as LTCG for link time code generation) can be used to allow the compiler to see across translation unit boundaries. In LTO, individual translation units are compiled to an intermediate form instead of machine code. During the link process—when the entire program (or dynamic linked library) is visible—machine code is generated. The compiler can take advantage of this to inline across translation units, or at least use information about the side effects of called functions to optimize.

Enabling LTO for optimized builds can be a good win in general, as the compiler can see your whole program. I now rely on LTO to let me move more function bodies out of headers to reduce coupling, compile time, and build dependencies for debug builds and tests, while still giving me the performance I need in final builds.

Despite being a relatively established technology (I used LTCG in the

early 2000s on the original Xbox), I have been surprised how few projects use LTO. In part this may be because programmers unintentionally rely on undefined behavior that becomes apparent only when the compiler gets more visibility: I know I have been guilty of this.

## Favorite Optimization Examples

Over the years, I have collected a number of interesting real-world optimizations, both from first-hand experience optimizing my own code and from helping others understand their code on Compiler Explorer. Here are some of my favorite examples of how clever the compiler can be.

**Integer division by a constant.** It may be surprising to learn that—until very recently—about the most expensive thing you could do on a modern CPU is an integer divide. Division is more than 50 times more expensive than addition and more than 10 times more expensive than multiplication. (This was true until Intel's release of the Cannon Lake microarchitecture, where the maximum latency of a 64-bit divide was reduced from 96 cycles to 18.<sup>6</sup> This is only around 20 times slower than an addition, and 5 times more expensive than multiplication.)

Thankfully, compiler authors have some strength reduction tricks up their sleeves when it comes to division by a constant. I am sure we have all realized that division by a power of two can often be replaced by a logical shift right—rest assured the compiler will do this for you. I would advise not writing a `>>` in your code to do division; let the compiler work it out for you. It's clearer, and the compiler also knows how to account properly for signed values: integer division truncates toward zero, and shifting down by itself truncates toward negative infinity.

However, what if you are dividing by a non-power-of-two value, as illustrated in (j)? Are you out of luck? Luckily, the compiler has your back again. The code compiles to (k) ([https://godbolt.org/z/acm19\\_div3](https://godbolt.org/z/acm19_div3)).

Not a divide instruction in sight. Just a shift, and a multiply by a strange large constant: the 32-bit unsigned input value is multiplied by `0xaaaaaab`, and the resulting 64-bit value is shifted down by 33 bits. The compiler has re-

placed division with a cheaper multiplication by the reciprocal, in fixed point. The fixed point in this case is at bit 33, and the constant is one-third expressed in these terms (it's actually 0.3333333337213844). The compiler has an algorithm for determining appropriate fixed points and constants to achieve the division while preserving the same rounding as an actual division operation with the same precision over the range of the inputs. Sometimes this requires a number of extra operations—for example, in dividing by 1022 as shown in (l, [https://godbolt.org/z/acm19\\_div1023](https://godbolt.org/z/acm19_div1023)).

The algorithm is well known and documented extensively in the excellent book, *Hacker's Delight*.<sup>8</sup>

In short, you can rely on the compiler to do a great job of optimizing division by a compile-time-known constant.

You might be thinking: Why is this such an important optimization? How often does one actually perform integer division, anyway? The issue is not so much with division itself as with the related modulus operation, which is often used in hash-map implementations as the operation to bring a hash value into the range of the number of hash buckets.

Knowing what the compiler can do

here can lead to interesting hash-map implementations. One approach is to use a fixed number of buckets to allow the compiler to generate the perfect modulus operation without using the expensive divide instruction.

Most hash maps support rehashing to a different number of buckets. Naively, this would lead to a modulus with a number known only at runtime, forcing the compiler to emit a slow divide instruction. Indeed, this is what the GCC libstdc++ library implementation of `std::unordered_map` does.

Clang's libc++ goes a little further: it checks if the number of buckets is a power of two, and if so skips the divide instruction in favor of a logical AND. Having a power-of-two bucket count is alluring as it makes the modulus operation fast, but in order to avoid excessive collisions it relies on having a good hash function. A prime-number bucket count gives decent collision resistance even for simplistic hash functions.

Some libraries such as `boost::multi_index` go a step further: instead of storing the actual number of buckets, they use a fixed number of prime-sized bucket counts (m).

That way, for all possible hash-table sizes the compiler generates the perfect modulus code, and the only extra

**You can rely on the compiler to do a great job of optimizing division by a compile-time-known constant.**

```
unsigned divideByThree(unsigned x)
{
    return x / 3;
}
(l)
```

```
divideByThree(unsigned int):
mov  eax, edi          ; eax = edi
mov  edi, 2863311531   ; edi = 0xaaaaaaaaab
imul rax, rdi         ; rax = rax * 0xaaaaaaaaab
shr  rax, 33          ; rax >>= 33
ret
(k)
```

```
divideBy1023(unsigned int):
mov  eax, edi
imul rax, rax, 4198405
shr  rax, 32
sub  edi, eax
shr  edi
add  eax, edi
shr  eax, 9
ret
(l)
```

```
size_t reduce(size_t hash, int bucketCountIndex) {
    switch (tableSizeIndex)
    {
        case 0: return hash % 7;
        case 1: return hash % 17;
        case 2: return hash % 37;
        // and so on...
    }
}
(m)
```

```
bool divisibleBy3(unsigned x)
{
    return x % 3 == 0;
}
(n)
```

```
divisibleBy3(unsigned int):
    imul edi, edi, -1431655765    ; edi = edi * 0xa5555555
    cmp edi, 1431655765          ; compare with 0x55555555
    setbe al                     ; return 1 if edi <= 0x55555555
    ret
(o)
```

```
int countSetBits(unsigned a)
{
    int count = 0;
    while (a != 0)
    {
        count++;
        a &= (a - 1); // clears the bottom set bit
    }
    return count;
}
(p)
```

```
countSetBits(unsigned int):
    xor eax, eax                ; count = 0
    test edi, edi               ; is a == 0?
    je .L4                     ; if so, return
.L3:
    inc eax                    ; count ++
    bsr edi, edi               ; a &= (a - 1);
    jne .L3                    ; jump back to L3 if a != 0
    ret
.L4:
    Ret
(q)
```

```
countSetBits(unsigned int):
    popcnt eax, edi            ; count = number of set bits in a
    ret
(r)
```

```
countSetBits(unsigned int):
    xor eax, eax                ; count = 0
    popcnt eax, edi            ; count = number of set bits in a
    ret
(s)
```

```
bool isWhitespace(char c)
{
    return c == ' '
        || c == '\r'
        || c == '\n'
        || c == '\t';
}
(t)
```

```
isWhitespace(char):
    xor eax, eax                ; result = false
    cmp dil, 32                 ; is c > 32
    ja .L4                     ; if so, exit with false
    movabs rax, 4294977024      ; rax = 0x100002600
    shr rax, rax, rdi           ; rax >>= c
    and eax, 1                  ; result = rax & 1
.L4:
    ret
(u)
```

cost is to dispatch to the correct piece of code in the switch statement.

GCC 9 has a neat trick (n) for checking for divisibility by a non-power-of-two ([https://godbolt.org/z/acm19\\_mul-tof3](https://godbolt.org/z/acm19_mul-tof3)) and compiles to (o).

This apparent witchcraft is explained very well by Daniel Lemire in his blog.<sup>2</sup> As an aside, it's possible to do

these kinds of integer division tricks at runtime too. If you need to divide many numbers by the same value, you can use a library such as `libdivide`.<sup>5</sup>

### Counting Set Bits

How often have you wondered, How many set bits are in this integer? Probably not all that often. But it turns

out this simple operation is surprisingly useful in a number of cases. For example, calculating the Hamming distance between two bitsets, dealing with packed representations of sparse matrices, or handling the results of vector operations.

You might write a function to count the bits as shown in (p). Of note is the bit manipulation “trick” `a &= (a - 1);`, which clears the bottom-most set bit. It's a fun one to prove to yourself how it works on paper. Give it a go.

When targeting the Haswell microarchitecture, GCC 8.2 compiles this code to the assembly shown in (q) ([https://godbolt.org/z/acm19\\_bits](https://godbolt.org/z/acm19_bits)). Note how GCC has cleverly found the `BLSR` bit-manipulation instruction to pick off the bottom set bit. Neat, right? But not as clever as Clang 7.0 illustrated in (r).

This operation is common enough that there is an instruction on most CPU architectures to do it in one go: `POPCNT` (population count). Clang is clever enough to take a whole loop in C++ and reduce it to a single instruction. This is a great example of good instruction selection: Clang's code generator recognizes this pattern and is able to choose the perfect instruction.

I was actually being a little unfair here: GCC 9 also implements this (s), and in fact shows a slight difference. At first glance this appears to be suboptimal: Why on earth would you write a zero value, only to overwrite it immediately with the result of the “population count” instruction `popcnt`?

A little research brings up Intel CPU erratum SKL029: “`POPCNT` Instruction May Take Longer to Execute Than Expected”—there is a CPU bug! Although the `popcnt` instruction completely overwrites the output register `eax`, it is incorrectly tagged as depending on the prior value of `eax`. This limits the CPU's ability to schedule the instruction until any prior instructions writing to `eax` have completed—even though they have no impact.

GCC's approach here is to break the dependency on `eax`: the CPU recognizes `xor eax, eax` as a dependency-breaking idiom. No prior instruction can influence `eax` after `xor eax, eax`, and the `popcnt` can run as soon as its input operand `edi` is available.

This affects only Intel CPUs and seems to be fixed in the Cannon Lake



microarchitecture, although GCC still emits XOR when targeting it.

### Chained Conditionals

Maybe you have never needed to count the number of set bits in an integer, but you have probably written code like the example in (t). Instinctively, I thought the code generation would be full of compares and branches, but both Clang and GCC use a clever trick to make this code pretty efficient. GCC 9.1's output is shown in (u; [https://godbolt.org/z/acm19\\_conds](https://godbolt.org/z/acm19_conds))

The compilers turn this sequence of comparisons into a lookup table. The magic value loaded into rax is a 33-bit lookup table, with a one-bit in the locations where you would return true (indices 32, 13, 10, and 9 for ' ', \r, \n, and \t, respectively). The shift and & then pick out the cth bit and return it. Clang generates slightly different but broadly equivalent code. This is another example of strength reduction.

I was pleasantly surprised to see this kind of optimization. This is definitely the kind of thing that—prior to investigating in Compiler Explorer—I would have written manually assuming I knew better than the compiler.

One unfortunate thing I did notice while experimenting is—for GCC, at least—the order of the comparisons can affect the compiler's ability to make this optimization. If you switch the order of the comparison of the \r and \n, GCC generates the code in (v).

There's a pretty neat trick with the and to combine the comparison of \r and \t, but this seems worse than the code generated before. That said, a simplistic benchmark on Quick Bench suggests the compare-based version might be a tiny bit faster in a predictable tight loop.<sup>a</sup> Who ever said this was simple, eh?

### Summation

Sometimes you need to add a bunch of things up. Compilers are extremely good at taking advantage of the vectorized instructions available in most CPUs these days, so even a pretty straightforward piece of code such as (w) gets turned into code whose core loop looks like (x) ([https://godbolt.org/z/acm19\\_sum](https://godbolt.org/z/acm19_sum)).

<sup>a</sup> <http://quick-bench.com/0TbNkJr6KkEXyy6ixHn3ObBEi4w>

The compiler has been able to process eight values per instruction, by separating the total into eight separate subtotals for each one. At the end it sums across those subtotals to make the final total. It's as if the code was rewritten for you to look more like the (y) example.

Simply place the compiler's optimization level at a high enough setting and pick an appropriate CPU architecture to target, and vectorization kicks in. Fantastic!

This does rely on the fact that separating the totals into individual subtotals and then summing at the end is equivalent to adding them in the order the program specified. For integers, this is trivially true; but for floating-point data types this is not the case. Floating point operations are not associative:  $(a+b)+c$  is not the same as  $a+(b+c)$ , as—among other things—the precision of

the result of an addition depends on the relative magnitude of the two inputs.

This means, unfortunately, that changing the `vector<int>` to be a `vector<float>` does not result in the code you would ideally want. The compiler could use some vector operations (it can square eight values at once), but it is forced to sum across those values serially as shown in (z; [https://godbolt.org/z/acm19\\_sumf](https://godbolt.org/z/acm19_sumf)).

This is unfortunate, and there is not an easy way around it. If you are absolutely sure the order of addition is not important in your case, you can give GCC the dangerous (but amusingly named) `-funsafe-math-optimizations` flag. This lets it generate this beautiful inner loop illustrated in (a'; [https://godbolt.org/z/acm19\\_sumf\\_unsafe](https://godbolt.org/z/acm19_sumf_unsafe)).

Amazing stuff: processing eight floats at a time, using a single instruction to ac-

```
isWhitespace(char):
  cmp dil, 32          ; is c == 32?
  sete al             ; al = 1 if so, else 0
  cmp dil, 10         ; is c == 10?
  sete dl             ; dl = 1 if so, else 0
  or al, dl           ; al |= dl
  jne .L3             ; if al is non-zero return it (c was ' ' or '\n')
  and edi, -5         ; clear bit 2 (the only bit that differs between
                      ; '\r' and '\t')
  cmp dil, 9          ; compare with '\t'
  sete al             ; dl = 1 if so, else 0
.L3:
  ret
(v)
```

```
int sumSquared(const vector<int> &v)
{
  int res = 0;
  for (auto i : v)
  {
    res += i * i;
  }
  return res;
}
(w)
```

```
.loop:
  vmovdqu ymm2, YMMWORD PTR [rax] ; read 32 bytes into ymm2
  add rax, 32                       ; advance to the next element
  vpmuldq ymm0, ymm2, ymm2         ; square ymm2, treating as
  ; 8 32-bit values
  vpaddq ymm1, ymm1, ymm0         ; add to sub-totals
  cmp rax, rdx                     ; have we reached the end?
  jne .loop                       ; if not, keep looping
(x)
```

```
int res_[] = {0,0,0,0,0,0,0,0};
for (; index < v.size(); index += 8)
{
  // This can be performed by parallel instructions without
  // an actual loop. The following boils down to a couple
  // of vector instructions:
  for (size_t j = 0; j < 8; ++j)
  {
    auto val = v[index + j];
    res_[j] += val * val;
  }
}
res = res_[0] + res_[1]
+ res_[2] + res_[3]
+ res_[4] + res_[5]
+ res_[6] + res_[7];
(y)
```

```

.loop:
  vmovups ymm4, YMMWORD PTR [rax] ; read 32 bytes into ymm4
  add rax, 32 ; advance
  vmulps ymm1, ymm4, ymm4 ; square 8 floats
                               ; (the one parallel operation)
  vaddss xmm0, xmm0, xmm1 ; accumulate the first value
  vshufps xmm3, xmm1, xmm1, 85 ; shuffle things around
                               ; (permutes the 8 floats
                               ; within the register)
  ; ...
  vshufps xmm2, xmm1, xmm1, 255 ; ...
  vaddss xmm0, xmm0, xmm3 ; accumulate the second value
  vunpckhps xmm3, xmm1, xmm1 ; more shuffling
  vextractf128 xmm1, ymm1, 0x1 ; ...
  vaddss xmm0, xmm0, xmm3 ; accumulate third...
  vaddss xmm0, xmm0, xmm2 ; and fourth value
  vshufps xmm2, xmm1, xmm1, 85 ; shuffling
  vaddss xmm0, xmm0, xmm1 ; accumulate fifth
  vaddss xmm0, xmm0, xmm2 ; and sixth
  vunpckhps xmm2, xmm1, xmm1 ; shuffle some more...
  vshufps xmm1, xmm1, xmm1, 255 ; ...
  vaddss xmm0, xmm0, xmm0, xmm2 ; accumulate the seventh
  vaddss xmm0, xmm0, xmm1 ; and final value
  cmp rax, rcx ; are we done?
  jne .loop ; if not, keep going

```

(z)

```

.loop:
  vmovups ymm2, YMMWORD PTR [rax] ; read 8 floats
  add rax, 32 ; advance
  vfmadd23ps ymm0, ymm2, ymm2 ; for the 8 floats:
                               ; ymm0 += ymm2 * ymm2
  cmp rax, rcx ; are we done?
  jne .loop ; if not, keep going

```

(a')

```

int sumToX(int x)
{
  int result = 0;
  for (int i = 0; i < x; ++i)
  {
    result += i;
  }
  return result;
}

```

(b')

```

sumToX(int): # @sumToX(int)
  test edi, edi ; test x
  jle .zeroOrBelow ; skip if x <= 0
  lea eax, [rdi - 1] ; eax = x - 1
  lea ecx, [rdi - 2] ; ecx = x - 2
  imul rcx, rax ; rcx = ecx * eax
  shr rcx ; rcx >>= 1
  lea eax, [rcx + rdi] ; eax = rcx + x
  add eax, -1 ; return eax - 1
  ret
.zeroOrBelow:
  xor eax, eax ; answer is zero
  ret

```

(c')

cumulate and square. The drawback is potentially unbounded precision loss. Additionally, GCC does not allow you to turn this feature on for just the functions you need it for—it's a per-compilation unit flag. Clang at least lets you control it in the source code with `#pragma Clang fp contract`.

While playing around with these kinds of optimizations, I discovered that compilers have even more tricks up their sleeves, check out (b'). GCC generates fairly straightforward code for this, and with appropriate compiler settings will use vector operations

as noted earlier. Clang, however, generates the code in (c'); ([https://godbolt.org/z/acm19\\_sum\\_up](https://godbolt.org/z/acm19_sum_up)).

First, note there is no loop at all. Working through the generated code, you see that Clang returns:

$$\frac{(x-1)(x-2)}{2} + (x+1)$$

It has replaced the iteration of a loop with a closed-form general solution of the sum. The solution differs from what I would naively write myself:

$$\frac{x(x-1)}{2}$$

This is presumably a result of the general algorithm Clang uses.

Further experimentation shows that Clang is clever enough to optimize many of these types of loops. Both Clang and GCC track loop variables in a way that allows this kind of optimization, but only Clang chooses to generate the closed-form version. It's not always less work: for small values of  $x$  the overhead of the closed-form solution might be more than just looping. Krister Walfridsson goes into great detail about how this is achieved in a blog post.<sup>7</sup>

It is also worth noting that in order to do this optimization, the compiler may rely on signed integer overflow being undefined behavior. As such, it can assume your code cannot pass a value of  $x$  that would overflow the result (65536, in this case). If Clang cannot make that assumption, it is sometimes unable to find a closed-form solution ([https://godbolt.org/z/acm19\\_sum\\_fail](https://godbolt.org/z/acm19_sum_fail)).

## Devirtualization

Although it seems to have fallen out of favor a little, traditional virtual-function-based polymorphism has its place. Whether it's to allow for genuine polymorphic behavior, or add a "seam" for testability, or allow for future extensibility, polymorphism through virtual functions can be a convenient choice.

As we know, though, virtual functions are slow. Or are they? Let's see how they affect the sum-of-squares example from earlier—something like (d').

Of course, this is not polymorphic yet. A quick run through the compiler shows the same highly vectorized assembly ([https://godbolt.org/z/acm19\\_poly1](https://godbolt.org/z/acm19_poly1)).

Now adding the virtual keyword in front of the `int` operator() should result in a much slower implementation, filled with indirect calls, right? Well, sort of ([https://godbolt.org/z/acm19\\_poly2](https://godbolt.org/z/acm19_poly2)). There is a lot more going on than before, but at the core of the loop is something perhaps surprising (e').

What is happened here is GCC has made a bet. Given that it has seen only one implementation of the `Transform` class, it is likely going to be that one implementation that is used. Instead of blindly indirecting through the virtual function pointer, it has taken the slight hit of comparing the pointer

```

struct Transform
{
    int operator()(int x) const { return x * x; }
};

int sumTransformed(const vector<int> &v,
                  const Transform &transform)
{
    int res = 0;
    for (auto i : v)
    {
        res += transform(i);
    }
    return res;
}
(d')

```

```

.L8:                                ; rdx points to the vtable
mov rax, QWORD PTR [rdx]           ; read the virtual function pointer
mov esi, DWORD PTR [rbx]           ; read the next int element
                                   ; compare the function pointer with the address of the only
                                   ; known implementation...
cmp rax, Transform::operator() (int) const
jne .L5                             ; if it's not the only known impl,
                                   ; then jump off to a more complex case
imul esi, esi                       ; square the number
add rbx, 4                          ; move to next
add r12d, esi                       ; accumulate the square
cmp rbp, rbx                        ; finished?
jne .L8                             ; if not, loop
(e')

```

against the only known implementation. If it matches, then the compiler knows what to do: it inlines the body of the `Transform::operator()` and squares it in place.

That's right: the compiler has inlined a virtual call. This is amazing, and was a huge surprise when I first discovered this. This optimization is called *speculative devirtualization* and is the source of continued research and improvement by compiler writers. Compilers are capable of devirtualizing at LTO time too, allowing for whole-program determination of possible function implementations.

The compiler has missed a trick, however. Note that at the top of the loop it reloads the virtual function pointer from the vtable every time. If the compiler were able to notice that this value remains constant if the called function does not change the dynamic type of `Transform`, this check could be hoisted out of the loop, and then there would be no dynamic checks in the loop at all. The compiler could use loop-invariant code motion to hoist the vtable check out of the loop. At this point the other optimizations could kick in, and the whole code could be replaced with the vectorized loop from earlier in the case of the vtable check passing.

You would be forgiven for thinking that the dynamic type of the object could not possibly change, but it's actu-

ally allowed by the standard: an object can placement new over itself so long as it returns to its original type by the time it's destructed. I recommend that you never do this, though. Clang has an option to promise you never do such horrible things in your code: `-fstrict-vtable-pointers`.

Of the compilers I use, GCC is the only one that does this as a matter of course, but Clang is overhauling its type system to leverage this kind of optimization more.<sup>4</sup>


C++11 added the final specifier to allow classes and virtual methods to be marked as not being further overridden. This gives the compiler more information about which methods may profit from such optimizations, and in some cases may even allow the compiler to avoid a virtual call completely ([https://godbolt.org/z/acm19\\_poly3](https://godbolt.org/z/acm19_poly3)). Even without the final keyword, sometimes the analysis phase is able to prove that a particular concrete class is being used ([https://godbolt.org/z/acm19\\_poly4](https://godbolt.org/z/acm19_poly4)). Such static devirtualization can yield significant performance improvements.

## Conclusion

Hopefully, after reading this article, you will appreciate the lengths to which the compiler goes to ensure efficient code generation. I hope that some of these optimizations are a pleasant surprise and will factor in your decisions

to write clear, intention-revealing code and leave it to the compiler to do the right thing. I have reinforced the idea that the more information the compiler has, the better job it can do. This includes allowing the compiler to see more of your code at once, as well as giving the compiler the right information about the CPU architecture you are targeting. There is a trade-off to be made in giving the compiler more information: it can make compilation slower. Technologies such as link time optimization can give you the best of both worlds.

Optimizations in compilers continue to improve, and upcoming improvements in indirect calls and virtual function dispatch might soon lead to even faster polymorphism. I am excited about the future of compiler optimizations. Go take a look at your compiler's output.

**Acknowledgments.** The author would like to extend his thanks to Matt Hellige, Robert Douglas, and Samy Al Bahra, who gave feedback on drafts of this article. 

## Related articles on [queue.acm.org](https://queue.acm.org)

### C Is Not a Low-level Language

David Chisnall

<https://queue.acm.org/detail.cfm?id=3212479>

### Uninitialized Reads

Robert C. Seacord

<https://queue.acm.org/detail.cfm?id=3041020>

### You Don't Know Jack about Shared Variables or Memory Models

Hans-J. Boehm, Sarita V. Adve

<https://queue.acm.org/detail.cfm?id=2088916>

## References

1. Godbolt, M. Compiler explorer, 2012; <https://godbolt.org/>.
2. Lemire, D. Faster remainders when the divisor is a constant: beating compilers and libdivide, 2019; <http://bit.ly/33nzs4/>.
3. LLVM. The LLVM compiler infrastructure, 2003; <https://llvm.org>.
4. Padlewski, P. RFC: Devirtualization v2. LLVM; <http://lists.llvm.org/pipermail/llvm-dev/2018-March/121931.html>.
5. ridiculous\_fish. Libdivide, 2010; <https://libdivide.com/>.
6. Uops. Uops.info Instruction Latency Tables; <https://uops.info/table.html>.
7. Walfridsson, K. How LLVM optimizes power sums, 2019; <https://kristerw.blogspot.com/2019/04/how-llvm-optimizes-geometric-sums.html>.
8. Warren, H.S. *Hacker's Delight*. 2nd edition. Addison-Wesley Professional, 2012.

**Matt Godbolt** is the creator of the Compiler Explorer website. He is passionate about writing efficient code. He currently works at Aquatic Capital, and has worked on low-latency trading systems, worked on mobile apps at Google, run his own C++ tools company, and spent more than a decade making console games.

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

DOI:10.1145/3364684

## Exploring the opportunities to use ML, the possible designs, and our experience with Microsoft Azure.

BY RICARDO BIANCHINI, MARCUS FONTOURA, ELI CORTEZ, ANAND BONDE, ALEXANDRE MUZIO, ANA-MARIA CONSTANTIN, THOMAS MOSCIBRODA, GABRIEL MAGALHAES, GIRISH BABLANI, AND MARK RUSSINOVICH

# Toward ML-Centric Cloud Platforms

CLOUD PLATFORMS, SUCH AS Microsoft Azure, Amazon Web Services (AWS), and Google Cloud Platform, are tremendously complex. For example, the Azure Compute fabric governs all the physical and virtualized resources running in Microsoft's datacenters. Its main resource management systems include virtual machine (VM) and container (hereafter we refer to VMs and containers simply as "containers") scheduling, server and container health monitoring and repairs, power and energy management, and other management functions.

Cloud platforms are also extremely expensive to build and operate, so providers have a strong incentive to optimize their use. A nascent approach is to leverage machine learning (ML) in the platforms'

resource management using supervised learning techniques, such as gradient-boosted trees and neural networks, or reinforcement learning. We also discuss why ML is often preferable to traditional non-ML techniques.

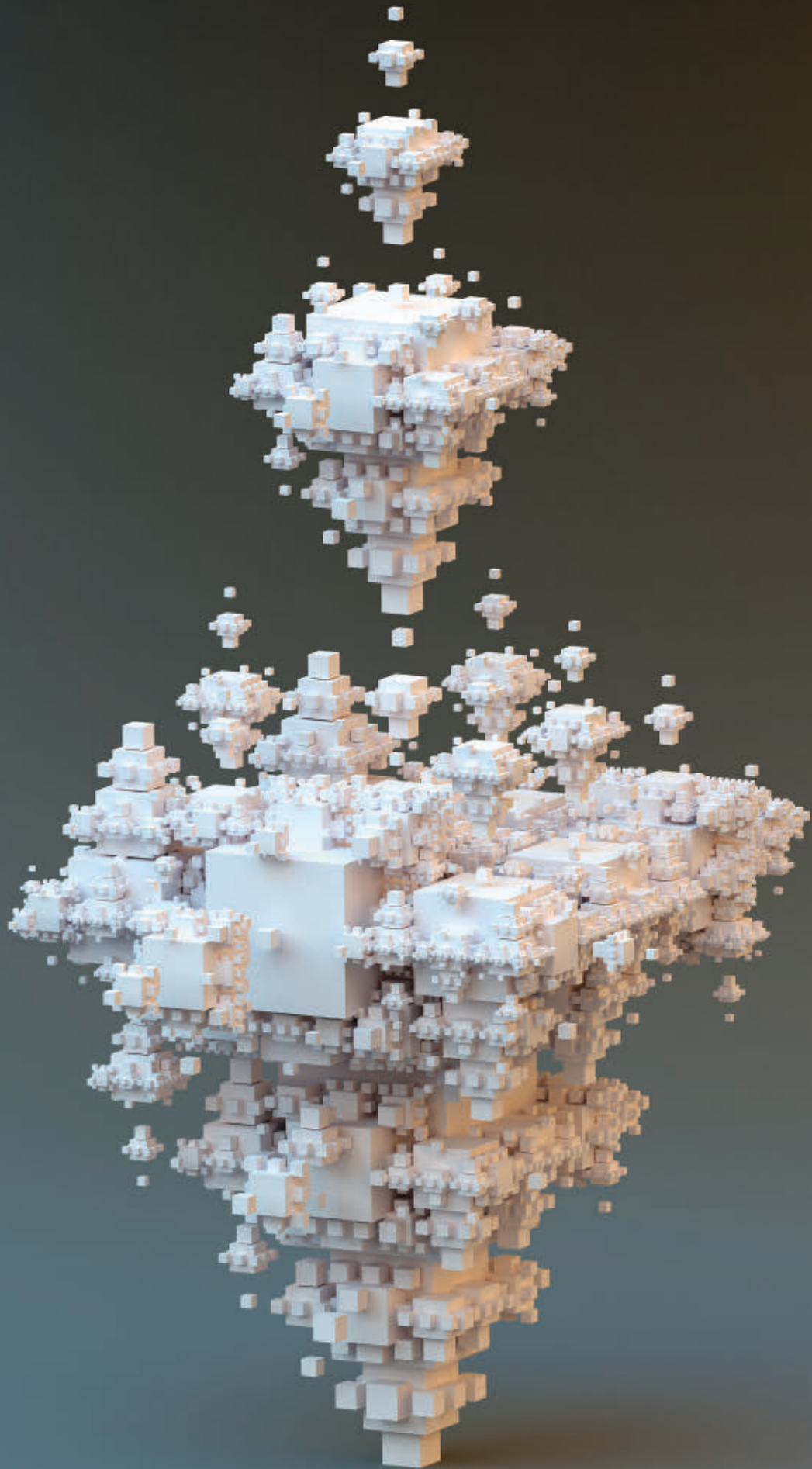
Public cloud providers are starting to explore ML-based resource management in production.<sup>9,14</sup> For example, Google uses neural networks to optimize fan speeds and other energy knobs.<sup>14</sup> In academia, researchers have proposed using collaborative filtering—a common technique in recommender systems—in scheduling containers for reduced with in-server performance interference.<sup>12</sup> Others proposed using reinforcement learning to adjust the resources allocated to co-located VMs.<sup>24</sup> Later, we discuss other opportunities for ML-based management.

Despite these prior efforts and opportunities, it is currently unclear how best to integrate ML into cloud resource management. In fact, prior approaches differ in multiple dimensions. For example, in some cases, the ML technique produces insights/predictions about the workload or infrastructure; in others, it produces actual resource management actions. In some cases, the ML is deeply integrated with the resource manager; in others, it is completely separate. In all cases, the ML addresses a single management problem; a different problem requires

### >> key insights

- **There are many potential uses of ML in cloud computing platforms. The challenge is in defining exactly how and where ML should be infused in these platforms.**
- **Leveraging ML-derived predictions has shown promise for many resource managers in Azure Compute. Having a general and independent ML framework/system has been key to increasing adoption quickly.**
- **Many research challenges remain open, including how to make action-prescribing ML general enough for wide applicability in cloud platforms, how to manage (potentially partial) feedback at scale, and how to debug misbehaviors (especially when the ML is tightly integrated with resource managers).**

IMAGE BY MARCEL CLEMENS



another approach. We discuss these dimensions, the possible integration designs, and their architectural, functional, and API implications.

As one point in this multi-dimensional space, we built Resource Central (RC)<sup>9</sup>—a general ML and prediction-serving system for providing workload and infrastructure insights to resource managers in the Azure Compute fabric. RC collects telemetry from containers and servers, learns from their prior behaviors and, when requested, produces predictions of their future behaviors. We are currently using RC to accurately predict many characteristics of the Azure Compute workload. We present an overview RC, its initial uses and results, and describe the lessons from building it.

Though RC has been successful so far, it has limitations. For example, it does not implement certain forms of interaction with resource managers. More broadly, the integration of ML into real cloud platforms in a general, maintainable, and at-scale manner is still in its infancy. We close the article with some open questions and challenges.

### ML vs. Traditional Techniques

Resource management in cloud platforms is often implemented by static policies that have two shortcomings. First, they are tuned offline based on relatively few benchmark workloads. For example, threshold-based policies typically involve hand-tuned thresholds that must be used for widely different workloads. In contrast, ML-informed dynamic policies can naturally adapt to actual production workloads.<sup>20,26</sup> For the same example, each server can learn different thresholds for its own resource management.

Second, the static policies tend to require reactive actions, and may incur unnecessary overheads and customer impact. As an example, consider a common policy for scheduling containers onto servers, such as best fit. It may cause some co-located containers to interfere in their use of resources (for example, shared cache space) and require (reactive) live migrations.<sup>23</sup> Live migration is expensive and may cause a period of unavailability (aka “black-out” time). In contrast, ML techniques enable predictive management: having accurate predictions of container inter-



**Cloud platforms are extremely expensive to build and operate, so providers have a strong incentive to optimize their use.**

ference enables the scheduler to select placements that do not require migrations.<sup>12</sup> In the Resource Central section, we mention our earlier results on the benefit of predictive container scheduling. In fact, non-predictive policies (for example, based on feedback control) are not even acceptable in some cases. For instance, blindly live migrating containers that will incur long blackout times is certain to annoy customers.

ML has been shown to produce more accurate predictions for cloud resource management than more traditional methods, such as regressions or time-series analysis. For example, Cao<sup>6</sup> and Chen<sup>8</sup> demonstrate that ML techniques produce more accurate resource utilization predictions than time-series models. Our results quantitatively compare some ML and non-ML methods.

### Opportunities for ML in Cloud Platforms

Cloud platforms involve a variety of resource managers, such as the container scheduler and the server health management system. Here, we discuss some of the ways in which managers can benefit from ML.

*Container scheduler.* The scheduler selects the server on which a container will run. It can use ML to identify (and avoid) container placements that would lead to performance interference, or to adjust its configuration parameters (for example, how tightly to pack containers on each server). It can also use ML-derived predictions of the containers’ resource utilizations to balance the disk access load, or to reduce the likelihood of physical resource exhaustion in oversubscribed servers. Predictions of server health are also useful for it to stop assigning containers to servers that are likely to fail soon. Finally, it can use predictions of container lifetime when considering servers that will undergo planned maintenance or software updates. We have used lifetime predictions to match batch workloads to latency-sensitive services with enough idle capacity for the container.<sup>27</sup>

*Server defragmenter/migration manager.* As containers arrive/complete, each server may be left with available resources that are insufficient for large containers. As a result, the server defragmentation system may decide to

live-migrate active containers onto a subset of the servers. A migration manager can also live-migrate (for example, low-priority) containers to alleviate any unexpected server resource contention or interference. It can use application-level performance information (when it is available) or ML techniques on lower-level performance counters to identify these behaviors. The manager can use predictions of the containers' expected lifetimes and blackout times to live-migrate only those that will likely remain active for a substantial amount of time and not incur a noticeable blackout time if migrated.

**Power capping manager.** This manager ensures the capacity of the (over-subscribed) power delivery system is not exceeded, using CPU speed scaling. To tackle a power emergency (the power draw is about to exceed a circuit breaker limit), this manager can use predictions of the performance impact of speed scaling on different workloads to guide its apportioning of the available power budget. Similarly, it can use predictions of workload interactivity as a guide. Ideally, containers executing interactive or highly sensitive workloads should receive all the power they want, to the detriment of containers running batch and background tasks. In this context, the container scheduler can use predictions of interactivity to smartly schedule interactive and delay-insensitive workloads across servers.

**Server health manager.** This manager monitors hardware health and takes faulty servers out of rotation for maintenance. When a server starts to misbehave, this manager can use predictions of the lifetime of the containers running on the server. Using these predictions, it can determine when maintenance can be scheduled, and whether containers need to be live-migrated to prevent unavailability.

This is only a partial list of opportunities for ML-based resource management. The challenge is determining the best system designs for exploiting these opportunities.

**Potential Designs for ML-Centric Clouds**

When deciding how to exploit ML in cloud resource management, we must consider: The ML techniques and their inputs and outputs, and

the managers and their mechanisms (management actions) and policies. We must also consider many questions: Can we use application-level performance data for learning? How should the ML and the managers interact? Should the ML produce behavioral insights/predictions or actual management actions? How tightly integrated with the managers should the ML be? How quickly does the ML need to observe the effect of the management actions? Is it possible to create general frameworks/APIs that can apply to many types of resource management? Next, we discuss our thoughts along these dimensions.

**Application performance vs. counters.** Managers must optimize resource usage without noticeably hurting end-to-end application performance. Thus, having direct data on application performance enables precise management with or without ML. Some application metrics are easier to obtain than others. For example, VM lifetimes are "visible" to the platform, whereas request latencies within VMs implementing a service often are not. When containers are opaque to the platform, the way to obtain application performance data is for developers to instrument

their codes with monitoring calls into the platform (for example, using AWS's CloudWatch<sup>3</sup> or Azure's Monitor<sup>21</sup>). When they do not, lower-level counters (for example, resource utilization, CPU performance counters) must be used as an imperfect proxy for application performance. Given that most workloads are not instrumented, we expect that ML techniques will most often use counters. Nevertheless, providers also run first-party workloads, which can potentially be instrumented.

**Predictions vs. actions.** Another dimension concerns the role of the ML techniques. One approach is for them to produce insights (for example, performance, load, container lifetime predictions) that managers can leverage to improve decisions as in Figure 1 (top). This approach gives managers sole control and understanding of the management policies. Another approach is for the ML to produce actual management actions (for example, migrate this container, change this resource allocation) to be taken by managers. In this case, the ML embodies a deeper understanding of the policies (or may itself define the policies). Targeting the ML at producing actions may lead to policies that more easily adapt to the actual

Figure 1. Two designs.

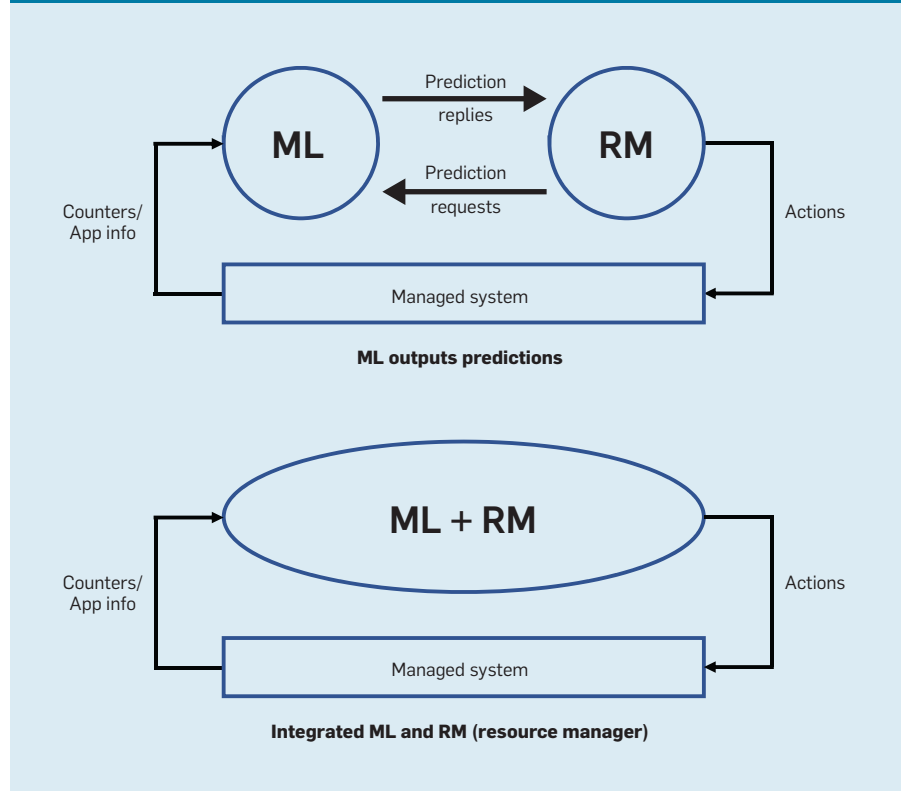
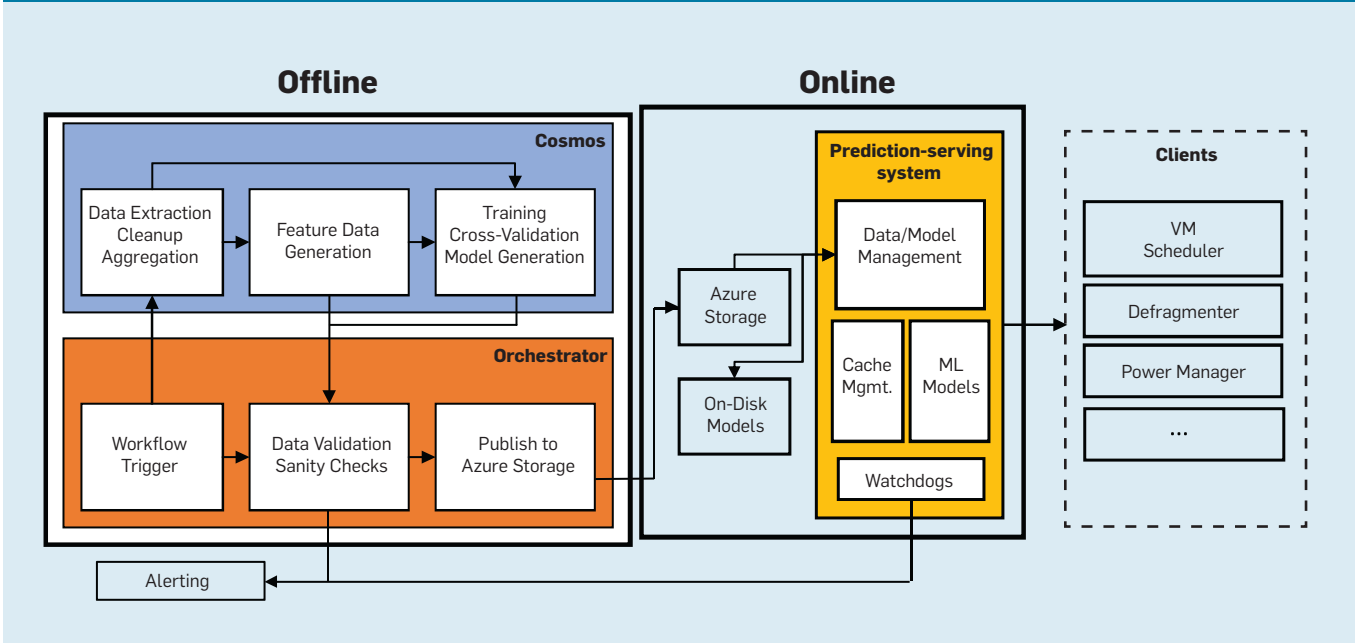


Figure 2. RC architecture comprising offline and online components.



workload and infrastructure behaviors, whereas leaving this responsibility to managers may produce policies that are unnecessarily general. Producing actions may also be the only alternative when it is impractical to collect labeled training data (for example, when fast management decisions must be made at the servers themselves, based on fine-grained performance data). On the other hand, leveraging ML for insights simplifies the managers, making them easier to understand and debug. In fact, relying on insights is less likely to cause negative feedback loops that could potentially degrade customer experience. Insights may also inform multiple managers (for example, container scheduler and power manager),

whereas actions are manager-specific.

*Integration vs. separation.* Related to the dimension here is the question of whether the ML should be fully integrated or completely separate from managers. When the ML outputs actions, the fully integrated design is a natural one, as in Figure 1 (bottom). For insights, both integration and separation are viable options. However, for generality and maintainability, cleanly separating the ML and the managers via well-defined APIs is beneficial: multiple managers can use the same ML implementation, which the platform can maintain independently of the managers.

*Immediate vs. delayed feedback.* A final dimension is whether the ML is

able to observe the result of its previous outputs or the manager actions within a short time. Designs that produce ML models offline will likely observe these effects only at a coarse time granularity (for example, daily). Such granularity is a good match when the input feature characteristics also change slowly. However, techniques such as reinforcement learning and bandit learning often benefit from actions being observable much sooner. For such techniques, offline model-learning may not be ideal.

**Resource Central**

RC is one point in this multidimensional space. We built it as a general ML and prediction-serving system into the Azure Compute fabric. RC<sup>9</sup> learns from low-level counters from all containers and servers, produces various behavioral models offline, and provides predictions online to multiple managers via a simple REST API.

RC leverages a wealth of historical data to produce accurate predictions. For example, from the perspective of each Azure subscription, many containers exhibit peak CPU utilizations in consistent ranges across executions; containers that execute user-facing workloads consistently do so across executions; tenant deployment sizes are unlikely to vary widely across executions, and so on.<sup>9</sup> In all these cases, prior behavior is a good predictor for future behavior.

Table 1. Behavior, ML modeling approaches, model and full feature dataset sizes.

Behavior	Approach	#features	Model size	Feature data size
Avg CPU utilization	Gradient Boosting Tree	247	414KB	416MB
Deployment size	Gradient Boosting Tree	41	351KB	296MB
Lifetime	Gradient Boosting Tree	247	438KB	416MB
Blackout time	Gradient Boosting Tree	998	290KB	4.5MB

Table 2. Behaviors and their buckets.

Behavior	Bucket 1	Bucket 2	Bucket 3	Bucket 4
Avg CPU utilization	0–25%	25%–50%	50%–75%	75%–100%
Deployment size	1	>1 and ≤10	>10 and ≤100	>100
Lifetime	≤15 mins	>15 and ≤60 mins	>1 and ≤24 hs	>24 hs
Blackout time	≤0.1 s	>0.1 and ≤1 s	>1 and ≤3 s	>3 s



RC uses customer, container, and/or server features to identify correlations that managers can leverage in their decision-making. The managers query RC with a subset of the features, expecting to receive predictions for the others. For example, the scheduler may query RC while providing the customer name and type, deployment type and time, and container role name. RC will then predict how large the deployment by this customer may become and how high these containers' resource utilization may get over time.

**Architecture.** *Design rationale.* Our design for RC follows several basic principles related to the dimensions we discussed previously and our ability to operate, maintain, and extend it at scale:

1. Since application-level performance data is rarely available, RC should learn from low-level counters.

2. For generality, modularity, and debuggability, RC should be oblivious to the management policies and, instead, provide workload and infrastructure behavior predictions. It should also provide an API that is general enough for many managers to use.

3. For performance and availability, RC should be an independent system that is off the critical performance and availability paths of the managers that use it whenever possible.

4. Since workload characteristics and server behaviors change slowly, RC can learn offline and serve predictions online. For availability, these two components should be able to operate independently of each other.

5. For maintainability, it should be simple and rely on any existing well-supported infrastructures.

6. For usability, it should require minimal modifications to the resource managers.

*Design.* Figure 2 illustrates how we designed RC based on these principles. The offline workflow consists of data extraction, cleanup, aggregation, feature data generation, training, validation, and ML model generation. RC does these tasks on Cosmos,<sup>7</sup> a massive data processing system that collects all the container and server telemetry from the fabric. RC orchestrates these phases, sanity-checks the models and feature data, and publishes them to Azure storage, a highly available store. RC cur-



## There are many potential uses and designs for ML in cloud platforms.



rently retrains models once a day.

The online part of RC is a REST (Representational State Transfer) service within which the models execute to produce predictions. RC's clients (for example, the container scheduler) call the service passing as input the model name and information about the container(s) for which they want predictions, for example, the subscription identification. The model may require historical feature data as additional inputs, which RC fetches from Azure Storage. As an example of feature data, the lifetime model requires information on historical lifetimes (for example, percentage of short-lived and long-lived containers to date) for the same subscription from the store. Each prediction result is a predicted value and a score. The score reflects the model's confidence in the predicted value. The client may choose to ignore a prediction when the score is too low. It may also ignore (or not wait for) a prediction if it thinks that RC is misbehaving (or unavailable).

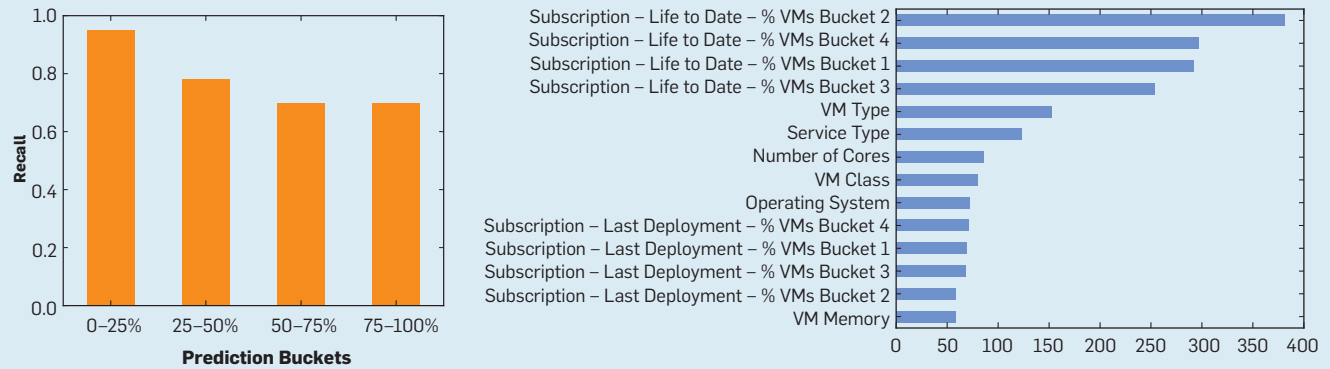
RC relies heavily on caching, as clients may have stringent performance requirements. It caches the prediction results, model, and feature data from the store in memory.

*Current ML models.* RC acts as a framework for offline training of ML models and serving predictions from them online; RC is agnostic to the specific modeling approach data analysts select. In our current implementation, analysts can select models from a large repository that runs on Cosmos. The three leftmost columns of Table 1 list some of the container behaviors we predict and the modeling approach we currently use: Gradient Boosting Trees (GBTs).<sup>18</sup> We are also experimenting with deep neural networks and plan to start using them in the next version of RC.

For classifying numeric behaviors, we divide the space of possible values into buckets (for example, 0%–24%, 25%–49%, and so on) and then predict a bucket. (As we will discuss, this approach has been more accurate for our datasets than using regression and then bucketizing the result.) When the prediction must be converted to a number, the client can assume the highest, middle, or lowest value for the predicted bucket.

*Feature engineering.* Each model takes many features as input, which we

Figure 3. Average CPU utilization recall per bucket (left) and attribute importance (right).



extract from the attributes available in our dataset. We split the attributes into three groups: categorical, boolean, and numerical. We model categorical attributes (for example, container type, guest operating system) as categorical features. We represent the features in a vector of pre-defined length. We concatenate the boolean attribute (for example, first deployment, production workload) values to the input feature vector. Similarly, we normalize and concatenate the numerical attribute (for example, number of cores, container memory size) values to the vector. Finally, we place the attributes that describe observed container/subscription behavior (for example, last observed container lifetime) into the buckets of Table 2 and use them as numerical features. We concatenate these features to the vector as well.

*Comparison to other systems and techniques.* As it focuses on producing predictions, RC fundamentally differs from action-prescribing systems, for example, Agarwal et al.,<sup>2</sup> and Moritz et al.<sup>22</sup> RC currently produces its predictions using TLC, a Microsoft-internal state-of-the-art framework that implements many learning algorithms. However, RC can also leverage recently proposed frameworks, such as TensorFlow,<sup>1</sup> for producing its ML models. RC’s online component is comparable to recent prediction-serving systems,<sup>10,11,16</sup> though with a different architecture and geared toward cloud resource management. We are not aware of any ML and prediction-serving frameworks/systems like RC in other real cloud platforms.

The literature on predicting workload behaviors is extensive. These

works predict resource demand, resource utilization, or job/task length for provisioning or scheduling purposes.<sup>5,6,8,15,17,19,25</sup> For example, Cao<sup>6</sup> recently explored Random Forests to predict CPU, memory, and disk utilizations, whereas Chen<sup>8</sup> used Residual Neural Networks for predicting VM CPU utilization. We predict a broader set of behaviors (including container lifetimes, maximum deployment sizes, and blackout times) for a broader set of purposes (including health management and power capping). Still, we do not argue that the models we use are necessarily the best. Instead, we show them simply as examples of ML models that we have integrated into the RC framework and work well in practice.

**Prediction accuracy.** A key requirement for RC is the ability to predict behaviors accurately. Obviously, this accuracy depends on the behavior one is trying to predict and on the modeling approach they use. As such, the best we can do is provide evidence from our experience with RC that many behaviors can be predicted accurately.

For our analysis, we use one month of data about all VMs in Azure. In this dataset, less than 1% of the VMs are from “new” subscriptions, that is, subscriptions that appear for the first time in the set. We trained RC’s models with the first three weeks and tested them on the fourth. We provide a similar dataset at <https://github.com/Azure/AzurePublicDataset>.

We divide the space of predictions for each behavior into the buckets listed in Table 2. Given these buckets, Figure 3 summarizes the RC prediction results for each VM-utilization bucket (left)

and the most important predictive attributes (right). Figure 4 shows the overall accuracy, prediction, and recall results (three rightmost bars in each group, respectively) for the VM behaviors in the tables. We measure accuracy as the percentage of predictions that were correct, assuming the predicted bucket is that with the highest confidence score; precision for a bucket as the percentage of true positives in the set of predictions that named the bucket; and recall for a bucket as the percentage of true positives in the set of predictions that should have named the bucket.

Figure 3 (left) shows recall between 70% and 95% across VM-utilization buckets. When we average bucket frequencies and recalls together, we find that overall VM-utilization recall is 89%. Figure 3 (right) shows that the most important attributes in terms of F1-score are the percentage of VMs of the same subscription that fell in each bucket to date. As we discussed in the RC paper,<sup>9</sup> subscriptions show low (<1) coefficient of variation (CoV = standard deviation divided by average) for the behaviors we study. Thus, it is unsurprising that prior observations of the behavior are good indicators. Still, our results show that other attributes are also important: service type (the name of a top first-party subscription or “unknown” for the others), VM type (for example, A1, A2), number of cores, VM class (IaaS vs PaaS), operating system, and deployment time; their relative importance depends on the metric. VM role names have little predictive value, for example, IaaS VMs often have arbitrary role names that do not repeat.

Figure 4 illustrates the high accu-

accuracy of our current GBT models, which ranges between 74% (lifetime) and 87% (blackout time). The GBT prediction quality is even higher when we discard predictions with low (< 60%) confidence: precision ranges between 84% (lifetime) and 90% (average CPU utilization and blackout time) without substantially hurting recall, which ranges between 72% (lifetime) and 99% (blackout time). Again, for all behaviors, the most important attributes are the percentage of VMs classified into each bucket to date.

We expect the prediction quality our current models provide will be enough for most clients. For example, a VM scheduler that oversubscribes CPU cores prevents resource exhaustion as effectively with RC's VM utilization predictions as with an oracle predictor.<sup>9</sup>

*Accuracy by VM group.* Interestingly, accuracy can be higher for the first VM deployments from new subscriptions than for deployments from subscriptions we have already seen in the dataset. For example, for average CPU utilization predictions, these accuracies are 92% and 81%, respectively. We conjecture that this is because users tend to experiment with their first VMs in similar ways, so feature data accounting for prior subscriptions is predictive of new ones.

We also compare the prediction accuracy for third- and first-party VMs, and for first-party production and non-production VMs. The former comparison shows that accuracy tends to be higher for third-party VMs. For example, for lifetime, the accuracies for third-party and first-party VMs are 83% and 74% respectively, whereas for average CPU utilization they are 84% and 80% respectively. When comparing production and non-production first-party VMs, the results are more mixed. For lifetime, accuracy is higher for production VMs (82% vs. 64%), whereas the opposite is true for average CPU utilization (79% vs. 83%). The wide diversity of production workloads makes utilization more difficult to predict, but at the same time their lifetimes are less diverse and easier to predict as production VM tend to live long.

*Comparison to other techniques.* As baselines for comparison, we experiment with three techniques: most recent bucket (MRB), most popular bucket (MPB), and logistic regression (LR). MRB

and MPB are non-ML techniques. MRB predicts the bucket that was most common for the VMs in the last deployment of the same subscription (lifetime and average CPU utilization), the same bucket as the last deployment of the same subscription (max deployment size), or the same bucket as the last VM migration of a similar size (blackout time). MPB predicts the bucket that has been most popular since the start of the subscription. LR predicts a bucket based on a non-linear probability curve computed using the maximum likelihood method. We train the LR models with the same feature vectors we described earlier. Figure 4 shows that MRB exhibits accuracies between 54% and 81%, whereas MPB stays between 42% and 78%, and LR in the 62%–80% range. Clearly, these accuracies are substantially worse than our GBT results. Compared to MRB and MPB, GBT relies on many features instead of a simple heuristic, giving it a broader context that improves predictions. Compared to LR, GBT performs better for higher dimensional data. In addition, GBT combines decision trees with different parameters to produce higher quality results.

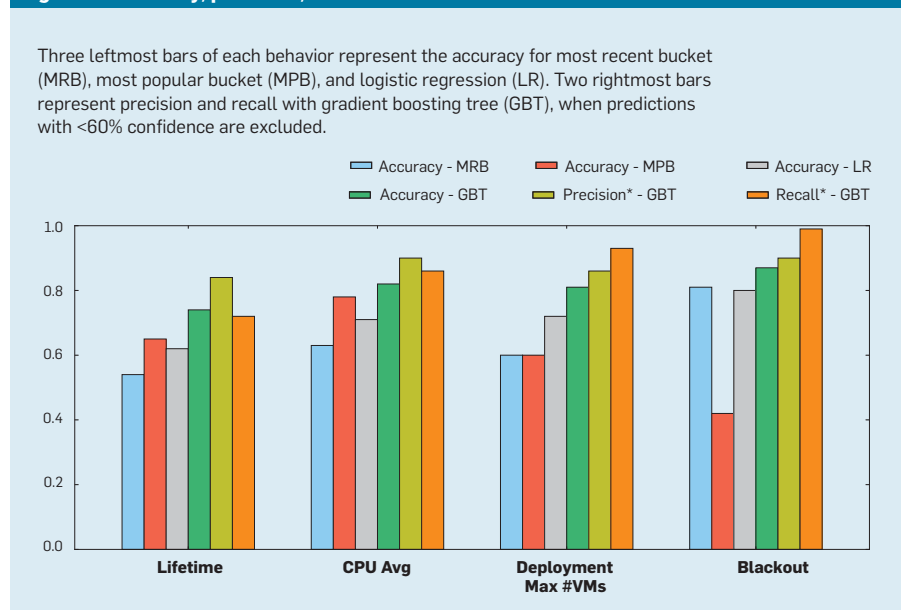
*Comparison to regression into buckets.* We also compare GBTs as classifiers into buckets with GBTs used for numerical regression and then bucketizing the results. We find that the former approach is substantially more accurate. The reason is that “noise” in the numerical values (for example, few VM de-

ployments that are exceptionally large) throw off the regression models and ultimately produce incorrect buckets.

**Initial uses and results in resource management.** In its first production instantiation, we have implemented RC's online component as a service in each Azure Compute cluster. A single version of its offline model-generation component runs on Cosmos. The first two major clients to use RC were the server defragmenter, which queries RC for lifetime and blackout time predictions (and VM metadata); and the container scheduler, which queries it for lifetime predictions (and metadata). As of March 2019, RC's clients are directing roughly 1.5 billion queries to it daily. The next major clients we will productize are the power capping manager, which will use RC's workload interactivity predictions; and a new predictive container rightsizing system, which will use RC's utilization predictions to recommend new container sizes. Several other uses of RC are being planned.

Our production results from the server defragmenter show that, from October 2018 to March 2019, RC enabled many tens of thousands of VM migrations, enabling more than 200 clusters (that would otherwise have been considered “full”) to continue receiving new VMs. Our earlier simulation study considered the use of RC-produced VM utilization predictions for safe core oversubscription.<sup>9</sup> It showed that an RC-informed oversubscribing VM

**Figure 4. Accuracy, precision, and recall for all behaviors.**



scheduler can accommodate more VMs, while producing 6× fewer cases of physical resource exhaustion than a baseline oversubscribing scheduler that does not consider utilization predictions.

**Lessons learned.** Thinking about ML-centric cloud platforms and through our experience with RC, we learned several important lessons:

*Separation of concerns.* Keeping predictions and management policies separate has worked well in RC. This separation is making policies easier to debug and their results easier to reproduce, as they are not obscured by complex ML techniques. In Azure’s current managers, policies tend to be rule-based and use RC predictions as rule attributes (for example, if expected lifetime is short, then place container in one of these servers). The rule-based organization has made it easier to integrate predictions into existing managers.

*Reach and extensibility.* The ML framework must act as a source of intelligence for many resource managers, not all of which will be known on day-one. Thus, it is critical to be able to easily integrate new data sources, predict/understand more behaviors, include multiple models for each behavior and implement versioning per model, among others. RC’s modular design has made these extensions easy.

*Model updates.* We designed RC to produce models and feature data offline, and then serve predictions and feature data online until it produces (in the background) an updated version of them. However, one resource manager we have come across requires models to be updated online, so that each prediction accounts for the effect of the previous one. As this scenario seems to be rare, we opted for RC’s more general design.

*Performance.* Many of the managers do not require extremely fast predictions. For example, the server defragmenter can easily deal with slow predictions. However, other managers require substantially higher performance. For example, the entire time budget for the container scheduler is less than 100 milliseconds. In these scenarios, prediction result, model, and feature data caching can be critical in prediction-serving, especially if models are large and complex. Caching also helps maintain operation even when the data store is unavailable.

*Integration with clients.* We explored two versions of RC’s online component:



**RC is by no means the only possible approach to ML-centric platforms. In fact, RC cannot currently accommodate certain types of ML integration that can be potentially useful.**



a first one was implemented as a runtime library to be linked with clients, and another as an independent service (as described). Interestingly, there is still no consensus on which implementation is ideal for Azure. Some teams like the ability to get predictions without leaving the client’s machine, which the library approach enables via RC’s caches. Other teams prefer the standard and higher-level interface of a service, and do not want to manage an additional library. Ultimately, we expect to build multiple online component implementations that will consume models and feature data from the same back-end source.

### **Open Challenges and Research Avenues**

As should be clear by now, RC is by no means the only possible approach to ML-centric platforms. In fact, RC cannot currently accommodate certain types of ML integration that can be potentially useful. Moreover, there are potential additional areas for ML integration that nobody has explored yet. Clearly, there is a need for more research on this topic. The following paragraphs identify some research challenges and avenues going forward.

*Broadly using application-level performance data.* As mentioned, low-level counters are an indirect measure of application performance. For resource management without performance loss, extracting high-level information from applications is key. Today’s extraction methods require effort from developers, who do not always have a strong incentive to provide the data. The challenge the cloud provider faces is creating stronger incentives or extraction methods that are automatic, privacy-preserving, and non-intrusive.

*Using action-prescribing ML while being general.* Increasingly popular ML techniques such as reinforcement and bandit learning prescribe actions. In the resource management context, this means the ML must understand the acceptable management mechanisms and policies (these techniques could define the policies themselves, but this would make manager debugging very difficult), and be adjusted for every manager that can benefit. Moreover, it must be safe/cheap to explore the space of available actions. The challenge is creating general designs for these ML techniques, perhaps via frameworks/APIs that take mechanism

and policy descriptions as inputs, so that they are easier to adopt.


**Quick feedback at scale.** When operating at scale, it is difficult to observe the result of predictions or management actions within a short time. For example, scheduling a large group of containers may impact the resource utilization and performance interference at many servers. Worse, other containers are constantly starting and finishing on these same servers. In this context, obtaining feedback from all the servers and isolating the impact of each prediction/action is extremely difficult. The problem is even harder when the feedback will only be available at an undetermined future time. For example, a container lifetime prediction can only be verified when the container finishes, which may take a long time. Thus, when quick feedback is needed, the challenges become determining when it is available, and accurately deriving it from the massive amount of collected data.

**Debuggability.** The cloud provider must be prepared for scenarios where an ML technique (and the managers that use it) suddenly starts to misbehave, producing poor predictions or actions. In these cases, regenerating models with more recent data may not be enough to fix the problem, because model features may have changed semantics or been eliminated altogether. Debugging misbehaviors is difficult for certain ML techniques, for example, neural networks, especially when they are tightly integrated with the manager. More research is needed on making debugging easier, if such techniques are to be used broadly in public clouds.

**Other aspects of cloud platforms.** Finally, our focus has been using ML in resource management for high efficiency and performance. However, ML can also benefit network traffic, availability, reliability, security, configuration management.<sup>4,13,28</sup> An open question is whether the ideas, designs, and trade-offs we discussed also apply to these other areas. In our future work, we will explore how to extend RC for these types management in a real platform.

## Conclusion

Public cloud providers invest billions of dollars into their software and hardware infrastructures. Maximizing the use of these expensive resources, while maintaining excellent performance and

availability, is critical for profitability and competitiveness. Infusing ML into cloud platforms has the potential to achieve these goals. In fact, we envision many opportunities and designs for bringing ML into cloud resource management. Taking advantage of some of these opportunities, we have been transforming the Azure Compute fabric to leverage predictions of container and server behaviors. This transformation has relied on Resource Central, our general ML and prediction-serving framework and system. Our initial experience shows that one can produce simple yet accurate ML models for many behaviors and enable resource managers to make smarter decisions. However, more research is needed on this topic, so we have made traces of the Azure Compute workload available for researchers to use at <https://github.com/Azure/AzurePublicDataset>. 

## References

- Abadi, M. et al. TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symp. Operating System Design and Implementation* (2016).
- Agarwal, A. et al. Making contextual decisions with low technical debt. *arXiv preprint arXiv:1606.03966* (2016).
- Amazon Web Services. Amazon CloudWatch; <https://aws.amazon.com/cloudwatch/>.
- Bodik, P., Griffith, R., Sutton, C., Fox, A., Jordan, M., and Patterson, D. Statistical machine learning makes automatic control practical for Internet datacenters. In *Proceedings of HotCloud* (2009).
- Calheiros, R. N., Masoumi, E., Ranjan, R., and Buyya, R. Workload prediction using ARIMA model and its impact on cloud applications' QoS. *IEEE Trans. Cloud Computing* 3, 4 (2015).
- Cao, R., Yu, Z., Marbach, T., Li, J., Wang, G., and Liu, X. Load prediction for data centers based on database service. In *Proceedings of the 42nd Annual Computer Software and Applications Conf.* (2018).
- Chaiken, R., Jenkins, B., Larson, P., Ramsey, B., Shakib, D., Weaver, S., and Zhou, J. SCOPE: Easy and efficient parallel processing of massive data sets. In *Proceedings of the 34th Intern. Conf. Very Large Data Bases* (2008).
- Chen, S., Shen, Y., and Zhu, Y. Modeling conceptual characteristics of virtual machines for CPU utilization prediction. In *Proceedings of the Intern. Conf. Conceptual Modeling* (2018).
- Cortez, E., Bonde, A., Muzio, A., Russinovich, M., Fontoura, M., and Bianchini, R. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In *Proceedings of the Intern. Symp. Operating Systems Principles* (2017).
- Crankshaw, D. et al. The missing piece in complex analytics: Low latency, scalable model management and serving with Velox. In *Proceedings of the 7th Biennial Conf. Innovative Data Systems Research* (2015).
- Crankshaw, D., Wang, X., Zhou, G., Franklin, M. J., Gonzalez, J. E., and Stoica, I. Clipper: A low-latency online prediction serving system. In *Proceedings of the 14th Symp. Networked Systems Design and Implementation* (2017).
- Delimitrou, C. and Kozyrakis, C. Paragon: QoS-aware scheduling for heterogeneous datacenters. In *Proceedings of the 18th Intern. Conf. Architectural Support for Programming Languages and Operating Systems* (2013).
- Fox, A., Kiciman, E., and Patterson, D. Combining statistical monitoring and predictable recovery for self-management. In *Proceedings of the 1st Workshop on Self-Managed Systems* (2004).
- Gao, J. Machine Learning Applications For Datacenter Optimization, 2014.
- Gong, Z., Gu, X., and Wilkes, J. Predictive elastic resource scaling for cloud systems. In *Proceedings of the Intern. Conf. Network and Service Management* (2010).
- Google. TensorFlow serving; <http://tensorflow.github.io/serving/>.
- Islam, S., Keung, J., Lee, K., and Liu, A. Empirical prediction models for adaptive resource provisioning in the cloud. *Future Generation Computer Systems* 28, 1 (2012).
- Ke, G. et al. LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems* 30 (2017).
- Khan, A., Yan, X., Tao, S., and Anerousis, N. Workload characterization and prediction in the cloud: A multiple time series approach. In *Proceedings of the Intern. Conf. Network and Service Management* (2012).
- Mao, H., Alizadeh, M., Menache, I., and Kandula, S. Resource management with deep reinforcement learning. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks* (2016).
- Microsoft Azure. Azure Monitor; <https://azure.microsoft.com/en-us/services/monitor/>.
- Moritz, P. et al. Ray: A distributed framework for emerging AI applications. In *Proceedings of the 13th USENIX Symp. Operating Systems Design and Implementation* (2018).
- Novakovic, D., Vasic, N., Novakovic, S., Kostic, D., and Bianchini, R. DeepDive: Transparently identifying and managing performance interference in virtualized environments. In *Proceedings of the USENIX Annual Technical Conf.* (2013).
- Rao, J., Bu, X., Xu, C.-Z., Wang, L., and Yin, G. VCONF: A reinforcement learning approach to virtual machine auto-configuration. In *Proceedings of the 6th Intern. Conf. Autonomic Computing* (2009).
- Roy, N., Dubey, A., and Gokhale, A. Efficient autoscaling in the cloud using predictive models for workload forecasting. In *Proceedings of the Intern. Conf. on Cloud Computing* (2011).
- Yadwadkar, N. J. *Machine learning for automatic resource management in the datacenter and the cloud*. Ph.D. thesis, UC Berkeley, 2018.
- Zhang, Y., Prekas, G., Fumarola, G. M., Fontoura, M., Goiri, I., and Bianchini, R. History-Based harvesting of spare cycles and storage in large-scale datacenters. In *Proceedings of the Intern. Symp. Operating Systems Design and Implementation* (2016).
- Zheng, W., Nguyen, T. D., and Bianchini, R. Automatic configuration of Internet services. In *Proceedings of the 2nd European Conf. Computer systems* (2007).

**Ricardo Bianchini** is a Distinguished Engineer at Microsoft Research, Redmond, WA, USA.

**Marcus Fontoura** is a Technical Fellow at Microsoft Research, Redmond, WA, USA.

**Eli Cortez** is a Principal Engineer at Microsoft Research, Redmond, WA, USA.

**Anand Bonde** is a senior engineer at Microsoft Research, Redmond, WA, USA.

**Alexandre Muzio** is a software engineer Microsoft Azure, Redmond, WA, USA.

**Ana-Maria Constantin** is a software engineer Microsoft Azure, Redmond, WA, USA.

**Thomas Moscibroda** is a partner research scientist at Microsoft Azure, Redmond, WA, USA.

**Gabriel Magalhaes** is a Ph.D. student at the University of Washington, and was an intern at Microsoft Azure during this work.

**Irish Bablani** is corporate vice president of Microsoft Azure, Redmond, WA, USA.

**Mark Russinovich** is a Technical Fellow and CTP at Microsoft Azure, Redmond, WA, USA.

© 2020 ACM 0001-0782/20/2



Watch the authors discuss this work in the exclusive *Communications* video. <https://cacm.acm.org/videos/ml-centric-cloud-platforms>

**Future PSM systems will require diversity-enhancing yet contextually sensitive designs.**

BY THOMAS OLSSON, JUKKA HUHTAMÄKI, AND HANNU KÄRKKÄINEN

# Directions for Professional Social Matching Systems

SUPPORTING HUMAN COLLABORATION has been a central driver of the development of information and communication technology. A relatively recent approach to this end is social matching, referring to computational ways of identifying and facilitating new social connections between people.<sup>36</sup> Social matching is most often connected with partnering for leisurely and romantic relationships—in fact, the most well-known social matching systems revolve around dating scenarios (for example, Tinder<sup>a</sup>) or triggering opportunistic interactions with strangers (for example, Happn<sup>b</sup>).

a <https://tinder.com/>

b <https://www.happn.com/en/>

This article focuses on Professional Social Matching (PSM), which we define as the matching of individuals or groups for vocational collaboration and co-creation of value. This covers organizational activities, including recruitment, headhunting, community building, and team formation within or across organizations as well as individually driven activities like mentoring, seeking advisory relationships, and general networking.

From a technological perspective, computer-supported PSM is based on computational approaches to profiling actors (organizations and individuals), modeling their qualities, analyzing their mutual social suitability and relevance, and presenting the recommendations to the users. For example, prescriptive data analytics<sup>9</sup> can utilize social network analysis (SNA) for explicating the social ties between actors and machine learning-based approaches to analyze their competences and interests and to identify suitable pairs of actors. The resulting computational system can manifest as proactive

## » key insights

- Professional Social Matching (PSM) is an emergent and potentially very impactful area of social matching systems, building on recommender systems, decision-support systems, social network analysis, and machine learning.
- Mindful of the ethics of computationally influencing professional matching activities, such as team formation and networking, the current computational approaches for profiling, matching, and recommending actors must be reconsidered.
- Future PSM systems should aim to enable unexpected encounters and social serendipity, incorporate a systemic perspective to the matching logic, help avoid human bias in decision-making, and identify optimal similarity-diversity trade-offs between actors.
- We argue that future PSM systems must be calibrated for different matching cases rather than for individual users; be based on multidimensional analytics of profiles and contextual data; support meaningful cooperation with the end user; and, feature proactive nudging to help the users avoid inherent human prejudice.



people recommender systems—or social recommender systems<sup>15</sup>—or other kinds of data-driven decision-support systems<sup>31</sup> for social matching. Conventional examples include recommendations of other professionals to follow in various social media services as well as more specific expert search systems.<sup>40</sup>

This fusion of various analytical and algorithmic approaches for a broad and multifaceted application area forms an intriguing, interdisciplinary space for research and development of novel algorithms and information systems. This article reviews relevant literature in three key scientific domains that present important understanding necessary to reach the potential of future computational systems (as illustrated in Figure 1).

Our synthesis is based on two central yet paradoxical ideas. On the one hand, computational support for choosing collaborators seems beneficial because purely human-driven matching is prone to limitations and biases in human decision-making and

the boundedly rational understanding of the breadth of alternatives. On the other hand, algorithmic matching involves risks of strengthening the human biases, for example, through biased training data,<sup>29,39</sup> providing socially unacceptable or seemingly illogical recommendations or introducing unanticipated detrimental mechanisms to professional collaboration and social structures.

To address the problem of role differentiation between human and computational reasoning in PSM decisions, this article builds on recent abundant discussions on the ethics of information technology (for example, O’Neill<sup>29</sup> and Shilton<sup>33</sup>). We review and problematize some premises in current paradigms of system design that might also seem intuitively appropriate when developing PSM systems. While the academic community has already acknowledged that simplistic aims and metrics, such as prediction accuracy in recommender systems, might misguide a research field at large,<sup>26,37</sup> we

follow a similar critical line of thinking on a broader scale. We underline the importance of area-specific application and systemic consideration when defining design goals and computational approaches.

### Motivating the Computational Approach

Why is computational support necessary in PSM? Traditionally, the identification and choice of professional partners take place manually by individuals (for example, matching an employer with a suitable employee, or choosing with which peers to collaborate). Long known in the science of cognition and reasoning, much of human decision-making is limited by the capacity of information processing (that is, bounded rationality<sup>34</sup>), is inclined to be based on intuition, heuristics, and cognitive shortcuts,<sup>19</sup> and strives for minimizing cognitive effort.

In choosing collaborators, human decision-making can result in tendencies like homophily, the preference

of interacting with like-minded others,<sup>21</sup> and leaning on existing social networks and a geographically limited pool of candidates. In social networking, people tend to strengthen existing social clusters (that is, triadic closure<sup>14</sup>) rather than reach for unknown communities and individuals. Similarly, formation of working groups within organizations is often based on arbitrary choices even though the combination of people can significantly influence group productivity and satisfaction.<sup>35</sup>

In other words, human biases can result in suboptimal collaboration and untapped co-creative potential, particularly in knowledge work and creative industries that demand cross-pollination of ideas and perspectives. It has been argued that fruitful collaboration and high innovation capability result from complementary viewpoints among a diverse group of actors.<sup>27</sup> Substantial research in management science and information systems hint that, in particular, activities that require divergent thinking in groups benefit from diversity<sup>2</sup> (for example, startups pursuing innovations or corporation boards aiming at holistically optimal strategic decisions). Heterogeneity in terms of knowledge has also been found important to managerial performance and in particular to innovation performance.<sup>32</sup>

While the jury is still out on the optimal balance of collaborators on the similarity-diversity continuum (for example, Aral et al.<sup>4</sup>), we argue that current computational solutions strengthen similarity-seeking behavior. Furthermore, it is noteworthy that diversity is not only about differences in identity-related qualities like gender, ethnicity, and personality types; it is also about cognitive diversity in terms of skills, competences, knowledge, and interests, and it is about social diversity in terms of social behavior and networks.

The need to identify optimal collaborators and fruitful skill combinations will increase in the future as dynamism in work life is expected to increase and co-creation chains turn into networks with increasingly complex structure. Examples of relevant trends and phenomena include increasing emphasis on ad hoc freelancer groups in creative industry, micro entrepreneurship, and

piecework<sup>3</sup> as well as strongly interdependent actors in business and innovation ecosystems. The more interdependency and collaborative value creation there is on a systemic scale, the more important and impactful the decision-making on social matching becomes.

### Opportunities for Computational PSM

The question of how computational systems could support PSM matching in particular remains relatively unexplored when considering the breadth of collaborative professional activities. Prior research and development work have addressed expert search (for example, Wang et al.<sup>40</sup>), recommendations for inspiring individuals or organizations to follow in social media, and machine learning solutions for identifying suitable candidates in recruitment and headhunting (for example, Faliangka<sup>11</sup>). Recommending new collaborators has been explored in a few algorithmic experiments and user studies (for example, Tsai et al.<sup>37</sup>).

However, this vein of research remains on the fringes of recommender systems research and is limited to the context of conferences (for example, Chen et al.<sup>8</sup>) or analyzing publication data in academia (for example, Pham et al.<sup>30</sup>). As for supporting teamwork, computational solutions have been explored in, for example, optimizing the organizing and managerial practices of a team.<sup>42</sup> However, the question of the ideal team composition remains underexplored. At the same time, the new technical enablers and current trends in work-life allow for novel PSM solutions. The amount of data on people and organizations is increasing, which has paved the way for the rapid evolution of machine learning-based techniques. The aforementioned trends in leadership and organizing knowledge work welcome new forms of collaboration, and networking in general is highly valued by both organizations and individuals.

Consequently, we have recently witnessed the birth of commercial applications for browsing candidates for professional interaction (for example, Brella,<sup>c</sup> Grip,<sup>d</sup> and Shapr<sup>e</sup>). Such ap-

plications tend to follow the so-called Tinder logic of user-based selection of seemingly interesting candidates based on simple profiles, often restricted to matching attendees at professional events. However, the simplistic and similarity-seeking nature of such applications calls for ethical reflection and expansion of the horizon in this area. We envision that future PSM systems could offer solutions on different levels of matching:

- ▶ Identifying optimal combinations of professional qualities and aims in certain professional activities (that is, matching skills and goals).
- ▶ Recommending partners for particular co-creative purposes, such as for business partnerships or mentoring relationships (that is, matching individuals).
- ▶ Optimizing team formations for a project (that is, matching multiple actors).
- ▶ Identifying suitably complementary actors for networked value creation (such as, matching at ecosystem level).
- ▶ Balancing the supply and demand in the job market by suggesting dedicated trainings or new job openings (such as, matching on societal level).

To provide an early framework of PSM activities, we identify three main tracks of social matching in professional life: one-to-one, one-to-many, and many-to-many (Figure 2). These categories display different levels of decision-making complexity and varying numbers of qualities to analyze.

Specific matching cases further vary in terms of the temporal nature of the decision: for example, long-term commitment in recruitment of a new employee vs. short-term perspective in forming a working group within an organization; intensity of collaboration: for example, choosing a new group member for a co-creative project vs. choosing a member for a young company's advisory board; and, probability and criticality of risk: for example, low-probability but high-cost risk of making an unsuccessful recruitment vs. high-probability but low-cost risk of extending one's personal network with new individuals. The bottom part of Figure 2 relates particularly to this latter aspect of the cost of suboptimal matching.

These qualities set boundaries and requirements for PSM system design,

c <https://www.brella.io/>

d <https://grip.events/>

e [www.shapr.co](http://www.shapr.co)



for example, in terms of the breadth of given alternatives (variety of matches), the need for transparency and explanatory capability of the logic behind recommendations, and the agency and role that a system has in the decision-making. In Figure 2, the recommendations related to increased complexity and cost of failure imply higher risks and require increased explanatory power of recommendations from PSM systems. Deciding whom to meet at an event has low risks as the interaction would be short term and of low intensity; therefore, a user might be satisfied with only a few algorithmic recommendations and superficial reasoning behind the recommendations. However, matching for which actor to choose for close business collaboration poses higher risks and costs of failure. For the user to trust and follow the recommendations, this requires both more in-depth algorithmic reasoning and better explanation capabilities in the user interface.

**Pitfalls in Computational PSM**

A key argument of this article is the fundamentals of state-of-the-art technologies and approaches that could be utilized to build PSM systems—for example, item recommenders, social network analysis, and machine learning—must be reconsidered in this application area. Directly applying the prevailing analysis or design patterns to PSM can introduce new risks with detrimental effects on the performance and collaboration practices of knowledge workers. The following critically reviews the suitability of common computational approaches.

First, it is imperative to realize that recommendation does not equal prediction,<sup>26</sup> particularly in PSM. To truly enhance professional collaboration, recommender algorithms should not reproduce or strengthen the biased human behavior. For example, in a recruitment system, using prior examples as training data for machine learning is expected to strengthen the demographic distribution that a certain organization or profession has traditionally had. As McNee et al.<sup>26</sup> pointed out, an accurate recommender engine might produce recommendations that are formally relevant as predictions, yet not very useful as recommendations

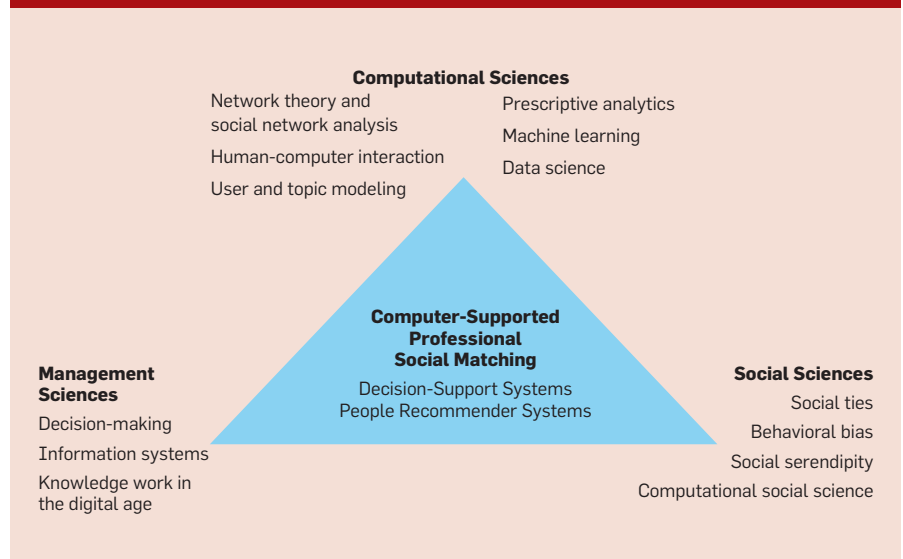
(for example, a well-connected person is recommended to most users).

Wallach gave a gender-related example of the same issue: “There is a substantial difference between a model that is 95% accurate because of noise and one that is 95% accurate because it performs perfectly for white men, but achieves only 50% accuracy when making predictions about women and minorities.”<sup>39</sup> Similarly, it would be straightforward to predict who someone might meet at an event based on their history of professional social encounters in similar situations. However, using that as a recommendation would strengthen their habitual behavior, which might be against their actual collaboration needs.

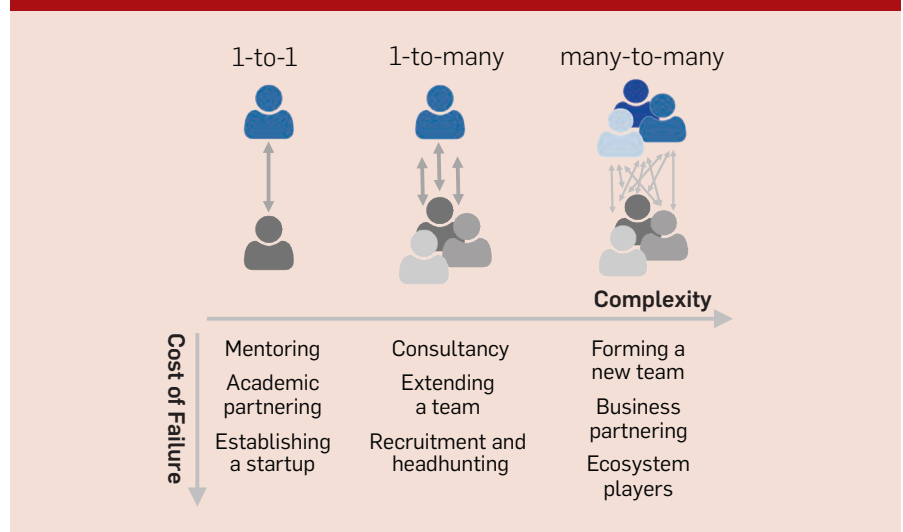
Second, social matching systems tend to look for maximal similarity; in fact, dating scenarios have particularly been found to benefit from emotional similarity.<sup>13</sup> A perfect match in dating services refers to the similarity of profiles, typically based on analyzing a simple user-defined, property-based profile content and clustering the pool of actors. This logic also seems to have affected the current vocational matching services: shared qualities tend to be highlighted in the user interface, and the brevity of profiles can lead the user to follow the natural tendency of seeking for similarity.

Third, from the perspective of social ties and networks, contemporary

**Figure 1. Overview of relevant scientific domains, concepts, and research areas to develop next-generation computer-supported PSM.**



**Figure 2. Main tracks of PSM, with examples of matching cases with different scales of cost upon a suboptimal matching decision. The complexity of matching decision-making increases as the number of actors increase.**



approaches for analyzing connections between individuals often utilize social network analysis and link prediction.<sup>24</sup> Following the triadic closure hypothesis, new ties are more likely to be formed between friends-of-friends or colleagues-of-colleagues,<sup>10</sup> that is, between actors that share a strong connection. The triadic closure mechanism can, however, enforce echo chambers and increase polarization—the typical pitfalls of social networking services in the 2010s. In knowledge work, we argue the narrowed thinking due to echo chambers is bound to reduce exposure to novel information and decrease divergent thinking and innovation capability.

Fourth, PSM system designers must consider that good matches cannot be generalized across individuals. In content or item recommender systems, the same news article or product is often recommended to several users with corresponding consumption behavior and ratings—that is, collaborative filtering.<sup>20</sup> However, in PSM, person A being a good partner for person B does not imply that A would also be a good match for person C, even if B and C had similar qualities. An optimal match in professional life is very case specific and determined by, among other things, the matched actors' current needs, interests, personality, and availability for collaboration. Matches

can be generalized only across similar cases. This means that narrowing down only to similar collaboration cases can lead to data sparsity and that the collaborative filtering approach would suffer from cold-start issues.

Finally, as professional activities are related to value creation for organizations—and, more broadly, communities and societies—PSM calls for a systemic perspective. For example, the same central and active individuals cannot practically be recommended to everybody (that is, the Matthew effect). The mechanism of preferential attachment<sup>5</sup> tends to lead to power-law distribution across the population. A system might recommend excessive collaboration opportunities for people that already have plenty of connections while undervaluing other criteria, such as urgency of the need for collaboration or the actors' practical capacities for exploring new collaboration opportunities.

### New Design Directions

The limitations in human decision-making and the pitfalls in the traditional computational approaches essentially imply that PSM is far from trivial bulk predictions targeted to masses. Neither the matching mechanisms in dating applications nor the analytics methods in contemporary content recommender systems and link prediction algorithms seem to fit with

the characteristics of PSM. To turn the focus from critique toward constructive thinking, we next propose high-level goals for the future generations of PSM systems. These are intended to advance this nascent research area toward socially meaningful and ethically sustainable design directions.

**Goal: Balance diversity and similarity.** Comparing to leisurely social matching applications or content recommenders, we argue that PSM systems should employ more diversity-enhancing approaches that shift the current aim for *convergence* toward *divergence*. While diversification has been recognized as a relevant aim for recommender systems in general,<sup>23</sup> matching people for professional collaboration makes a particularly strong case for this.

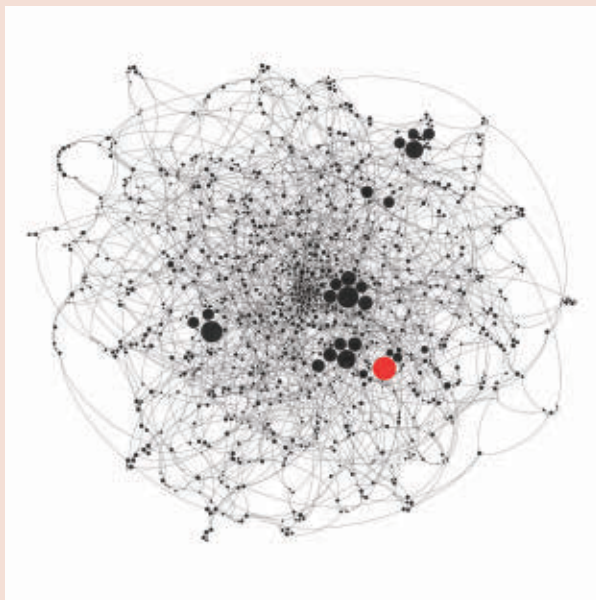
At the same time, neither optimizing for similarity nor diversity should be taken as a maxim: the optimum is a moving goal somewhere between these extremes. Successful collaboration requires mutual trust and shared working culture (that is, similarity as a consolidating, convergent power) as well as openness for change and different perspectives (that is, diversity as a divergent power). Similarity is useful when the aim is to validate ideas or methods and there is a need for agility in short-term collaboration. Strong common denominators can also open minds to appreciate individual differences. Diversity is beneficial in developing novel ideas with the help of complementary perspectives, establishing long-term business ventures utilizing complementary social capital, or making well-informed strategic decisions based on diverse knowledge.

Identifying the optimal balance requires analysis and modeling of not only the actors but also the characteristics of the intended collaboration and the collaboration context. Compared to leisurely matching, professional life is more dynamic in terms of interaction needs, interests, and resources at different times.

**Goal: Enable experiences of social serendipity.** We call for systems that help people make professional matching decisions with positive long-term benefit. We argue that the experience of *serendipity* is an indicator of successful knowledge work, making it a desir-

**Figure 3. Illustrative example of matchmaking potential within a community.**

Nodes represent actors that are interconnected through collaboration, and node size represents similarity between actors' interests. Several existing clusters of actors similar to the active user (in red) are identifiable either computationally or manually through an interactive visualization.



able goal for designing PSM systems. *Social serendipity* can be considered as a strong experience of both unexpectedness and instrumental benefit from social encounters and collaboration.<sup>25</sup>

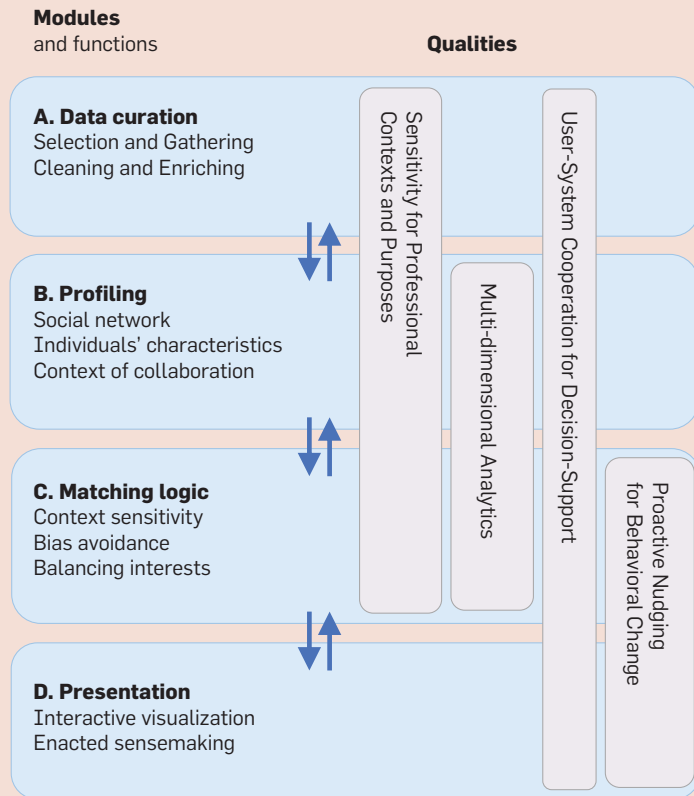
The unexpectedness can arise from encounters with people outside one's conventional social circles or with seemingly different qualities than oneself. However, turning such *chance* into serendipity calls for features that help the user to identify the possible benefits and gain advantage of the offered chance. The element of benefit can result from collaboration that provides value beyond what one could personally attain, for example, because of having complementary skills or knowledge or having formed an unparalleled team. Similar thinking can be identified in content recommenders where it is necessary to establish certain level of familiarity for the user and, at the same time, support the discovery of interesting new content.<sup>22</sup> However, as social serendipity involves several actors, it is more challenging to design for it than for *information serendipity*, the more common aim of content recommender systems.

**Goal: Support a systemic perspective in defining ideal matches.** We argue that matching algorithms should not only optimize matches for an individual user but also consider what is ideal on a systemic level: across the user population and social structures like organizations. While social link prediction is typically based on relationships on the local scale, PSM systems introduce an opportunity to intervene the evolution of social networks to optimize for systemic diversity.

For example, in individual fields, a system should avoid reinforcing the processes of increasing homogeneity<sup>21</sup> and preferential attachment<sup>5</sup> by reproducing existing social network evolution mechanisms and recommending the same central actors to everyone. Instead, the recommendations need to be considered on an ecosystemic or even global level (for example, within a profession-based community). Organizations must balance the workload across employees; the most likable and versatile individuals cannot practically contribute to all the groups or organizational actions. Furthermore, the actors' interdependency demands bidi-

**Figure 4. Key modules and functions in a high-level system architecture and the system qualities mapped to the different modules along the analytics pipeline.**

(A) Data curation refers to functions of gathering, cleaning, and managing data and enriching these with relevant semantic labels. (B) Profiling covers the functions for deriving insight from the data in order to truthfully present and compare different actors and matching cases. (C) Matching logic is about the algorithmic procedures for defining and predicting which social combinations would be most appropriate in a given matching case. (D) Presentation refers to the user interface and human-computer interaction solutions that support sensemaking, comparison, and well-informed decision-making.

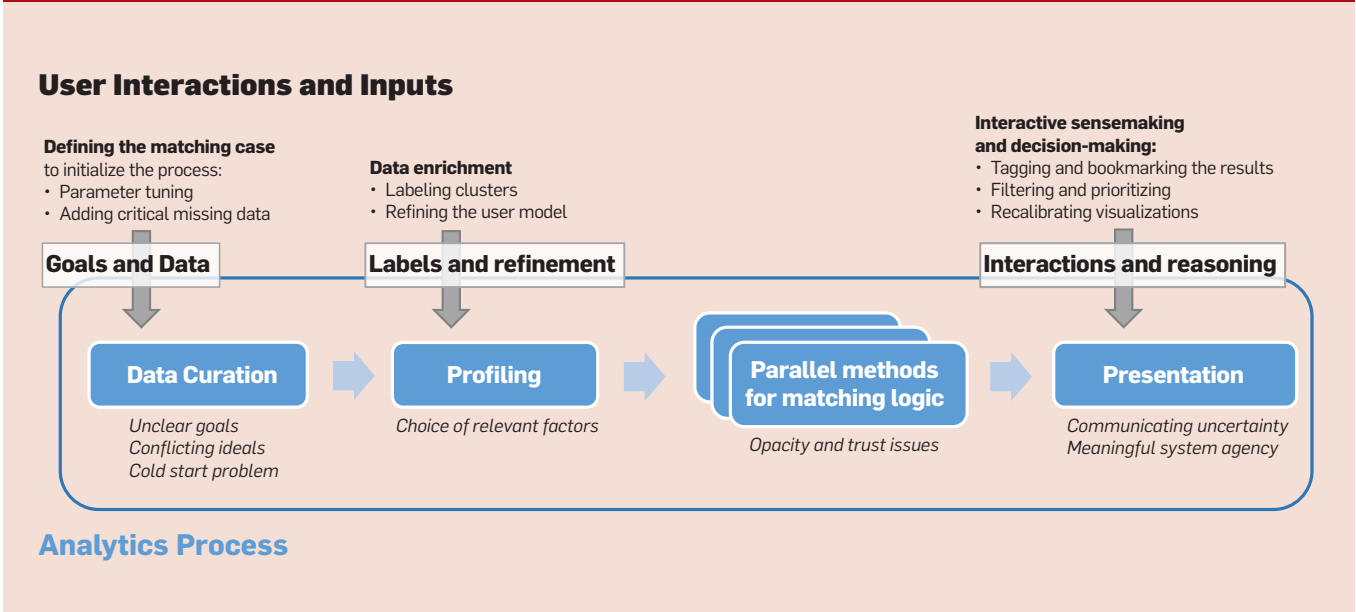


rectional optimization: while person A could seem like a beneficial match to person B, person B might not perceive person A relevant enough. Matches that are optimized only for an individual can create collaboration proposals with asymmetric benefit and thus remain inefficient or be rejected.

**Goal: Support the utilization of the existing social structures.** We urge PSM developers to take advantage of existing social structures in organizations. First, existing weak or dormant ties (for example, people who already know each other but lack understanding of all the mutual collaboration opportunities) can be re-introduced through recommendations. Second, from an organizational perspective, diversity-enhancing matching should enable making use of existing echo chambers

by identifying potential pairs of actors that are able to connect and bridge the flow of knowledge between established clusters (as illustrated in Figure 3).

This objective differs fundamentally from the practices of straightforwardly predicting the likelihood of the formation of new social connections.<sup>24</sup> We point to Burt<sup>7</sup> who discussed social capital from two viewpoints: whether scarce or dense social networks produce social capital. Weak ties that gap structural holes can increase creativity and support the career development of those that occupy bridging roles. At the system level, these brokers serve as conduits of information flow. A related matching strategy that contributes to system-level diversity is identifying *tertius jungens* (or “third who joins”), that is, individuals that can serve as pro-

**Figure 5. Outlining relevant user interactions in the human-in-the-loop analytics process. Typical research and design challenges in italics.**

ies in introducing “disconnected individuals or facilitating new coordination between connected individuals.”<sup>28</sup> Utilizing social proxies can be expected to facilitate building trust between the apparently diverse actors.

### PSM System Qualities and Future Directions

To address the aforementioned goals, we introduce a model of the analytical elements that we consider crucial for future PSM systems. Figure 4 outlines a high-level system architecture with key modules, inspired by Terveen,<sup>36</sup> and their main functions. These are related to an analytics pipeline as well as general system qualities that need to be addressed across the presented modules. With this overview, we underline key requirements for future work and relevant research directions.

*Sensitivity for professional contexts and purposes.* Perhaps the most analytically challenging requirement is that systems should model not only the potential actors and their social structure but also the context of the intended professional relationship. The produced models should drive tailoring of the matching logic accordingly (for example, changing priorities and weights of variables). The notions of context and context-awareness have been extensively discussed across the sciences, and various contextual variables have been investigated in relation to information systems in general

and recommender systems in particular (for example, location, culture, user personality, task context).<sup>1</sup> In what follows, we focus on a few aspects that are particularly relevant when considering the matching logic.

The quest for a sweet spot between maximal diversity and similarity is already challenging per se. Furthermore, it is a moving, context-dependent target. The optimal degree of diversity depends on individuals and the targeted collaboration in question. For example, the end user’s personality has been found to affect the readiness for accepting diversity of recommendations in content and item recommenders,<sup>41</sup> and we expect that such inherent human factors also have an equally significant role in people recommenders. As for organizations, characteristics like openness, tolerance of difference, and the different purposes of collaboration presumably also have an effect.

Similarly, an understanding of the current social structures is needed to identify meaningful configurations of the matching logic. Our premise is that a typical structure is composed of clusters that are densely interconnected but loosely connected to other clusters. Consequently, we suggest that PSM developers draw from the diversity-bandwidth trade-off theory<sup>4</sup> to navigate the design space. Ties of different strengths (weak–strong) all have potential as conduits of novel information but in different ways in

different domains of work. For example, the highly dynamic environment of a startup company highlights the importance of strong ties as high-bandwidth sources of relevant information. At the other end of the spectrum, an academic research group produces new knowledge over years rather than days. Therefore, the actors are able to invest in utilizing low-bandwidth weak ties due to their potential in serving non-redundant, novel knowledge.

In sum, many of these aspects represent largely unobservable contextual factors that are subjectively defined. This forms a cyclical relationship between context and collaboration where the collaboration activity gives rise to context and the context influences activity.<sup>1</sup> Moreover, the context forms as a mixture of several individuals’ and organizations’ contextual characteristics. Formalizing this contextual variance requires new methods and frameworks for conceptualizing PSM in more detail. This also means the plethora of data needed to reach certain analytical targets is unpractical without new mechanisms of granting access to personal data. PSM development endeavors need to carefully consider which practically available data can best support the context-specific analytical goals and how to compensate missing data.

We suggest there are two alternative design strategies to enable context sensitivity in PSM systems. *Reactive* design

leaves the active user in charge of driving the system in a way that contextual requirements are met. *Proactive* design follows the logic of prescriptive analytics where the system suggests actions or even takes them on behalf of the active user. Whereas the proactive design introduces higher requirements for data availability and quality, it also provides a stronger basis for pursuing serendipity and diversity.

*Multi-dimensional analytics.* As noted, contextuality and the goal of supporting systemic perspective call for multidimensional analysis with several parallel relevance algorithms and various data about the actors. Extant research on recommender systems shows that the perceived relevance of recommendations and the perceived usefulness of the recommender system increase when multiple different recommendation strategies are in parallel play. For example, Hupa et al.<sup>18</sup> discuss the advantages of multidimensional social-network recommenders to increase interdisciplinary collaboration. Tsai and Brusilovsky<sup>37</sup> used four different recommender engines to support identifying new academic collaborators using conference data (topic similarity, social similarity, interest similarity, and geographical distance) and showed that the users who are able and willing to use multiple engines in parallel receive more relevant recommendations.

Although the triangulation of these approaches can already help identification of relevant matches, we call for consideration of additional perspectives in the prescriptive analytics for people recommendations. For example, geographical distance must be considered, especially in long-term collaboration. Level of seniority (or expertise) can affect the preferred symmetry of benefit and trust: matching for mentoring or advisory relationships calls for high difference in expertise, while matching for a production team often demands relatively equal levels of seniority. The personalities and organization cultures need to be similar enough to enable matches that are sustainable in the long term (for example, no conflicts due to drastically different ways of working or level of commitment).

Overall, developing analytics procedures and recommender engines

for different perspectives is a step toward so-called *hybrid recommender systems*.<sup>37</sup> At the same time, this introduces practical challenges. In addition to the axiomatic data availability issue, defining the logic in which the different analytical functions are combined in a context-sensitive manner requires deliberate research. Because PSM can potentially be affected by such a broad range of human features, all of them cannot practically be embedded in the algorithm design. We can only call for multidisciplinary research collaboration where social scientific understanding would support the identification of top-priority factors and formalizing this vast space.

*User-system cooperation for decision support.* Due to the dynamic and inherently complex nature of PSM, we argue that the matching decisions cannot merely be automated or offloaded to algorithms. For example, machine learning generally produces relevant results only if initialized with good-quality training data and well-defined goals, whereas human reasoning is suited for multi-faceted challenges where the desired pattern is unknown a priori. The different limitations and strengths in the human and computational analytical capabilities call for effective collaboration between computational intelligence (*deep yet narrow*) and human intelligence (*broad yet shallow*). This relates to the general notions of augmented intelligence and human-in-the-loop thinking. As this approach has already shown its power in, for example, classification problems, the complexity of PSM offers an even more opportune application area.

The human-in-the-loop approach has been envisioned<sup>f</sup> as useful, for example, when: (a) the cases that need to be identified are rare (class imbalance, for example, rare type of collaboration); (b) the cost of error is high (for example, time spent on browsing irrelevant matching options); (c) human annotations are already used (for example, recruiting processes); and (d) generic pre-trained models exist but need to be customized. The holistic thinking and contextual adaptability of the user are needed, for example, to steer the deep yet narrow algorithms (for example,

refinement of what an ideal match is, or prioritizing the sought features for each matching case) and to make sense of and choose between the resulting recommendations. Particularly when considering non-experts as users of recommender systems, we need user interface solutions that support decision-making with alternative options, a multi-dimensional systemic viewpoint, and ways to communicate and deal with the algorithmic uncertainty that this application area entails.

Figure 5 outlines the potential collaboration points along the computational analytics process. First, we need methods that guide the user in selecting appropriate training data and analytical goals to enable accurate profiling and, eventually, predictions. Semi-supervised approaches could allow training with a very small amount of labeled training data, minimizing both the *cold start problem* and need for manual work by the user. Second, we need feedback from the user regarding which factors are of top priority in the current matching case. Third, we need to support user exploration for enacted sensemaking, which has been found to be important both in visual analytics in general<sup>6</sup> and visual network analytics in particular.<sup>17</sup>

Subscribing to the call for transparency in AI and algorithmic systems, we argue that also PSMs should be able to better explain the reasoning behind the recommendations. For example, the uncertainty of the prediction should be translated into human-comprehensible forms so that the user can trust them, is able to assess the factual relevance of the recommendation, and can adjust their preferences accordingly.

Throughout the process, the cost in terms of burdening the user must be in line with the benefits of using the system. This raises the question of what user input and feedback is sufficient to hone the algorithms while causing minimal burden with tedious tasks for the user. In other words, it introduces a trade-off between the certainty of the matching decision and invested user effort.

An example of interactive visualization that can support exploration of relevant other people is the Conference Navigator.<sup>38</sup> The recommended conference talks are presented as sets, each of which are identified by a recom-

<sup>f</sup> <http://bit.ly/2mvBU5m>

mendation engine. The active user is able to explore different recommendation strategies and their combinations through an interactive interface. This kind of hybrid recommender system with several visualization approaches supports user-driven exploration.

In the long term, following Leifer's Hu-mimesis epiphany,<sup>8</sup> we envision future PSM systems taking the form of collaborative robots, that is, *cobots* or socially interactive agents working alongside the user. The user could delegate some of the decision making to a personal agent that, over time, builds an increasingly accurate model of the user and their preferences and typical cognitive biases. This could allow maximally automated tasks (for example, continuous pre-selection of relevant candidates among the whole population and recommending them at an opportune moment), implementing the idea of prescriptive analytics.<sup>9</sup> Such automatic modeling of other people and their suitability naturally poses challenges considering not only data availability but also data protection regulations and other ethical issues, which can become greater hindrances than the engineering of such mechanisms.

*Proactive nudging for behavioral change.* While the previous section revolved around how the user steers the system, user-system cooperation also touches the system's influence on user behavior. To address the goals regarding serendipity, optimal diversity, and systemic perspective, we expect that typical end users need to be nudged away from their habitual and individually oriented preferences in networking. The notion of *diversity-sensitive design* can be considered to mitigate trust issues in exposing the user to diversity.<sup>16</sup> To this end, the concepts of persuasive computing<sup>12</sup> and gamification have been long studied in human-computer interaction as mechanisms for supporting behavioral change. While the general approaches have been successfully applied in application areas like exercising and education, it remains an open question how user interfaces could meaningfully support and affect



## Proactive systems for PSM pose new challenges for user interface design as they manifest design ethics beyond the will of a user—the traditional fundament in human-computer interaction.



the rather intimate decisions around professional collaboration.

Matching people and interests is already enabled by various digital platforms and social media. However, to date, there are few solutions that would take a more explicit and proactive role in social matching (for example, recommending a person to meet someone when the situation is opportune). In content recommender systems, the recommendations are typically provided while the user is actively using the service. However, PSM calls for technology that takes a more active role and has its own judgment of what are optimal matches—that is, a moral stance—as well as when and where to present the recommendations.

Proactive systems for PSM pose new challenges for user interface design as they manifest design ethics beyond the will of a user—the traditional fundament in human-computer interaction. The computational design ethics would need to balance between an individual's preferences (that is, their habitual behavior) and the greater good for an organization, business ecosystem, or other social entity (that is, systemic perspective). This requires careful encoding of the ideals and mechanisms for the user to adjust them when necessary. Proactive PSM also requires new interface design solutions that sufficiently preserve the user's agency to maintain trust and acceptance.

The conventional subtle ways of enabling serendipity in content recommenders, such as adding randomness to the results or removing top-ranked results, might not be enough for PSM systems. Rather, this application area calls for solutions that avoid bias and prejudice. For example, relevant approaches could include hiding a recommended person's profile picture or name (as strong indicators of gender, ethnicity, and socioeconomical stance) or hiding details about professional background and education. Revealing some commonalities is important for building interpersonal trust, whereas the relevance of the complementary qualities needs to be considered with respect to the collaboration need at hand. Rankings and ratings tend to encode human bias,<sup>29</sup> which means the presentation of the matches should focus on qualitative descriptions rather than numeric ones.


<sup>8</sup> Hu-mimesis: Design Requirements for Personal Relevance, <https://www.youtube.com/watch?v=pg0xU-6PQII>

Persuasion can also manifest itself via facilitation of forming the connection. For example, in an event context with low-risk matching, the matched actors can be given a playful challenge to explore the commonalities and complementarity by not revealing the exact reason behind the recommendation, such as in Chen et al.<sup>8</sup> Here, the human capabilities in identifying possible matches could also be harnessed beyond the primary end user, for example, encouraging a third party to serve as a matchmaker between people they know, thus facilitating newly recommended connections and validating the ill-reasoned recommendations that an algorithm might provide.

## Conclusion

Social matching is ubiquitous in professional life, yet suboptimally supported by computational systems. Based on a multidisciplinary review of the literature, we outline the complex problem space of Professional Social Matching, and we define design goals for computer-supported PSM and requirements for developing next-generation systems.

We suggest that many conventional approaches in recommender systems and social link prediction, when applied to PSM, could have detrimental long-term implications for organizations' and individuals' performance. Conventional mechanisms, such as optimizing for similarity and triadic closure, involve risks of strengthening the human biases of homophily and echo chambering. We call for diversity-enhancing and contextually sensitive designs of future PSM systems, tapping on multi-dimensional analytics of not only the potential matched actors but also the intended type of collaboration and organizational context. Further, we call for a paradigmatic shift from automated recommender systems toward decision-support systems that are based on meaningful user-system cooperation.

All in all, analysis of this application area underlines the importance and urgency of rethinking some paradigmatic algorithmic approaches. In such a complex and interdisciplinary area as PSM, reconsidering the conventions is not only beneficial; it is a necessity. 

## References

- Adomavicius, G. and Tuzhilin, A. Context-aware recommender systems. *Recommender Systems Handbook*. F. Ricci, L. Rokach, and B. Shapira, eds. Springer, 2011, Boston, MA, 217–253.
- Aggarwal, I. and Woolley, A.W. Do you see what I see? The effect of members' cognitive styles on team processes and errors in task execution. *Organizational Behavior and Human Decision Processes* 122, 1 (2013), 92–99; <https://doi.org/10.1016/j.obhdp.2013.04.003>.
- Alkhatib, A., Bernstein, M.S. and Levi, M. Examining crowd work and gig work through the historical lens of piecework. In *Proceedings of the 2017 CHI Conf. Human Factors in Computing Systems*, 4599–4616. ACM Press, New York, NY; <http://doi.org/10.1145/3025453.3025974>.
- Aral, S. and Van Alstyne, M. The diversity-bandwidth trade-off. *American J. Sociology* 117, 1 (2011), 90–171; <http://doi.org/10.1086/661238>.
- Barabási, A.-L. and Albert, R. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512; <http://doi.org/10.1126/SCIENCE.286.5439.509>.
- Bendoly, E. Fit, bias and enacted sensemaking in data visualization: Frameworks for continuous development in operations and supply chain management analytics. *J. Business Logistics* 37, 1 (2016), 6–17.
- Burt, R.S. Structural holes and good ideas. *American J. Sociology* 110, 2 (2004), 349–399; <http://doi.org/10.1086/421787>.
- Chen, J. and Abouzied, A. One LED is enough: Catalyzing face-to-face interactions at conferences with a gentle nudge. In *Proceedings of the 19th ACM Conf. Computer-Supported Cooperative Work & Social Computing*, 2016, 172–183.
- Davenport, T.H. Analytics 3.0. *Harvard Business Review* (Dec. 2013).
- Easley, D. and Kleinberg, J. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, 2010.
- Faliagka, E., Ramantas, K., Tsakalidis, A. and Tzimas, G. Application of machine learning algorithms to an online recruitment system. In *Proceedings of the 7th Intern. Conf. Internet and Web Applications and Services*, 2012.
- Fogg, B. J. Persuasive technology: Using computers to change what we think and do. *Ubiquity* (Dec. 2002).
- Gonzaga, G.C., Campos B., Bradbury T. Similarity, convergence, and relationship satisfaction in dating and married couples. *J. Pers Soc Psychol.* 93, 1 (July 2007), 34–48; <https://www.ncbi.nlm.nih.gov/pubmed/17605587>.
- Granovetter, M. The strength of weak ties. *American J. Sociology* 78, 6 (May 1973), 1360–1380.
- Guy, I. Social recommender systems. *Recommender Systems Handbook*. F. Ricci, L. Rokach, and B. Shapira, eds. Springer, 2011, 511–543; [https://doi.org/10.1007/978-1-4899-7637-6\\_15](https://doi.org/10.1007/978-1-4899-7637-6_15).
- Helberger, N., Karppinen, K. and D'Acunato, L. Exposure diversity as a design principle for recommender systems. *Information, Communication & Society* 21 (2018), 191–207.
- Huhtamäki, J., Russell, M.G., Rubens, N. and Still, K. Ostinato: The exploration-automation cycle of user-centric, process-automated data-driven visual network analytics. *Transparency in Social Media*, Springer, 2015, 197–222; <http://bit.ly/2lGmvyM>.
- Hupa, K.R., Wierzbicki, A. and Datta, A. Interdisciplinary matchmaking: Choosing collaborators by skill, acquaintance and trust. *Comput. Commun. Netw.* (2010), 319–347.
- Kahneman, D. and Tversky, A. On the psychology of prediction. *Psychological Review* 80, 4 (1973), 237–251; <http://dx.doi.org/10.1037/h0034747>.
- Koren Y. and Bell R. Advances in collaborative filtering. *Recommender Systems Handbook*. F. Ricci, L. Rokach, and B. Shapira, eds. Springer, 2011, Boston, MA.
- Kossinets, G. and Watts, D.J. Origins of homophily in an evolving social network. *American J. Sociology* 115, 2 (2009), 405–450; <http://doi.org/10.1086/599247>.
- Kotkov, D., Wang, S. and Veijalainen, J. A survey of serendipity in recommender systems. *Knowledge-Based Systems* 111 (2016), 180–192.
- Kunaver, M. and Požrl, T. Diversity in recommender systems—A survey. *Knowledge-Based Systems* 123 (2017), 154–162; <https://doi.org/10.1016/j.knosys.2017.02.009>.
- Li, Z., Fang, X. and Sheng, O.R.L. A survey of link recommendation for social networks. *ACM Trans. Management Information Systems* 9, 1 (2017), 1–26; <http://doi.org/10.1145/3131782>.
- McCay-Peet, L. and Toms, E.G. The process of serendipity in knowledge work. In *Proceedings of the 3rd Symp. Information Interaction in Context*. ACM, New York, NY, 2010, 377–382.
- McNee, S.M., Riedl, J. and Konstan, J.A. Being accurate is not enough. In *Proceedings of CHI 2006 Extended Abstracts on Human Factors in Computing Systems*. ACM Press, New York, NY; <http://doi.org/10.1145/1125451.1125659>.
- Mitchell, R. and Nicholas, S. Knowledge Creation in Groups: The Value of Cognitive Diversity, Transactive Memory, and Open-mindedness Norms. *The Electronic J. Knowledge Management* 4, 1 (2006), 67–74; <https://www.ejkm.com>.
- Obstfeld, D. Social networks, the tertius iungens orientation, and involvement in innovation. *Administrative Science Quarterly* 50, 1 (2005), 100–130; <http://asa.sagepub.com/content/50/1/100.short>.
- O'Neill, C. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown, 2016.
- Pham, M.C., Kovachev, D., Cao, Y., Mbogos, G.M. and Klamra, R. Enhancing academic event participation with context-aware and social recommendations. In *Proceedings of the 2012 IEEE/ACM Intern. Conf. Advances in Social Networks Analysis and Mining*, 464–471.
- Power, D.J. Understanding data-driven decision support systems. *Information Systems Management* 25, 2 (2008), 149–154. <http://doi.org/10.1080/10580530801941124>.
- Rodan, S. and Galunic, C. More than network structure: how knowledge heterogeneity influences managerial performance and innovativeness. *Strategic Management J.* 25, 6 (June 2004). John Wiley & Sons, Ltd. 541–562.
- Shilton, Katie Values and Ethics in Human-Computer Interaction. *Foundations and Trends in Human-Computer Interaction* 12, 2 (2018), 107–171.
- Simon, H.A. *Models of Man*. New York, Wiley & Sons, 1957.
- Tanghe, J., Wisse, B. and van der Flier, H. The formation of group affect and team effectiveness: The moderating role of identification. *Br. J. Manag.* 21, 2 (July 2010), 340–358.
- Terveen, L. and McDonald, D.W. Social matching: A framework and research agenda. *ACM Trans. Comput.-Hum. Interact.* 12, 3 (Sept. 2005), 401–434; <https://doi.org/10.1145/1096737.1096740>.
- Tsai, C.-H. and Brusilovsky, P. Beyond the ranked list: User-driven exploration and diversification of social recommendation. In *Proceedings of the 2018 Conference on Human Information Interaction and Retrieval*.
- Verbert, K., Parra, D. and Brusilovsky, P. Agents vs. users: Visual recommendation of research talks with multiple dimension of relevance. *ACM Trans. Interactive Intelligent Systems* 6, 2 (2016), 11:1–11:42; <http://doi.org/10.1145/2946794>.
- Wallach, H. Computational social science ≠ computer science + social data. *Comm. ACM* 61, 3 (Mar. 2018), 42–44; <http://doi.org/10.1145/3132698>.
- Wang, G.A., Jiao, J., Abrahams, A.S., Fan, W. and Zhang, Z. ExpertRank: A topic-aware expert finding algorithm for online knowledge communities. *Decision Support Systems* 54, 3 (2013), 1442–1451; <http://doi.org/10.1016/j.dss.2012.12.020>.
- Wu, W., Chen, L. and Zhao, Y. Personalizing recommendation diversity based on user personality. *User Modeling and User-Adapted Interaction* (July 2018); <https://doi.org/10.1007/s11257-018-9205-x>.
- Zhou, S., Valentine, M. and Bernstein, M.S. In search of the dream team: Temporally constrained multi-armed bandits for identifying effective team structures. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY; <https://doi.org/10.1145/3173574.3173682>.

**Thomas Olsson** (thomas.olsson@tuni.fi) is an associate professor at Tampere University, Finland.

**Jukka Huhtamäki** (jukka.huhtamaki@tuni.fi) is a postdoctoral researcher at Tampere University, Finland.

**Hannu Kärkkäinen** (hannu.karkkainen@tuni.fi) is a professor of information knowledge and management at Tampere University, Finland.

## Reviewing software testing techniques for finding security vulnerabilities.

BY PATRICE GODEFROID

# Fuzzing: Hack, Art, and Science

*FUZZING*, OR FUZZ TESTING, is the process of finding security vulnerabilities in input-parsing code by repeatedly testing the parser with modified, or fuzzed, inputs.<sup>35</sup> Since the early 2000s, fuzzing has become a mainstream practice in assessing software security. Thousands of security vulnerabilities have been found while fuzzing all kinds of software applications for processing documents, images, sounds, videos, network packets, Web pages, among others. These applications must deal with untrusted inputs

encoded in complex data formats. For example, the Microsoft Windows operating system supports over 360 file formats and includes millions of lines of code just to handle all of these.

Most of the code to process such files and packets evolved over the last 20+ years. It is large, complex, and written in C/C++ for performance reasons. If an attacker could trigger a buffer-overflow bug in one of these applications, s/he could corrupt the memory of the application and possibly hijack its execution to run malicious code (elevation-of-privilege attack), or steal internal information (information-disclosure attack), or simply crash the application (denial-of-service attack).<sup>9</sup> Such attacks might be launched by tricking the victim into opening a single malicious document, image, or Web page. If you are reading this article on an electronic device, you are using a PDF and JPEG parser in order to see Figure 1.

Buffer-overflows are examples of security vulnerabilities: they are programming errors, or bugs, and typically triggered only in specific hard-to-find corner cases. In contrast, an exploit is a piece of code which triggers a security vulnerability and then takes advantage of it for malicious purposes. When exploitable, a security vulnerability is like an unintended backdoor in a software application that lets an attacker enter the victim's device.

There are approximately three main ways to detect security vulnerabilities in software.

*Static program analyzers* are tools that automatically inspect code and

### » key insights

- Fuzzing means automatic test generation and execution with the goal of finding security vulnerabilities.
- Over the last two decades, fuzzing has become a mainstay in software security. Thousands of security vulnerabilities in all kinds of software have been found using fuzzing.
- If you develop software that may process untrusted inputs and have never used fuzzing, you probably should.





flag unexpected code patterns. These tools are a good first line of defense against security vulnerabilities: they are fast and can flag many shallow bugs. Unfortunately, they are also prone to report false alarms and they do not catch every bug. Indeed, static analysis tools are typically unsound and incomplete in practice in order to be fast and automatic.

*Manual code inspection* consists in peer-reviewing code before releasing it. It is part of most software-development processes and can detect serious bugs. Penetration testing, or pen testing for short, is a form of manual code inspection where security experts review code (as well as design and architecture) with a specific focus on

security. Pen testing is flexible, applicable to any software, easy to start (not much tooling required), and can reveal design flaws and coding errors that are beyond the reach of automated tools. But pen testing is labor-intensive, expensive, and does not scale well since (good) pen testers are specialized and in high demand.

*Fuzzing* is the third main approach for hunting software security vulnerabilities. Fuzzing repeatedly executes an application with all kinds of input variants with the goal of finding security bugs, like buffer-overflows or crashes. Fuzzing requires test automation, that is, the ability to execute tests automatically. It also requires each test to run fast (typically in a few seconds at most)

and the application state to be reset after each iteration. Fuzzing is therefore more difficult to set up when testing complex distributed applications, like cloud or server applications running on multiple machines. In practice, fuzzing is usually most effective when applied to standalone applications with large complex data parsers. For each bug found, fuzzing provides one or several concrete inputs that can be used to reproduce and examine the bug. Compared to static analysis, fuzzing does not generate false alarms, but it is more computationally expensive (running for days or weeks) and it can also miss bugs.

Over the last two decades, fuzzing has been shown to be remarkably ef-

**Figure 1. How secure is your JPEG parser?**

fective in finding security vulnerabilities, often missed by static program analysis and manual code inspection. In fact, fuzzing is so effective that it is now becoming standard in commercial software development processes. For instance, the Microsoft Security Development Lifecycle<sup>21</sup> requires fuzzing at every untrusted interface of every product. To satisfy this requirement, much expertise and tools have been developed since around 2000.

This article presents an overview of these techniques, starting with simple techniques used in the early fuzzing days, and then progressively moving on to more sophisticated techniques. I also discuss the strengths and limitations of each technique.

Note this article is not an overview of the broader areas of automatic test generation, search-based software testing, program verification, or other applications of fuzzing techniques beyond security testing. When it was first introduced,<sup>8</sup> the term *fuzz testing* simply meant feeding random inputs to applications, without a specific focus on security. However, today, fuzzing is commonly used as a shorthand for security testing because the vast majority of its applications is for finding security vulnerabilities. Indeed, fuzzed inputs are often improbable or rather harmless unless they can be triggered and controlled by an attacker who can exploit them to deliberately break into a system and cause significant damage.

### Blackbox Fuzzing

The first and simplest form of fuzzing is *blackbox random fuzzing*, which randomly mutates well-formed application inputs, and then tests the application with these modified inputs.<sup>8</sup>

Figure 2 shows a simple program for blackbox random fuzzing. The program takes as input a well-formed input seed (line 1). It then chooses a random number of bytes that will be fuzzed in that input (line 2). That number `numWrites` varies from 1 to the length of the seed input divided by 1,000. This arbitrary 1,000 value is optional, but it prevents fuzzing too many bytes in the original seed. Next, the loop of lines 4–8 repeatedly selects a random location `loc` in the input (line 5) and a new random byte value (line 6) that is then written at that location (line 7), until the selected number `numWrites` of bytes have been fuzzed. The program then executes the application under test with that `newInput` (line 9), and reports an error if a bug is detected (line 10). In practice, the application is being run under the monitoring of a runtime checking tool (like Purify, Valgrind, AppVerifier or AddressSanitizer) in order to increase the chances of finding non-crashing security vulnerabilities (like buffer overflows).

The program of Figure 2 can be repeatedly executed to generate as many new fuzzed inputs as the user wants. Despite its simplicity, this fuzzing strategy can already be effective in finding security vulnerabilities in ap-

plications, especially if they have never been fuzzed before and if they process binary input formats. Indeed, binary formats, like the JPEG image format used for Figure 1, typically use raw byte values to encode key input properties, like image sizes, dimensions, and input-file data pointers; fuzzing these key byte values (whose locations vary from image to image) in otherwise well-formed inputs may already reveal buffer-overflow bugs due to incomplete input validation.

In practice, the effectiveness of blackbox random fuzzing crucially depends on a diverse set of well-formed seed inputs to start the fuzzing process. Indeed, well-formed seed inputs will exercise more code more quickly in the application to be fuzzed, covering various options and encodings supported by the input format. In contrast, fuzzing without well-formed seed inputs will very likely generate pure garbage, which the application under test will quickly detect and discard. This is why the program of Figure 2 defines the constant 1,000 in line 2 as its fuzzing density: if every byte in a seed input was fuzzed, the new input generated would be completely garbled and random; but if at most one byte every 1,000 bytes is fuzzed on average, fuzzing adds only limited noise to the original seed input, and testing with this slightly corrupted new input is more likely to exercise more error-handling code in more diverse parts of the application under test, hence increasing the chances of finding bugs.

### Grammar-Based Fuzzing

Blackbox random fuzzing provides a simple fuzzing baseline, but its effectiveness is limited: the probability of generating new interesting inputs is low.<sup>35</sup> This is especially true when fuzzing applications with structured input formats, like XML or JSON dialects: randomly fuzzing inputs in these formats is likely to break key structural properties of the input, which the application will quickly detect in a first lexical analysis and then discard, hence exercising little of the application code.

*Grammar-based fuzzing* is a powerful alternative for fuzzing complex formats. With this approach, the user

provides an input grammar specifying the input format of the application under test. Often, the user also specifies what input parts are to be fuzzed and how. From such an input grammar, a grammar-based fuzzer then generates many new inputs, each satisfying the constraints encoded by the grammar. Examples of grammar-based fuzzers are Peach,<sup>29</sup> SPIKE,<sup>32</sup> and Sulley,<sup>34</sup> among others.<sup>35</sup>

Figure 3 shows a code fragment representing input constraints which a grammar-based fuzzer like SPIKE uses to generate new inputs. The input grammar is represented here directly using code, which can be interpreted, for example, in Python. In line 2, the user specifies with a call to `s_string` that a constant string with the fixed value `POST /api/blog/ HTTP/1.2` should be generated first. Then another constant string `Content-Length:`  should be appended (line 3). The call to `s_blocksize_string` in line 4 adds a string of length 2 (second argument) with the size of the block named `blockA` (first argument) as value. A call to `s_block_start` defines the start of a block (line 5), while `s_block_end` denotes the end of a block (line 9), whose name is specified in the argument. In line 7, the user specifies with a call to `s_string_variable` that a fuzzed string is to be appended at this location; this string can be the constant `XXX` specified in the call or any other string value taken out of a user-defined dictionary of other values (not shown here). Tools like SPIKE support several ways of defining such fuzzing dictionaries as well as custom fuzzing rules (for example, for numeric values). By executing the code shown in Figure 3, SPIKE might generate this string sequence (shown here on 2 lines):

```
POST /api/blog/ HTTP/1.2
Content-Length:10{body:XXX}
```

This approach is very general: the user can specify how to generate input strings using nearly arbitrary code, including function recursion to generate hierarchies of well-balanced delimiters, like `{` and `}` in the example here, and strings of various sizes.

Grammar-based fuzzing is very powerful: the user expertise is used to focus and guide fuzzing toward specific input corner cases of interest, which

would never be covered with blackbox random fuzzing in practice. Sophisticated grammar-based fuzzers exist for finding security vulnerabilities in Web browsers,<sup>19</sup> which must take as untrusted inputs Web pages including complex HTML documents and JavaScript code, as well as for finding complex bugs in compilers.<sup>38</sup> Grammar-based fuzzing also works well for network-protocol fuzzing where sequences of structured messages need to be fed to the application under test in order to get good code coverage and find bugs.<sup>31</sup>

Work on grammar-based test input generation can be traced back to the 1970s.<sup>17</sup> Test generation from a grammar is usually either done using random traversals of the production rules of a grammar,<sup>26</sup> or is exhaustive and covers all its production rules.<sup>24</sup> Imperative generation<sup>6</sup> is a related approach in which a custom-made program generates the inputs (in effect, the program encodes the grammar), as shown in Figure 3.

Grammar-based fuzzing is also related to model-based testing.<sup>36</sup> Given an abstract representation of a program—called a model—model-based testing consists in generating tests by analyzing the model in order to check the conformance of the program with respect to the model. Test generation algorithms used in model-based test-

ing often try to generate a minimum number of tests covering, say, every state and transition of a finite-state machine model in order to generate test suites that are as small as possible. Similar algorithms can be used to cover all production rules of a grammar without exhaustively enumerating all possible combinations.

How to automatically learn input grammars from input samples for fuzzing purposes is another recent line of research. For instance, context-free grammars can be learned from input examples using custom generalization steps,<sup>1</sup> or using a dynamic taint analysis of the program under test in order to determine how the program processes its inputs.<sup>20</sup> Statistical machine-learning techniques based on neural networks can also be used to learn probabilistic input grammars.<sup>16</sup> While promising, the use of machine learning for grammar-based fuzzing is still preliminary and not widely used today.

In summary, grammar-based fuzzing is a powerful approach to fuzzing that leverages the user's expertise and creativity. Unfortunately, grammar-based fuzzing is only as good as the input grammar being used, and writing input grammars by hand is laborious, time consuming, and error-prone. Because the process of writing gram-

**Figure 2. Sample blackbox fuzzing code.**

```
1 RandomFuzzing(input seed) {
2   int numWrites = random(len(seed)/1000)+1;
3   input newInput = seed;
4   for (int i=1; i<=numWrites; i++) {
5     int loc = random(len(seed));
6     byte value = (byte)random(255);
7     newInput[loc] = value;
8   }
9   result = ExecuteAppWith(newInput);
10  if (result == crash) print("bug found!");
11 }
```

**Figure 3. Sample SPIKE fuzzing code.**

```
1 ...
2 s_string("POST /api/blog/ HTTP/1.2 ");
3 s_string("Content-Length: ");
4 s_blocksize_string("blockA", 2);
5 s_block_start("blockA");
6 s_string("{body:");
7 s_string_variable("XXX");
8 s_string("}");
9 s_block_end("blockA");
10 ...
```

mars is so open-ended and there are so many possibilities for fuzzing rules (what and how to fuzz), when to stop editing a grammar further is another practical issue.

### Whitebox Fuzzing

Blackbox random fuzzing is practically limited, and grammar-based fuzzing is labor intensive. Moreover, when can one safely stop fuzzing? Ideally, further testing is not required when the program is formally verified, that is, when it is mathematically proved not to contain any more bugs.

Cost-effective program verification has remained elusive for most software despite 40+ years of computer-science research.<sup>18,22</sup> However, significant advances in the theory and engineering of program analysis, testing, verification, model checking, and automated theorem proving have been made over the last two decades. These advances were possible in part thanks to the increasing computational power available on modern computers, where sophisticated analyses have now become more affordable. Whitebox fuzzing is one of these advances.

Starting with a well-formed input, *whitebox fuzzing*<sup>14</sup> consists of symbolically executing the program under test *dynamically*, gathering constraints on inputs from conditional branches encountered along the execution. The collected constraints are then systematically negated one-by-one and solved with a constraint solver, whose solu-

tions are mapped to new inputs that exercise different program execution paths. This process is repeated using systematic search techniques that attempt to sweep through all (in practice, many) feasible execution paths of the program while checking simultaneously many properties (like buffer overflows) using a runtime checker.

For example, consider this simple program:

```
int foo (int x) { // x is an input
  int y = x + 3;
  if (y == 13) abort (); // error
  return 0;
}
```

Dynamic symbolic execution of this program with an initial concrete value 0 for the input variable  $x$  takes the else branch of the conditional statement, and generates the path constraint  $x + 3 \neq 13$ . After negating this input constraint and solving it with a constraint solver,<sup>7</sup> the solver produces a solution  $x = 10$ . Running the program with this new input causes the program to follow the then branch of the conditional statement and finds the error. Note that blackbox random fuzzing has only 1 in  $2^{32}$  chances of exercising the then branch if the input variable  $x$  has a randomly-chosen 32-bit value, and it will never find the error in practice. This intuitively explains why whitebox fuzzing usually provides higher code coverage.

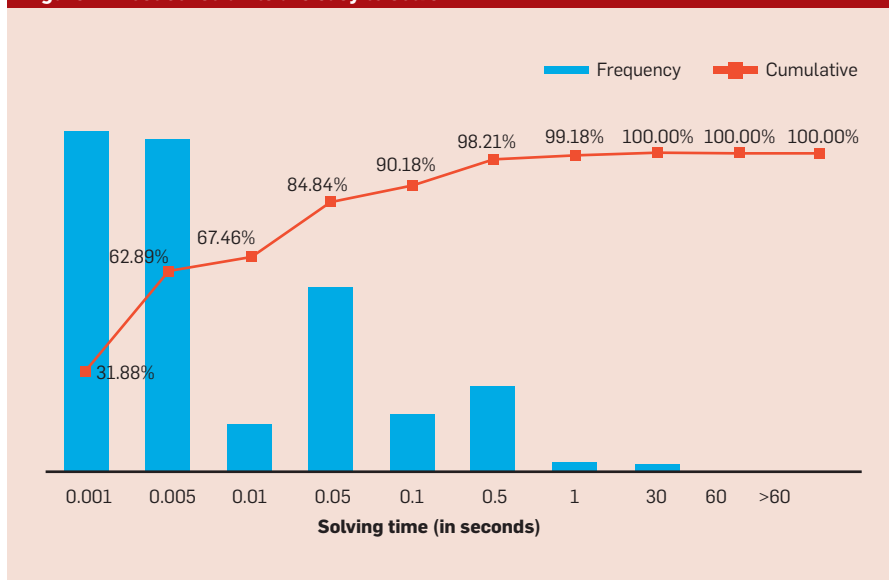
Whitebox fuzzing can generate inputs that exercise more code than other

approaches because it is more precise.<sup>11</sup> It can therefore find bugs missed by other fuzzing techniques, even without specific knowledge of the input format. Furthermore, this approach automatically discovers and tests code corner cases where programmers may fail to properly allocate memory or manipulate buffers, leading to security vulnerabilities. Note that full program statement coverage is a necessary but not sufficient condition to find all the bugs in a program.

In theory, exhaustive whitebox fuzzing provides full program path coverage, that is, program verification (for any input up to a given size). The simple program `foo` has two feasible execution paths, which can be exhaustively enumerated and explored in order to prove that that this program does not contain any buffer overflow. In practice, however, the search is typically incomplete because the number of execution paths in the program under test is huge, and because symbolic execution, constraint generation, and constraint solving may be imprecise due to complex program statements (pointer manipulations, floating-point operations, among others), calls to external operating system and library functions, and large numbers of constraints which cannot all be solved perfectly in a reasonable amount of time. Because of these limitations, whitebox fuzzing, like blackbox fuzzing, still relies on a diverse set of seed inputs to be effective in practice.

Whitebox fuzzing was first implemented in the tool SAGE,<sup>14</sup> and extends the scope of prior work on dynamic test generation,<sup>4,13</sup> also called execution-generated tests or concolic testing, from unit testing to security testing of large programs. SAGE performs dynamic symbolic execution at the x86 binary level, and implements several optimizations that are crucial for dealing with huge execution traces with hundreds of millions of machine instructions, in order to scale to large file parsers embedded in applications with millions of lines of code, like Microsoft Excel or PowerPoint. SAGE also uses search heuristics based on code coverage when exploring large state spaces: instruction coverage is measured for every test executed, and tests that discover many new instructions

Figure 4. Most constraints are easy to solve.



are symbolically executed with higher priority so that their unexplored neighborhoods are explored next. Tests and symbolic executions can be run in parallel, on multiple cores or machines. Like blackbox fuzzing, whitebox fuzzing starts with a small diverse set of seed inputs, whenever possible, in order to give its search a head start. Such seed inputs can also be generated using grammar-based fuzzing when an input grammar is available.<sup>25</sup>

Since 2008, SAGE has been running in production for over 1,000 machine-years, automatically fuzzing hundreds of applications.<sup>2</sup> This is the “largest computational usage ever for any Satisfiability-Modulo-Theories (SMT) solver” according to the authors of the Z3 SMT solver,<sup>7</sup> with around 10 billion constraints processed to date. On a sample set of 130 million constraints generated by SAGE while fuzzing 300 Windows applications, Figure 4 shows that about 99% of all constraints are solved by Z3 in one second or less.<sup>2</sup>

During all this fuzzing, SAGE found many new security vulnerabilities (buffer overflows) in hundreds of Windows parsers and Office applications, including image processors, media players, file decoders, and document parsers. Notably, SAGE found roughly one third of *all* the bugs discovered by file fuzzing during the development of Microsoft’s Windows 7,<sup>15</sup> saving millions of dollars by avoiding expensive security patches for nearly a billion PCs worldwide. Because SAGE was typically run last, these bugs were missed by everything else, including static program analysis and blackbox fuzzing.

Today, whitebox fuzzing has been adopted in many other tools. For instance, the top finalists of the DARPA Cyber Grand Challenge,<sup>37</sup> a competition for automatic security vulnerability detection, exploitation and repair, all included some form of whitebox fuzzing with symbolic execution and constraint solving in their solution. Other influential tools in this space include the open-source tools KLEE,<sup>3</sup> S2E,<sup>5</sup> and Symbolic PathFinder.<sup>28</sup>

### Other Approaches

Blackbox random fuzzing, grammar-based fuzzing and whitebox fuzzing are the three main approaches to fuzz-



**Fuzzing is commonly used as a shorthand for security testing because the vast majority of its applications is for finding security vulnerabilities.**



ing in use today. These approaches can also be combined in various ways.

*Greybox fuzzing* extends blackbox fuzzing with whitebox fuzzing techniques. It approximates whitebox fuzzing by eliminating some of its components with the goal of reducing engineering cost and complexity while retaining some of its intelligence. AFL<sup>40</sup> is a popular open source fuzzer which extends random fuzzing with code-coverage-based search heuristics as used in SAGE, but without any symbolic execution, constraint generation or solving. Despite (or because) of its simplicity, AFL was shown to find many bugs missed by pure blackbox random fuzzing. AFL is related to work on search-based software testing<sup>27</sup> where various search techniques and heuristics (such as genetic algorithms or simulated annealing) are implemented and evaluated for various testing scenarios. Another form of greybox fuzzing is *taint-based fuzzing*,<sup>10</sup> where an input-taint analysis is performed to identify which input bytes influence the application’s control flow, and these bytes are then randomly fuzzed, hence approximating symbolic execution with taint analysis, and approximating constraint generation and solving with random testing.

*Hybrid fuzzing*<sup>33,39</sup> combines blackbox (or greybox) fuzzing techniques with whitebox fuzzing. The goal is to explore trade-offs to determine when and where simpler techniques are sufficient to obtain good code coverage, and use more complex techniques, like symbolic execution and constraint solving, only when the simpler techniques are stuck. Obviously, many trade-offs and heuristics are possible, but reproducible statistically-significant results are hard to get.<sup>23</sup> Grammar-based fuzzing can also be combined with whitebox fuzzing.<sup>12,25</sup>

*Portfolio approaches* run multiple fuzzers in parallel and collect their results, hence combining their complementary strengths. Project Springfield<sup>30</sup> is the first commercial cloud fuzzing service (renamed Microsoft Security Risk Detection in 2017), and uses a portfolio approach. Customers who subscribe to this service can submit fuzzing jobs targeting their own software. Fuzzing jobs are processed by

creating many virtual machines in the cloud and by running different fuzzing tools and configurations on each of these machines. Fuzzing results (bugs) are continually collected by the service and post-processed for analysis, triage and prioritization, with final results available directly to customers on a secured website.

## Conclusion

Is fuzzing a hack, an art, or a science? It is a bit of all three. Blackbox fuzzing is a simple *hack* but can be remarkably effective in finding bugs in applications that have never been fuzzed. Grammar-based fuzzing extends it to an *art*<sup>a</sup> form by allowing the user's creativity and expertise to guide fuzzing. Whitebox fuzzing leverages advances in computer *science* research on program verification, and explores how and when fuzzing can be mathematically “sound and complete” in a proof-theoretic sense.

The effectiveness of these three main fuzzing techniques depends on the type of application being fuzzed. For binary input formats (like JPEG or PNG), fully-automatic blackbox and whitebox fuzzing techniques work well, provided a diverse set of seed inputs is available. For complex structured non-binary formats (like JavaScript or C), the effectiveness of blackbox and whitebox fuzzing is unfortunately limited, and grammar-based fuzzing with manually-written grammars are usually the most effective approach. For specific classes of structured input formats like XML or JSON dialects, domain-specific fuzzers for XML or JSON can also be used: these fuzzers parse the high-level tree structure of an input and include custom fuzzing rules (like reordering child nodes, increasing their number, inverting parent-child relationships, and so on) that will challenge the application logic while still generating syntactically correct XML or JSON data. Of course, it is worth emphasizing that no fuzzing technique is guaranteed to find all bugs in practice.

What applications should be fuzzed also depends on a number of parameters. In principle, any application that

may process untrusted data should be fuzzed. If the application runs in a critical environment, it should definitely be fuzzed. If the application is written in low-level code like C or C++, the danger is even higher, since security vulnerabilities are then typically easier to exploit. If the application is written in higher-level managed code like Java or C#, fuzzing might reveal unhandled exceptions which may or may not be security critical depending on the context (service-side code is usually more critical).

Despite significant progress in the art and science of fuzzing over the last two decades, important challenges remain open. How to engineer exhaustive symbolic testing (that is, a form of verification) in a cost-effective manner is still an open problem for large applications. How to automate the generation of input grammars for complex formats, perhaps using machine learning, is another challenge. Finally, how to effectively fuzz large distributed applications like entire cloud services is yet another open challenge. **□**

## References

- Bastani, O., Sharma, R., Aiken, A. and Liang, P. Synthesizing program input grammars. In *Proceedings of the 38<sup>th</sup> ACM SIGPLAN Conf. Programming Language Design and Implementation*, 2017, 95–110.
- Bounimova, E., Godefroid, P., and Molnar, D. Billions and billions of constraints: Whitebox fuzz testing in production. In *Proceedings of 35<sup>th</sup> Intern. Conf. Software Engineering*, (San Francisco, May 2013), 122–131.
- Cadar, C., Dunbar, D., and Engler, D. KLEE: Unassisted and automatic generation of high-coverage tests for complex systems programs. In *Proceedings of OSDI'08* (Dec 2008).
- Cadar, C. and Engler, D. Execution generated test cases: How to make systems code crash itself. In *Proceedings of 12<sup>th</sup> Intern. SPIN Workshop on Model Checking of Software 3639* (San Francisco, CA, Aug. 2005) Lecture Notes in Computer Science, Springer-Verlag.
- Chipounov, V., Kuznetsov, V. and Candea, G. S2E: A platform for in-vivo multi-path analysis of software systems. In *Proceedings of ASPLOS'2011*.
- Claessen, K. and Hughes, J. QuickCheck: A lightweight tool for random testing of Haskell programs. In *Proceedings of ICFP'2000*.
- de Moura, L. and Bjorner, N. Z3: An Efficient SMT Solver. In *Proceedings of 14<sup>th</sup> Intern. Conf. Tools and Algorithms for the Construction and Analysis of Systems 4963* (Budapest, April 2008), 337–340. Lecture Notes in Computer Science, Springer-Verlag.
- Forrester, J.E. and Miller, B.P. An empirical study of the robustness of Windows NT applications using random testing. In *Proceedings of the 4<sup>th</sup> USENIX Windows System Symp.*, Seattle, (Aug. 2000).
- Gallagher, T., Jeffries, B., and Landauer, L. *Hunting Security Bugs*. Microsoft Press, 2006.
- Ganesh, V., Leek, T., and Rinard, M. Taint-based directed whitebox fuzzing. In *Proceedings of ICSE'2009*.
- Godefroid, P. Higher-order test generation. In *Proceedings of ACM SIGPLAN 2011 Conf. Programming Language, Design and Implementation* (San Jose, June 2011), 258–269.
- Godefroid, P., Kiezun, A., and Levin, M.Y. Grammar-based whitebox fuzzing. In *Proceedings of ACM SIGPLAN 2008 Conf. Programming Language Design and Implementation*, (Tucson, AZ, USA, June 2008), 206–215.
- Godefroid, P., Klarlund, N., and Sen, K. DART: Directed automated random testing. In *Proceedings of ACM SIGPLAN 2005 Conf. Programming Language Design and Implementation* (Chicago, IL, June 2005), 213–223.
- Godefroid, P., Levin, M.Y., and Molnar, D. Automated whitebox fuzz testing. In *Proceedings of Network and Distributed Systems Security* (San Diego, Feb. 2008), 151–166.
- Godefroid, P., Levin, M.Y., and Molnar, D. SAGE: Whitebox fuzzing for security testing. *Commun. ACM* 55, 3 (Mar. 2012), 40–44.
- Godefroid, P., Peleg, H., and Singh, R. Learn&Fuzz: Machine Learning for Input Fuzzing. In *Proceedings of the 32<sup>nd</sup> IEEE/ACM Intern. Conf. Automated Software Engineering*, Nov. 2017.
- Hanford, K.V. Automatic generation of test cases. *IBM Systems J.* 9, 4 (1970).
- Hoare, C.A.R. An axiomatic approach to computer programming. *Commun. ACM* 12, 10 (1969), 576–580.
- Holler, C., Herzig, K., and Zeller, A. Fuzzing with code fragments. In *Proceedings of the 21st USENIX Security Symp.*, 2012.
- Hörschele, M. and Zeller, A. Mining input grammars with AUTOGRAM. In *Proceedings of ICSE-C'2017*, 31–34.
- Howard, M. and Lipner, S. *The Security Development Lifecycle*. Microsoft Press, 2006.
- King, J.C. Symbolic execution and program Ttesting. *J. ACM* 19, 7 (1976), 385–394.
- Klees, G.T., Ruef, A., Cooper, B., Wei, S., and Hicks, M. Evaluating fuzz testing. In *Proceedings of the ACM Conf. Computer and Communications Security*, 2018.
- Lämmel, R. and Schulte, W. Controllable combinatorial coverage in grammar-based testing. In *Proceedings of TestCom*, 2006.
- Majumdar, R. and Xu, R. Directed test generation using symbolic grammars. In *Proceedings of ASE*, 2007.
- Maurer, P.M. Generating test data with enhanced context-free grammars. *IEEE Software* 7, 4 (1990).
- McMinn, P. Search-based software test data generation: A survey. *Software Testing, Verification and Reliability* 14, 2 (2004).
- Pasareanu, C. S., Visser, W., Bushnell, D., Geldenhuys, J., Mehlitz, P., and Rungta, N. Symbolic pathFinder: Integrating symbolic execution with model checking for Java bytecode analysis. *Automated Software Engineering*, 2013, 20:391–425.
- Peach Fuzzer; <http://www.peachfuzzer.com/>.
- Project Springfield; <https://www.microsoft.com/springfield/>, 2015.
- Protos; <http://www.ee.oulu.fi/research/ouspg/protos/>.
- SPIKE Fuzzer; <http://resources.infosecinstitute.com/fuzzer-automation-with-spike/>.
- Stephens, N. et al. Driller: Augmenting fuzzing through selective symbolic execution. In *Proceedings of Network and Distributed Systems Security*, 2016.
- Sulley; <https://github.com/OpenRCE/sulley>.
- Sutton, M., Greene, A., and Amini, P. Fuzzing: Brute Force Vulnerability Discovery. Addison-Wesley, 2007.
- Utting, M., Pretschner, A., and Legeard, B. A Taxonomy of model-based testing approaches. *Intl. J. Software Testing, Verification and Reliability* 22, 5 (2012).
- Walker, M. et al. DARPA Cyber Grand Challenge, 2016; <http://archive.darpa.mil/cybergrandchallenge/>.
- Yang, X., Chen, Y., Eide, E., and Regehr, J. Finding and understanding bugs in C compilers. In *Proceedings of PLDI'2011*.
- Yun, I., Lee, S., Xu, M., Jang, Y., and Kim, T. Qsym: A practical concolic execution engine tailored for hybrid fuzzing. In *Proceedings of the 27<sup>th</sup> USENIX Security Symp.*, 2018.
- Zalewski, M. AFL (American Fuzzy Lop), 2015; <http://lcamtuf.coredump.cx/afl/>

Patrice Godefroid (pg@microsoft.com) is a partner researcher at Microsoft Research, Redmond, WA, USA.

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



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

# DEBS 2020

14th ACM INTERNATIONAL CONFERENCE ON DISTRIBUTED AND EVENT-BASED SYSTEMS

13th - 17th July 2020

École de technologie supérieure  
Montreal, Quebec, Canada

Over the past decade, the ACM International Conference on Distributed and Event-based Systems (DEBS) has become the premier venue for cutting-edge research in the field of event processing and distributed computing, and the integration of distributed and event-based systems in relevant domains such as Big Data, AI/ML, IoT, and Blockchain. The objectives of the ACM International Conference on Distributed and Event-Based Systems (DEBS) are to provide a forum dedicated to the dissemination of original research, the discussion of practical insights, and the reporting of experiences relevant to distributed systems and event-based computing. The conference aims at providing a forum for academia and industry to exchange ideas through its tutorials, research papers, and the Grand Challenge. This year's Grand Challenge focuses on Non-Intrusive Load Monitoring (NILM). The goal of the challenge is to detect when appliances contributing to an aggregated stream of voltage and current readings from a smart meter are switched on or off. The prize money for the winning submission of the Challenge is 1000 USD.

## Important Dates

### Research and industry track

Abstract submission for research track - March 13th, 2020

Paper submission - March 20th, 2020

Author notification - May 3rd, 2020

**Tutorial submission** - March 3rd, 2020

**Poster, demo & doctoral symposium submission** - May 10th, 2020

### Grand challenge

Solution Submission - April 7th, 2020

Short paper submission - April 14th, 2020

Notification of acceptance - May 1st, 2020

**Camera ready for all tracks** - May 28th, 2020

<https://2020.debs.org>

[https://twitter.com/ACM\\_DEBS](https://twitter.com/ACM_DEBS)

<https://www.facebook.com/ACM.DEBS>

### General Chair

Kaiwen Zhang, ÉTS  
Khuzaima Daudjee, University of Waterloo  
Bettina Kemme, McGill University

### Program Chair

Mohammad Sadoghi, UC Davis  
Lukasz Golab, University of Waterloo

### Proceedings Chair

Julien Gascon-Samson, ÉTS

### Industry Chair

Vinod Muthusamy, IBM  
Jan Rellermeier, TU Delft

### Grand Challenge Chair

Vincenzo Gulisano, Chalmers University  
Ruben Mayer, TU Munich  
Daniel Jorde, TU Munich  
Dimitris Palyvos-Giannas, Chalmers University  
Hannaneh Najdataei, Chalmers University

### Tutorials/Workshop Chair

Tilmann Rabl, Hasso Plattner Institute  
Leonardo Querzoni, Sapienza University of Rome

### Doctoral Symposium Chair

Roman Vitenberg, University of Oslo  
David Eyers, University of Otago

### Demos & Posters Chair

Vinay Setty, University of Stavanger  
Aris Leivadreas, ÉTS

### Web Chair

Sardar Basiri, ÉTS

### Publicity Chair

Sukanya Bhowmik, University of Stuttgart  
Christoph Doblender, TU Munich



# ACM Transactions on Social Computing (TSC)

Open for  
Submissions

Seeks to publish work that covers the full spectrum of social computing including theoretical, empirical, systems, and design research contributions



*ACM Transactions on Social Computing (TSC)* seeks to publish work that covers the full spectrum of social computing including theoretical, empirical, systems, and design research contributions. The editorial perspective is that social computing is fundamentally about computing systems and techniques in which users interact, directly or indirectly, with what they believe to be other users or other users' contributions. TSC welcomes research employing a wide range of methods to advance the tools, techniques, understanding, and practice of social computing, including: theoretical, algorithmic, empirical, experimental, qualitative, quantitative, ethnographic, design, and engineering research. Social computing will continue to be shaped by foundational algorithmic, econometric, psychological, sociological, and social science research and these broad-based perspectives will continue to have a profound influence on how social computing systems are designed, built and how they grow.

For more information and to submit your work, please visit:

TSC particularly solicits research that designs, implements or studies systems that mediate social interactions among users, or that develops or studies theory or techniques for application in those systems. Examples of such social computing systems include, but are not limited to: instant messaging, blogs, wikis, social networks, social tagging, social recommenders, collaborative editors and shared repositories.

[tsc.acm.org](http://tsc.acm.org)



Association for  
Computing Machinery



# research highlights

---

P. 80

**Technical  
Perspective**  
**Lighting the Way  
to Visual Privacy**

By Marco Gruteser

P. 81

**Automating Visual Privacy  
Protection Using a Smart LED**

By Shilin Zhu, Chi Zhang, and Xinyu Zhang

# Technical Perspective

## Lighting the Way to Visual Privacy

By Marco Gruteser

AS CAMERAS PERVADE OUR lives, a defining question is how to design technologies that can protect proprietary information and respect the privacy preferences of individuals. Mobile devices have already significantly reduced the effort to capture and share images. Moreover, technology trends are moving toward continuous visual sensing where even mobile cameras remain always active to monitor the environment and user context. What technical approaches help balance the convenience and usefulness of such applications against the preferences of subjects whose likeness or property is captured in such recordings?

Such questions are frequently addressed through fair information principles such as disclosure and consent that should be incorporated into the system design. Such consent solutions have been difficult to apply to camera images since a subject may be unaware of the images being captured. Short of hiding from sight or using masks, few systems offer provisions for opting out of having images of one's likeness or property recorded. Some ad hoc opt-out solutions allow subjects to have photographs showing their likeness or property removed from sharing services. This places responsibility on the subject to identify where recordings of them or their property are shared, a process that is becoming increasingly onerous as the number of services using and sharing imagery multiply.

The following paper presents a new technique to address this issue at the point of image capture—it stops most digital cameras from recording a useful image while the scene remains visible to human eyes. How is this possible? The authors introduce a smart high-intensity LED light that flickers at a high rate so that it creates a striping effect that severely degrades the image. It exploits the rolling shutter image sensors used in a vast major-


**Short of hiding from sight or using masks, few systems offer provisions for opting out of having images of one's likeness or property recorded.**

ity of digital cameras, where each row in the pixel array is exposed at a slightly different time while the camera exposure control remains adjusted to the average brightness over the frame period. When the illuminating light flickers at a rate higher than the frame capture rate, this results in sets of rows alternatingly over- and under-exposed and hence the striping effect. The human eye, however, will not perceive it, provided the flicker frequency of the LED light is high enough, since it essentially acts as a low-pass filter. The degradation therefore only materializes in captured images with rolling shutter cameras.

The paper also introduces two variants of this technology. First, it shows how specific cameras can be allowed to take uncompromised images while the degradation remains in effect for other unauthorized cameras. In a corporate setting, for example, this could allow taking photos with company devices while making it more difficult to capture proprietary information with personal or visitor devices. Second, it shows how this technology can be used to encode a watermark in the captured image. This can offer a weaker form of protection, particularly in areas where lighting conditions do not support

degradation of the image. Such a watermark could make it easier to identify unauthorized images that are shared or published.

How might this technology evolve? Further research and development could aim to support a wider range of lighting conditions and camera to subject/light source distances than those studied to date. Research could also tackle power consumption to enable deployment in the form of battery-powered tokens or even in wearable applications. Additionally, one might study how a level of protection can also be realized for global shutter cameras, which the current technology cannot address, and further understand the level of protection offered against a broader range of image processing and computational photography techniques. More generally, one may ask whether related ideas apply to other types of sensors that can capture sensitive information such as microphones.

Overall, this work represents a significant step toward privacy-enhancing technologies that can protect against the capture of information and suggests that there is a richer design space waiting to be explored. 

**Marco Gruteser** is a research scientist at Google AI, chair of ACM SIGMOBILE, and the Peter D. Cherasia Faculty Scholar and professor (on leave) in the Department of Electrical and Computing Engineering at Rutgers University, New Brunswick, NJ, USA.

# Automating Visual Privacy Protection Using a Smart LED

By Shilin Zhu, Chi Zhang, and Xinyu Zhang

## Abstract

**The ubiquity of mobile camera devices has been triggering an outcry of privacy concerns, whereas existing privacy protection solutions still rely on the cooperation of the photographer or camera hardware, which can hardly be enforced in practice. In this paper, we introduce LiShield, which automatically protects a physical scene against photographing, by illuminating it with smart LEDs flickering in specialized waveforms. We use a model-driven approach to optimize the waveform design, so as to ensure protection against the (uncontrollable) cameras and potential image-processing-based attacks. We have also designed mechanisms to unblock authorized cameras and enable graceful degradation under strong ambient light interference. Our prototype implementation and experiments show that LiShield can effectively destroy unauthorized capturing while maintaining robustness against potential attacks.**

## 1. INTRODUCTION

Cameras are now pervasive on consumer mobile devices, such as smartphones, tablets, drones, smart glasses, first-person recorders, etc. The ubiquity of these cameras, paired with pervasive wireless access, is creating a new wave of visual sensing applications, for example, autonomous photograph, quantified-self (life-logging), photo-sharing social networks, physical analytics in retail stores,<sup>12</sup> and augmented reality applications that navigate users across unknown environment.<sup>19</sup> Zooming into the photo-sharing application alone, statistics report that 350 million photos/videos are uploaded to Facebook every day, majority of which are from mobile users.<sup>15</sup> Many of these applications automatically upload batches of images/videos online, with a simple one-time permission from the user. Although these technologies bring significant convenience to individuals, they also trigger an outcry of privacy concerns.

Privacy is ultimately a subjective matter and often varies with context. Yet, many of the privacy-sensitive scenes occur in indoor environment and are bound to specific locations. For example, recent user studies<sup>2</sup> showed that people's acceptability of being recorded by augmented reality glasses has a strong correlation with location. User studies of life-logging cameras<sup>6</sup> also indicate that 70.2% of the cases when the user disables capturing are associated with specific locations. In numerous real-world scenarios, cameras are forbidden, for example, concerts, theaters, museums, trade shows, hospitals, dressing rooms and exam rooms, manufacturing plants, etc. However, *visual privacy protection in such passive physical spaces still heavily relies on rudimentary approaches like warning signs and human monitors, and there is no way to automatically enforce the requirements. In personal visual sensing applications like life-logging, even if a user*

were to disable the camera in private space, malware could perform remote reconnaissance and target visual theft by hijacking the victim's camera.<sup>16</sup>

In this paper, we propose LiShield, a system that deters photographing of sensitive indoor physical space and automatically enforces location-bound visual privacy protection. LiShield protects the physical scenes against undesired recording without requiring user intervention and without disrupting the human visual perception. Our key idea is to illuminate the environment using smart LEDs, which are intensity-modulated following specialized waveforms. We design the waveform in such a way that its modulation pattern is imperceptible by human eyes but can interfere with the image sensors on mobile camera devices.

**Adversary model and protection goals.** LiShield aims to prevent ad-hoc capturing from benign camera-phone holders. The physical space under protection can be static or dynamic. In either case, we assume that one or multiple LiShield-enabled smart LEDs can cover the whole area, while providing illumination similar to normal office lighting without human-perceptible flickering. Although conventional lighting and sunlight may co-exist with LiShield's smart LEDs, covering the entire target scene with LiShield will ensure the strongest protection.

Now consider an unauthorized user (attacker) who wants to take pictures or videos within the protected space, with cameras and lenses embedded in smartphones, but with no professional equipment such as global shutter cameras, filters, or tripods. The attacker has full control over the camera parameters (e.g., exposure time, capturing time, and white-balancing) and can run any postprocessing on the captured images. Nonetheless, with LiShield's protection, the image frames are corrupted, so that major fraction of each frame is either blank or overexposed, whereas colors are distorted (Section 2), which deters image viewing/sharing.

In addition, LiShield should possess the following capabilities for practical usage scenarios: (i) allowing an authorized camera, which shares secret configuration information with the LED, to recover the image or video frames it captures. (ii) when strong ambient light interferes with the smart LED, LiShield cannot ensure full protection, but it can still emit structured light which embeds invisible "barcode" into the physical environment. The embedded information can convey a "no distribution" message, allowing online servers (e.g., from Facebook and Instagram) to block and prevent the image from being distributed.

The original version of this paper appeared in *ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2017.

### How does LiShield disrupt camera image capturing?

Cameras and human eyes perceive scenes in fundamentally different ways. Human eyes process continuous vision by accumulating light signals, whereas cameras slice and sample the scene at discrete intervals. Consequently, human eyes are not sensitive to high frequency flickers beyond around 80 Hz either in brightness or chromaticity,<sup>7</sup> whereas cameras can easily pick up flicker above a few kHz.<sup>20</sup> Equally importantly, human eyes perceive brightness in a nonlinear fashion, which gives them huge dynamic range, whereas cameras easily suffer from overexposure and underexposure when signals with disparate intensities mix in the same scene.

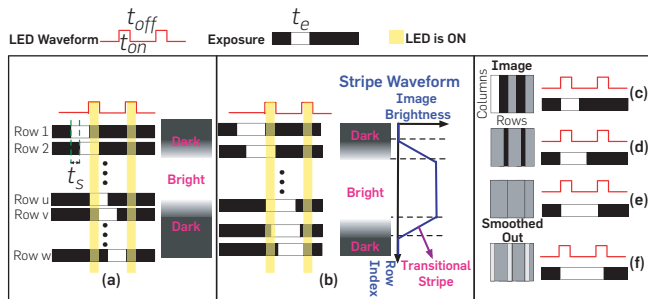
Unlike professional or industrial cameras which may have global shutters that mimic human eyes to some degree, nearly all consumer digital cameras, pinhole cameras, and smartphones use the rolling shutter sampling mechanism,<sup>8</sup> which is the main contributor to their high-frequency sensitivity. When capturing an image frame, a *rolling shutter camera exposes each row sequentially*.

LiShield harnesses the disparity between cameras and eyes to disrupt the camera imaging without affecting human vision. It modulates a smart LED to generate high-frequency flickering patterns. The reflection intensity (or brightness) of target scene also flickers following the same pattern as the LED's illumination, albeit at reduced intensity due to reflection loss. LiShield uses the On-Off Keying (OOK) as the basic modulation waveform (Figure 1), which does not require complicated analog front-ends and is widely supported by smart LEDs. Due to rolling-shutter sampling, the rows of pixels that are fully exposed in the ON period will be bright, and rows in the OFF period become dark, thus causing striped patterns on the captured image (Figure 1(a, b)). Partially exposed rows experience moderate brightness. Meanwhile, human eyes can only perceive the smooth averaged intensity as long as the OOK frequency goes beyond 80 Hz.<sup>7,21</sup>

In addition, LiShield can turn different numbers of LED bulb/chip on to generate different intensities and control the RGB channels of the LEDs to vary the color. In Section 2, we will show how such flickering corrupts the spatial patterns captured by a camera.

**Summary of results.** We have implemented LiShield based on a customized smart LED, which allows reconfiguration of intensity modulation waveforms on each color channel.

**Figure 1. (a) and (b) Bright, dark and transitional stripes and their width changing with exposure time; (c)–(f) stripe pattern of image changes under different exposure times.**



Our experiments on real world scenes demonstrate that LiShield can corrupt the camera capturing to an illegible level, in terms of the image brightness, structure, and color. The impact is resilient against possible attacks, such as multi-frame combining and denoising. On the other hand, it enables authorized cameras to recover the image perfectly, as if no modulation is present. Even under strong sunlight/flashlight interferences, LiShield can still sneak barcode into the physical scenes, which can be decoded with around 95% accuracy.

## 2. DISRUPTING CAMERA CAPTURING USING SMART LIGHTING

### 2.1. Maximizing image quality degradation

LiShield aims to *minimize the image capturing quality by optimizing the LED waveform, characterized by modulation frequency, intensity, and duty cycle*. To this end, we derive a model that can predict the image quality as a function of the LiShield's waveform and attacker's camera parameters. For simplicity, we start with monochrome LED with a single color channel that illuminates the space homogeneously. We denote  $P$  as the reference image taken under a nonflickering LED and  $Q$  as the one taken under LiShield's LED with the same average brightness. We assume each image has  $m$  rows and  $n$  columns, and the light energy received by each pixel is denoted by  $P(i, j)$  and  $Q(i, j)$ , respectively. Our model focuses on two widely adopted image quality metrics: *PSNR*, which quantifies the disruption on individual pixel intensity levels, and *SSIM*,<sup>18</sup> which measures the structural distortion to the image (i.e., deformation effects such as stretching, banding, and twisting). In general, the minimum PSNR and SSIM corresponding to acceptable viewing quality are in the range of 25–30 and 0.8–0.9, respectively.<sup>1</sup>

**Decomposing the image.** To compute the image quality, we need to model the intensity and width of each stripe caused by LiShield. As illustrated in Figure 1, we use  $t_{on}$ ,  $t_{off}$ , and  $I_p$  to denote the on/off duration and peak intensity of the flickering light source, and  $t_e$  and  $t_s$  are the exposure time and sampling interval of the rolling shutter camera. For convenience, denote the period of the light source as  $t_l = t_{on} + t_{off}$  and duty cycle as  $D_c = t_{on}/t_l$ . For pixel  $j$  in row  $i$  which starts exposure at time  $t_i$ , its light accumulation would be:

$$Q(i, j) = \alpha_{i,j} \int_{t_i}^{t_i+t_e} \pi_l(\tau) d\tau \quad (1)$$

where  $\alpha_{i,j}$  is the aggregated path-loss for pixel  $(i, j)$ , such as attenuation and reflection on the photographed object, and  $\pi_l(\tau)$  represents the illumination waveform of the LED:

$$\pi_l(\tau) = \begin{cases} I_p, & 0 < \tau \bmod t_l \leq t_{on} \\ 0, & t_{on} < \tau \bmod t_l \leq t_l \end{cases} \quad (2)$$

When the camera's exposure time is equal to or shorter than the LED's OFF period ( $t_e \leq t_{off}$ ), the image will contain rows that are completely dark (Figure 1(c)). On the other hand, when  $t_e > t_l$ , one row-exposure period of the camera will overlap multiple ON periods of the LED, accumulating higher intensity (Figure 1(f)). The special case happens when  $t_e = t_l$ , where the integration of LED waveform and exposure has fixed value, which eventually smooths out dark stripes (Figure 1(e)). Without loss of generality, assume that the

exposure starts right at the beginning of the ON period. Let  $N = \lfloor t_e / t_p \rfloor$ , which is the number of whole flicker cycles covered by exposure time, and  $t_{\text{rem}} = (t_e \bmod t_p)$ , which is the remaining duration after multiple whole cycles, and the light accumulation of the brightest rows  $Q_B$  is:

$$Q_B(i, j) = \begin{cases} \alpha_{i,j} I_p (N t_{\text{on}} + t_{\text{rem}}), & 0 < t_{\text{rem}} \leq t_{\text{on}} \\ \alpha_{i,j} I_p (N + 1) t_{\text{on}}, & t_{\text{on}} < t_{\text{rem}} \leq t_l \end{cases} \quad (3)$$

Since the brightest rows appear when the exposure captures most ON periods possible (e.g., row 2 to row  $u$  in Figure 1(a)) and rolling shutter effect converts temporal variation into pixels with sampling interval  $t_s$ , the width of  $Q_B$  is:

$$W_B = \lfloor t_{\text{rem}} - t_{\text{on}} \rfloor / t_s \quad (4)$$

Likewise, when the exposure captures least ON periods possible (e.g., from row  $v$  to row  $w$  in Figure 1(a)), we get the darkest rows with light accumulation  $Q_D$ :

$$Q_D(i, j) = \begin{cases} \alpha_{i,j} I_p N t_{\text{on}}, & 0 < t_{\text{rem}} \leq t_{\text{off}} \\ \alpha_{i,j} I_p (N t_{\text{on}} + t_{\text{rem}} - t_{\text{off}}), & t_{\text{off}} < t_{\text{rem}} \leq t_l \end{cases} \quad (5)$$

and the width of  $Q_D$  is:

$$W_D = \lfloor t_{\text{rem}} - t_{\text{off}} \rfloor / t_s \quad (6)$$

We refer to a collection of consecutive brightest rows as “bright stripe” and consecutive dark rows as “dark stripe,” as shown in Figure 1(b). In addition, there exist intermediate rows containing linear intensity transition between dark and bright, referred to as “transitional stripe.”

Meanwhile, if the LED were not flickering and provided the same average brightness, the pixel intensity would be:

$$P(i, j) = \alpha_{i,j} I_p \cdot D_c \cdot t_e \quad (7)$$

Since  $D_c \cdot t_e$  remains constant within each frame, *the image captured under LiShield is equivalent to the original image multiplied by a piecewise function* (cf. Equations (3) and (5)).

Other common camera parameters (i.e., ISO, white balance, and resolution) do not affect the structure of the stripe pattern. By default, we assume that the attacker sets the ISO to its minimum (typically 100) to maximally suppress noise.

**Optimizing the LED waveform.** Since the stripe pattern follows a piecewise function, a closed form expression of PSNR and SSIM becomes hard to analyze. We thus use numerical simulation to evaluate the impact of LiShield, based on the above model. We generate the piecewise function with  $Q_B(i, j)$ ,  $W_B$ ,  $Q_D(i, j)$ , and  $W_D$  and multiply it on reference images to obtain the disrupted image  $Q$  just like the process inside real cameras. We assume  $t_s = 1/75,000$  s, which matches the capability of a Nexus 5 camera. The quality metrics are calculated between the reference image  $P$  and LiShield-corrupted image  $Q$  with same average intensity. We make pixel intensity range infinite, which allows quantifying quality loss caused by overexposure.

By default, we use OOK waveform with frequency  $f = 100$  Hz, peak intensity  $I_p = 10$  kLx, and duty cycle  $D_c = 0.5$ . We vary one parameter while keeping others to the defaults. Note that the

typical light intensity is  $\sim 700$  Lx in office environments and  $\sim 100,000$  Lx outdoor in sunny days. Our numerical results lead to the following design choices for LiShield.

**(i) A single frequency cannot ensure robust protection.** For a given waveform frequency  $f$ , there exist several exposure time settings that lead to high-quality images. This is because when  $t_e \approx N t_p$ , the stripes become smoothed out (Figure 1(e)). Although the waveform parameters are unknown to the attacker, a determined attacker may launch a brute-force search for the  $t_e$  that satisfies this condition, thus circumventing the protection. To counteract such attackers, *LiShield includes a countermeasure called frequency randomization*, which we discuss in “Frequency scrambling” section.

**(ii) LiShield must leverage overexposure to prevent attackers from using long exposures.** The image quality increases with exposure time  $t_e$ , until overexposure happens, because longer exposure leads to more waveform cycles being included as a constant base in the brightness of the stripes (larger  $N$  in Equations (3) and (5)), making the contrast of stripes  $Q_B/Q_D$  lower and weakening the quality degradation. *LiShield should leverage overexposure to limit attacker’s exposure time.* On the other hand, when exposure goes beyond a threshold, the image always suffers from over-exposure. If not, the image is always corrupted due to the dominance of stripes under LiShield’s frequency randomization mechanism (“Frequency scrambling” section). With power efficiency and eye health in mind, *LiShield sets  $I_p$  to 20 kLx by default.* Optimal parameters may vary slightly across different scenes (e.g., different reflectivity) but can be easily obtained by running the aforementioned simulation.

## 2.2. Circumventing potential attacks

Based on the foregoing analysis, we identify the following potential holes that can be exploited by attackers to overcome the striping effect. *(i) Manual exposure attack.* If an attacker can configure  $t_e$  to satisfy  $t_e \approx N t_p$ , it can guarantee that every row receives almost the same illumination, thus eliminating the stripes during a capture (Figure 1(e)). In practice,  $t_l$  is unknown to the attacker, but it can try to capture images with different  $t_e$ , until seeing a version without obvious stripes. *(ii) Multiframe attack.* When the scene is static, an attacker may also combine multiple frames (taking a video and playback) to mitigate the stripes with statistical clues, for example, by averaging or combining rows with maximum intensities from multiple frames. Note that the attacker must keep the camera highly stable, otherwise even pixel-level shift will cause severe deformation when combining multiple frames. *(iii) Post-processing attack.* Common postprocessing techniques (e.g., denoising and de-banding) might be used to repair the corrupted images.

In what follows, we introduce countermeasures to the first two attacks. In Section 6.4, we will verify that LiShield’s distortion does not fit empirical noise or banding models, so the common post-processing schemes become ineffective.

**Frequency scrambling.** To thwart the manual exposure attack, we design a *frequency scrambling* mechanism, which packs multiple waveforms with different frequencies within each image frame duration. Since the camera exposure time

$t_e$  is always fixed within each frame, no single  $t_e$  can circumvent all the frequency components.

However, we cannot choose and switch the flickering frequencies in an arbitrary manner, for three reasons. (i) Multiple frequency values that share a common divisor can satisfy  $t_e = Nt_l$  under the same  $t_e$  (recall  $N$  can be an arbitrary integer). We need to ensure the common divisor is small enough (i.e., least common multiplier of  $t_l$  large enough), so that overexposure occurs even for the smallest  $N$ . (ii) Frequencies should be kept low to maximize image corruption, as evident in Optimizing the LED waveform section, since camera’s analog gain decreases at high frequencies.<sup>20</sup> (iii) Switching between different frequencies may create an additional level of modulation, which will spread the spectrum and generate unexpected low frequency components that become perceivable by eyes.

To explore the design space under these constraints, suppose we switch among  $M$  frequencies  $f_1, f_2, \dots, f_M$  (in ascending order) at a switching rate  $f_B$ . The whole pattern thus repeats itself at rate  $f_p = f_B/M$ . To pack at least  $M$  different frequencies in an image frame, we need  $f_B > (M - 1)f_r$  or, preferably,  $f_B > Mf_r$ , where  $f_r$  is the frame rate, typically around 30 Hz (fps). To maximize image corruption, we choose the smallest value for  $f_1$  (i.e.,  $f_1 = f_B$ ) and empirically set  $f_n = f_B + (n - 1)\Delta f$ ,  $n = 2, 3, \dots, M$ , where  $\Delta f$  is frequency increment, set to  $\Delta f \neq f_B$  to lower the common divisor frequency.

The frequency scrambling can be considered as an M-FSK modulation, thus creating side lobes around each scrambling frequency, spacing  $f_p$  apart (Interested readers can refer to the full version of this work for the theoretical underpinning.<sup>22</sup>). These side lobes might appear at low-frequency region and become perceptible by eyes. To ensure no side lobe exists below the perceivable threshold  $f_{th} \approx 80$  Hz, we need a small  $M$  and large  $f_B$  and hence higher flickering frequency components  $f_n$ . Yet, increasing the flickering frequencies may weaken LiShield’s protection. To find the optimal  $\Delta f$  and showcase the effectiveness of the frequency scrambling, we repeat the numerical simulation (Section 2.1) to evaluate the attacker’s maximum image quality. Based on the simulation, we set  $\Delta f = 50$  Hz to maximize image disruption. The optimal  $\Delta f$  for other peak intensity settings can be obtained following a similar procedure.

**Illumination intensity randomization.** If attackers repetitively capture a static scene for a sufficiently long duration, they may eventually find at least one clean version for each row across all frames, thus *recovering* the image. LiShield can increase the number of frames needed for image recovery, so that the attack becomes infeasible unless the camera can stay perfectly still over a long period of time, during which the attackers may have already been discovered by the owners of the physical space. LiShield achieves the goal by employing illumination intensity randomization, where it randomly switches the magnitude of each ON period across multiple predefined levels, which extends the attacker’s search space.

### 3. SCENE RECOVERY WITH AUTHORIZED CAMERAS

To allow authorized users to capture the scene while maintaining protection against unauthorized attackers, we need

to impose additional constraints on the LED waveform. LiShield’s solution leverages a secure side channel (e.g., WiFi<sup>4</sup>) between authorized users and the smart LED, which conveys secret information such as frame timing and waveform parameters.

A naive solution is to stop flickering when authorized users are recording. However, since attackers may be co-located with the authorized users, this enables them to capture one or more frames that have part of the clean scene, which compromises privacy and security. Instead, we design special waveforms for the LED to counteract such cases.

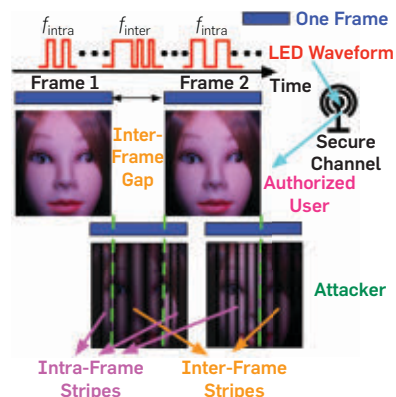
#### 3.1. Authorized video recording

To authorize a camera to capture a dynamic scene, each individual frame within the video must be recoverable. To achieve this, the authorized camera needs to convey its exposure time setting  $t_e^u$  to the smart LED via the secure side channel and synchronize its clock (for controlling capturing time) with the smart LED’s clock (for controlling the waveform), so the smart LED can send recoverable waveforms precisely during the capture of the authorized camera. State-of-the-art time synchronization mechanisms (e.g.,<sup>4</sup>) can already achieve  $\mu s$  of accuracy, sufficient to synchronize the LiShield smart LED with camera at a resolution that is finer than the rolling shutter period (typically tens of  $\mu s$ ).

Recall that the camera can evade the striping effects if  $t_e = Nt_l$ . So to authorize the user with exposure  $t_e^u$ , LiShield simply needs to set its flickering frequency  $f_a^u = 1/t_l = N/t_e^u$  ( $N = 1, 2, \dots$ ) and maintain its peak intensity within each frame. In addition, the  $t_e^u$  and corresponding flickering frequency  $f_a^u$  can be varied on a frame by frame basis, making it impossible for an attacker to resolve the correct exposure time by trial-and-error (Section 2.2).

Meanwhile, when the authorized camera is not recording at its maximum possible rate, there will be an interval (i.e., inter-frame gap) where the camera pauses capturing. LiShield packs random flickering frequencies  $f_{inter}$  other than  $f_{intra} = f_a$  into the interframe gap, so as to achieve the same scrambling effect as described in “Frequency scrambling” section, without compromising the authorized capturing, as shown in Figure 2.

**Figure 2. Enabling authorized users to capture dynamic scenes while corrupting unauthorized users.**



### 3.2. Static scene recovery

When the target scene is static, the authorized user may capture a few complementary frames at a specific time to recover the scene as depicted in Figure 3, where frequency and intensity randomization (Section 2.2) are employed in each frame to ensure robustness. Although it does require recording a very short video, the process is extremely short (200 ms at most) and barely noticeable to the authorized user. Meanwhile, an out-of-sync attacker will still receive corrupted images that cannot reconstruct the original scene by direct frame combination.

Suppose a static scene is to be recovered using  $L_f$  frames, referred to as *critical frames*. To prevent attackers from launching the multiframe attack, the timing of the critical frames is negotiated only between the smart LED and the authorized user through the secure side channel. These  $L_f$  frames together must contain the information of the entire scene, that is, they must be complementary, as shown in Figure 3. Meanwhile, all other frames will follow the normal flickering pattern. Since the attackers can neither identify nor predict the timing of the critical frames, the best they can do is to launch the brute-force multiframe attack, which has proven to be ineffective (“Illumination intensity randomization” section).

## 4. AUTOMATIC PHYSICAL WATERMARKING FOR PRIVACY ENFORCEMENT

High-intensity ambient light sources (e.g., sunlight, legacy lighting, and ash lights) can create strong interference to LiShield’s illumination waveform, degrading the contrast by adding a constant intensity to both the bright and dark stripes, which may weaken LiShield’s protection. In such scenarios, LiShield degrades itself to a *barcode mode*, where it embeds barcode in the physical scene to convey privacy policies. The barcode forms low-contrast stripes, which may not fully corrupt the images of the scene, but can still be detected by online photo-distributing hubs (e.g., social website servers), which automatically enforce the policies, without co-operation of the uploader or evidence visible by naked eye. LiShield forms the watermark with just a single light fixture, instead of active displays (e.g., projectors) that are required by conventional systems. The key challenge here is how should LiShield encode the information, so that it can be robustly conveyed to the policy enforcers, despite the (uncontrollable) attacker camera settings?

**Embedding.** LiShield’s barcode packs multiple frequencies in every image following “Frequency scrambling” section but aims to map the *ratios between frequencies* into

digital information. Suppose LiShield embeds two waveforms with frequencies  $F_0$  and  $F_1$ , it chooses the two frequency components such that  $F_1/F_0$  equals to a value  $R_p$  well known to the policy enforcers. In other words, the presence of  $R_p$  conveys “no distribution/sharing allowed.” Although width of stripes is affected by sampling interval  $t_s$  and exposure time  $t_e$  (Figure 1(a) and (b)), ratio of stripe widths resulted from two frequencies (which equals to  $R_p$ ) remains constant. Therefore, this encoding mechanism is robust against camera settings.

Since physical scenes usually comprise a mix of spatial frequencies, and spectral power rolls off in higher spatial frequencies, thanks to camera lenses’ limited bandwidth while temporal frequencies are unaffected, LiShield’s barcode uses frequencies that are much higher than the natural frequencies (>400 Hz) in the scene to reduce interference. It is worth noting that since the rolling-shutter sampling rate of all cameras falls in a range (30 kHz to slightly over 100 kHz<sup>20</sup>), LiShield limits its highest flickering frequency to 15 kHz, which respects the Nyquist sampling theorem, so that the barcode can eventually be recovered without any aliasing effect.

To further improve robustness, LiShield leverages redundancy. It embeds multiple pairs of frequency components to make multiple values of  $R_p$  either at different rows of the image or in different color channels, further mitigating interference caused by intrinsic spatial patterns within the scene.

**Detection.** Since the barcode contains  $M$  frequencies, that is,  $f_n = f_b + (n - 1)\Delta f$ ,  $n = 2, 3, \dots, M$  (“Frequency scrambling” section), there are  $M_R = C_M^2$  possible frequency ratio values across the image for monochrome barcode ( $M_R = C_{M \times 3}^2$  for RGB barcode).  $\Delta f$  must be set large enough to avoid confusion ( $\Delta f = 200$  Hz in experiments). The barcode decoder, running on the policy enforcer, recognizes the image as protected if there are at least  $M_b$  values that roughly match the known ratio  $R_p$ , that is, when the value falls within  $T_b$  of  $R_p$ . Suppose  $M_{att}$  is the number of  $R_p$  removed by manual exposure attack (Section 2.2), these parameters are determined by bounding the false positive rate following an empirical procedure (to be discussed in Section 6.3).

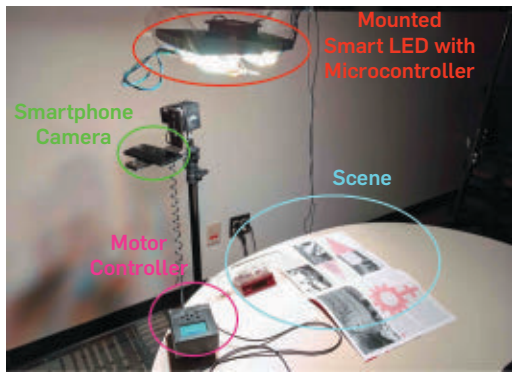
To detect the frequency ratios, LiShield averages the intensity of each row to get a one-dimension time series  $s_r$ . LiShield then runs FFT over each series to extract the  $M_p$  strongest frequencies. Finally, LiShield combines all unique frequencies extracted and computes all frequency ratios. The redundancy in barcode ensures that it can be robustly detected.

## 5. IMPLEMENTATION

**Testbed setup.** Figure 4 shows our smart LED prototype, and the target scenes containing five capture-sensitive objects (document and painting are 2-D objects and others are all 3-D objects). We mount the LED inside a diffusive plastic cover similar to conventional ceiling light covers. We use a programmable motor to hold the camera and control its distance/orientation, in order to create static or dynamic scene setup in a repeatable manner.

**Figure 3. The impact of multiframe recovery on authorized user and attacker, respectively.**



**Figure 4. Experimental setup of LiShield.**

**Smart LED modules.** Commercial-of-the-shelf (COTS) household LED bulbs rely on integrated drivers to regulate LED's current. A dimming input is usually available on these drivers for controlling the current dynamically. We build our smart bulb based on the same topology as these COTS LED bulbs. We use 19V DC laptop power supplies and NCL30160 LED drivers, which allow dimming at nearly 100 kHz with arbitrary OOK waveform. The smart bulb has built-in independent RGB/white channels for controlling color/intensity. Each channel can be controlled by a separate waveform, with four LED chips in series, at driving current of 800 mA. In total, the three channels consume approximately 25 W peak power, close to common office LED troffer fixtures. However, since LiShield's OOK waveform has a duty cycle much lower than one (Section 2), the actual perceptible brightness is significantly lower.

The dimming input signals of each channel are controlled by an STM32 microcontroller unit (MCU), which generates the OOK waveform as specified by LiShield. For flexible reconfiguration, we generate digitized waveforms in MATLAB on a laptop or Android app on a smartphone instead, which are then passed to the MCU via USB.

**Android app for normal, authorized and attacker's cameras.** Unless otherwise noted, we use Nexus 5 with stock ROM as our benchmark device. We assume that normal users use the stock camera app with default settings (such as auto exposure), whereas a malicious attacker can manually tune the camera parameters (e.g., using the Open Camera app). By default, the camera ISO is set to the lowest value (100), since it is most beneficial for attackers, as it allows longer exposure to smooth out the stripes without causing overexposure. To implement the authorization mechanism (Section 3), we develop a specialized app for the authorized smartphone, which uses Android's Camera2 API<sup>5</sup> to precisely control the exposure time, as well as communicating with the smart LED's MCU via USB. Since current Android camera APIs do not support precise frame timing, the app requests the smart LED to synchronize with the camera by altering its waveform.

**Attacker's image processing.** We have implemented the attacking algorithms in Section 2.2, which are specifically designed to combine/process the captured image, aiming to eliminate LiShield's stripe distortion. In addition, we implement classical image processing techniques, such

as denoising and debanding, which may be attempted by attackers. For denoising, we use the Haar-wavelet thresholding, non-local-means (NLmeans), and BM3D, which are among the most popular algorithms.<sup>14</sup> As for debanding, we use the Banding Denoise and Unstrip in the G'MIC plugin.

**Metrics.** Small displacement and vibration of the camera are inevitable in physical environment, which is known to affect the SSIM. Thus, we quantify the image quality degradation with the enhanced CW-SSIM,<sup>13</sup> which is insensitive under such translations. PSNR shows similar trends with SSIM. Besides, we employ the CIEDE2000<sup>9</sup> to compute the degradation of the images' color quality when the RGB LED is used.

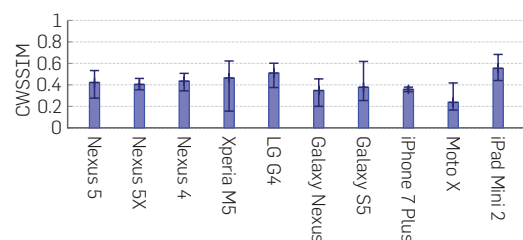
## 6. EXPERIMENTAL EVALUATION

### 6.1. Effectiveness of physical scene disruption

**Impact of scenes and device heterogeneity.** We first verify LiShield's basic protection scheme (Section 2) with five static scenes, monochrome LEDs, and OOK waveform without frequency randomization, although the attacker's camera uses auto-exposure. Without LiShield, the measured image quality stays high, with PSNR > 30 dB and CW-SSIM > 0.9 (slightly lower than simulation results due to digital noises in real cameras). LiShield degrades the image quality to 3–10 dB for PSNR and 0.25–0.6 for CW-SSIM (Figure 5). We cross-validate the impact of LiShield on 10 common mobile cameras. Although the image quality varies slightly due to different sampling rates across devices, the quality remains at an intolerably low level across devices. Thus *LiShield's protection mechanism works across typical smartphone camera models*. As a visual quality benchmark, Figure 6 plots the same scene with different qualities under flickering (Figure 7).

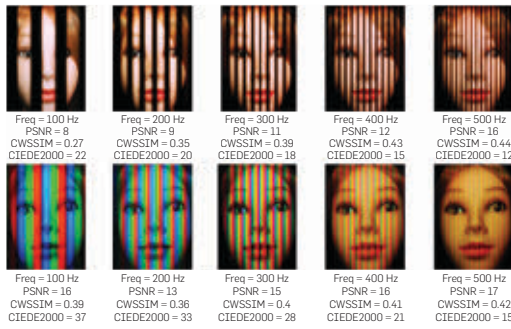
We also notice that *the quality worsens slightly as flickering frequency decreases* from 500 Hz to 100 Hz (CW-SSIM decreases by  $\approx 0.1$ ), as the image sensor has higher analog gain at lower flickering frequencies.<sup>20</sup>

**Impact of RGB color distortion.** When the RGB flickering is turned on, *the quality degradation is stronger if the RGB LED has the same average intensity with monochrome LED*. Besides, the color distortion makes an additional independent impact. The corresponding CIEDE2000 metric escalates up to 45, way beyond the *human-tolerable threshold* 6.<sup>9</sup> This implies *the scene is no longer considered acceptable by average viewers*.

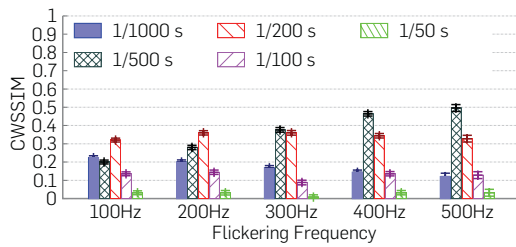
**Figure 5. Quality with autoexposure camera. Error bars show std. across OOK waveforms with different frequencies (100–500 Hz) and scenes.**



**Figure 6. Image quality levels on a benchmark image.**



**Figure 7. Quality with fix-exposure camera.**



Two *bonus effects* from our RGB LED are observed: (i) The structural distortion from the stripes disrupts the camera’s auto-focus function, often *making the captured scene extremely blur*. This is because under LiShield, contrast of bands no longer depends on focusing accuracy, which breaks the assumption of auto-focus mechanism. (ii) *The color bands also mislead the automatic white balance function across all five different scenes*, since the camera can no longer identify a clean region in the image to calibrate itself and thus hesitates.

**Impact on dynamic scenes.** To create a dynamic scene, we use the motor to rotate the smartphone, creating relative motion at three different speeds (45, 100, and 145 degrees/second). Our experiment shows the average CW-SSIM among all three speeds further decreases by 0.1, which indicates that *dynamic scene experiences worse quality under LiShield* due to motion blur. Moreover, if the exposure time is larger than 1/100 s, then overexposure and motion blurs together further reduce the quality (PSNR < 6, CW-SSIM < 0.1). Thus, *dynamic objects further decrease the adjustment range of exposure time and make manual exposure attack more ineffective*.

## 6.2. Effectiveness of user authorization

We developed an app (Section 5) that allows a user to capture critical frames on static scene protected by our RGB LED and then recover the scene following Section 3. The resulting image quality (PSNR = 25dB, CW-SSIM = 0.9, CIEDE2000 = 5) is comparable to the ideal setting when we disable LiShield’s LED modulation (Figure 8 shows example frames extracted from a recorded video). In contrast, the

attacker suffers intolerable image corruption (PSNR = 13dB, CW-SSIM = 0.56, CIEDE2000 = 34) by combining same number of randomly selected frames (“Illumination intensity randomization” section).

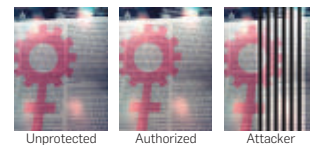
For the dynamic scene, we set  $f_{\text{intra}} = 1$  kHz and  $f_{\text{inter}} = 300$  Hz (Section 3.1). From Figure 9, we can see the authorized user has much higher quality (PSNR = 30dB, CW-SSIM = 0.98 in average) compared with attacker (PSNR = 10dB, CW-SSIM = 0.6 in average). This can be seen by resulting image frames in Figure 8, where attacker suffers from both intra-frame and inter-frame stripes. Thus *LiShield’s authorization scheme is effective in unblocking specific users while maintaining protection against attackers*.

## 6.3. Effectiveness of barcode embedding

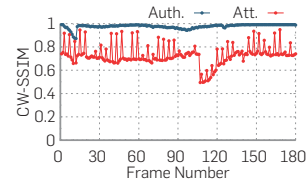
We first determine general optimal parameters for LiShield’s barcode detector in Section 4, based on the following metrics. (i) False alarm rate. We run the detector on 200 images (random real-world scenes) and measure the probability that a barcode is detected from clean image. (ii) Detection rate. We embed monochrome barcodes with different  $f_1$  from 400 Hz to 10 kHz with 200 Hz switching frequency. For each  $f_1$ , we embed three frequencies with  $\Delta f = 200$  Hz interval and capture 300 images with these barcodes over a benchmark scene to obtain detection rate. Considering the trade-off between false alarm and detection, we choose  $T_b = 0.05$  to bound the false alarm rate below 5%, while ensuring around 90% detection rate for monochrome barcode (Figure 10).

Using the above configuration, we found the detection rate for RGB barcode is close to 100% with or without manual exposure attack, while being slightly below

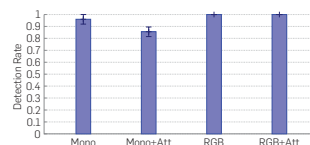
**Figure 8. Frames observed by authorized users and attackers.**



**Figure 9. Video quality with and without authorization.**



**Figure 10. Detection rate of monochrome and RGB barcode design.**



90% for monochrome barcodes if attacked. We conclude that *LiShield's barcode detector provides reliable detection, whereas RGB barcodes are more detectable and robust than monochrome ones*, thanks to extra redundancy provided by color channels.

#### 6.4. Robustness against attacks

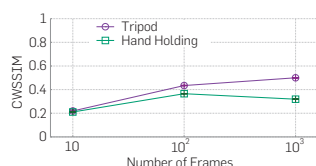
**Manual exposure attack.** One possible attack against LiShield is to manually set the exposure time  $t_e$  to smooth out the flickering patterns (Section 2.2). Our experiment shows that although the image quality first increases with  $t_e$ , it drops sharply as overexposure occurs. Therefore, *LiShield traps the attacker in either extremes by optimizing the waveform* (Section 2.1) and *thwarts any attempts through exposure time configuration*.

We also tested the effectiveness of randomization with auto-exposure (except for attacker). We set  $f_1 = f_b = 200, 300, \dots, 600$  Hz,  $\Delta f = 50$  Hz, and  $M = 2, 3, \dots, 6$  to scramble multiple frequencies. We found that the image degradation with scrambling is comparable with single frequency setup, thus *frequency randomization does not harm LiShield's protection*.

**Multiframe attack.** Figure 11 plots the recovered scene's quality under the multiframe attack. Here, we set  $t_e$  to be 1/500 s to avoid overexposure and then record a video in 30 fps. CW-SSIM remains low at 0.5 using 1000 frames, which means *the impact of stripes on structure of scene is still strong, making quality still unacceptable for professionals who spend such a great cost*. We also ask five volunteers to hold the smartphone as stable as they can on a table, and Figure 11 shows the quality is even lower, because it is impossible to completely avoid dithering with hands. Extending the recording duration increases disturbance and probability of being identified by the protected user, making it impractical for the attack to occur.

**Image recovery processing attack.** We evaluate the image quality after postprocessing with denoising or debanding (Section 5). The denoising methods fail to improve the quality significantly (CW-SSIM  $\approx 0.3$ – $0.4$ ) as the disruption pattern of LiShield does not fit most known Gaussian noise model. The deformation removal methods (i.e., debanding and unstriping) do not help too much (CW-SSIM  $\approx 0.4$ – $0.5$ ), since interpolation process cannot bring back the exact pixel values. The CIEDE2000 color metric also shows a low quality (around 35). Thus, *it is hard to fully remove LiShield's impact by simple image restoration*. More advanced computer vision techniques may provide better recovery, but even they will not recover the *exact original scene* since information is already lost at capture time.

**Figure 11. Image quality by multiframe ensemble.**



**Impact of ambient light.** We have also evaluated LiShield's performance under different types of ambient lights. We found the stripes are almost completely removed under direct sunlight due to its extremely high intensity. Flash light can increase the quality slightly thanks to its close distance to the scene, but the improvement is marginal and far from unprotected. In addition, we only found a marginal decrease of barcode detection rate in every case. Thus, we conclude that *LiShield is robust against most ambient lights*.

#### 7. RELATED WORK

Camera recording of copyright screen-displayed videos (e.g., in a movie theater) accounts for 90% of pirated online content.<sup>21</sup> Since screen refresh rate is much higher than video frame rate, Kaleido<sup>21</sup> scrambles multiple frames within the frame periods to deter recording, while preserving viewing experience by taking advantage of human eyes' flicker fusion effects. Many patented technologies addressed the same issue. In contrast, the problem of automatic protection of private and passive physical space received little attention. Certain countries dictate that smartphone cameras must make shutter sound to disclose the photo capturing actions, yet this does not enforce the compliance, cannot block the photo distribution, and cannot automatically protect against video recording.

Certain optical signaling systems can remotely ban photography in concerts, theaters, and other capturing-sensitive sites. For example, BlindSpot<sup>17</sup> adopts a computer vision approach to locate retro-reflective camera lenses and pulses a strong light beam toward the camera to cause overexposure. Such approaches fail when multiple cameras coexist with arbitrary orientations.

Conventional visual privacy-protection systems have been relying on postcapture processing. Early efforts employed techniques like region-of-interest masking, blurring, mosaicking, etc.,<sup>11</sup> or re-encoding using encrypted scrambling seeds.<sup>3</sup> There also exists a vast body of work for hiding copyright marks and other information in digital images/videos (e.g.,<sup>10</sup>). LiShield's barcode protection is inspired by these schemes, but it aims to protect physical scenes before capturing.

#### 8. CONCLUSION

Privacy protection in passive indoor environment has been an important but unsolved problem. In this paper, we propose LiShield, which uses smart-LEDs and specialized intensity waveforms to disrupt unauthorized cameras, while allowing authorized users to record high quality image and video. We implemented and evaluated LiShield under various representative indoor scenarios, which demonstrates its effectiveness and robustness. We consider LiShield as a first exploration of automatic visual privacy enforcement and expect that it can inspire more research along the same direction.

#### Acknowledgment

This project was partially supported by the NSF under Grant CNS-1506657, CNS-1518728, and CNS-1617321. 

## References

1. Barni, M. *Document and Image Compression*, CRC Press, Boca Raton, FL, 2006.
2. Denning, T., Dehlawi, Z., Kohno, T. In situ with bystanders of augmented reality glasses: Perspectives on recording and privacy-mediating technologies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*, 2014.
3. Dufaux, F., Ebrahimi, T. Scrambling for privacy protection in video surveillance systems. *IEEE Trans. Circuits Syst. Video Technol.* 18, (2008), 8.
4. Ferrari, F., Zimmerling, M., Thiele, L., Saukh, O. Efficient network flooding and time synchronization with glossy. In *Proceedings of the ACM/IEEE IPSN*, 2011.
5. Google. android.hardware.camera2.
6. Hoyle, R., Templeman, R., Armes, S., Anthony, D., Crandall, D., Kapadia, A. Privacy behaviors of lifeloggers using wearable cameras. In *Proceedings of the ACM UbiComp*, 2014.
7. Jiang, Y., Zhou, K., He, S. Human visual cortex responds to invisible chromatic flicker. *Nat. Neurosci.* 10, 5 (2007), 657–662.
8. Liang, C.K., Chang, L.W., Chen, H.H. Analysis and compensation of rolling shutter effect. *IEEE Trans. Image Processing* 17, 8 (2008), 1323–1330.
9. Luo, M.R., Cui G., Rigg, B. The development of the CIE 2000 colour-difference formula: CIEDE2000. *Color Res. Appl.* 26, 5 (2001), 340–350.
10. Naor, M., Shamir, A. Visual cryptography. In *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT)*, 1995.
11. Newton, E.M., Sweeney, L., Malin, B. Preserving Privacy by De-Identifying Face Images. *IEEE Trans. Knowl. Data Eng.* 17, 2 (2005), 232–243.
12. Rallapalli, S., Ganesan, A., Chintalapudi, K., Padmanabhan, V.N., Qiu, L. Enabling physical analytics in retail stores using smart glasses. In *Proceedings of the ACM MobiCom*, 2014.
13. Sampat, M.P., Wang, Z., Gupta, S., Bovik, A.C., Markey, M.K. Complex wavelet structural similarity: A new image similarity index. *IEEE Trans. Image Processing* 18, 11 (2009), 2385–2401.
14. Shao, L., Yan, R., Li, X., Liu, Y. From heuristic optimization to dictionary learning: A review and comprehensive comparison of image denoising algorithms. *IEEE Trans. Cybern.* 44, 7 (2014), 1001–1013.
15. Social Pilot. 125 Amazing Social Media Statistics You Should Know, 2016.
16. Templeman, R., Rahman, Z., Crandall, D., Kapadia, A. PlaceRaider: Virtual theft in physical spaces with smartphones. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2013.
17. Truong, K.N., Patel, S.N., Summet, J.W., Abowd, G.D. Preventing camera recording by designing a capture resistant environment. In: *Proceedings of ACM International Conference on Ubiquitous Computing (UbiComp)* (2005), 73–86.
18. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing* 13, 4 (2004), 600–612.
19. Winterhalter, W., Fleckenstein, F., Steder, B., Spinello, L., Burgard, W. Accurate indoor localization for rgb-d floor plans. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2015.
20. Zhang, C., Zhang, X. LiTell: Robust indoor localization using unmodified light fixtures. In *Proceedings of the ACM MobiCom*, 2016.
21. Zhang, L., Bo, C., Hou, J., Li, X.-Y., Wang, Y., Liu, K., Liu, Y. Kaleido: You can watch it but cannot record it. In *Proceedings of the ACM MobiCom*, 2015.
22. Zhu, S., Zhang, C., Zhang, X. Automating visual privacy protection using a Smart LED. In *Proceedings of the ACM MobiCom*, 2017.

Shilin Zhu, Chi Zhang, and Xinyu Zhang  
[shz338, c4zhang, xyzhang]@eng.ucsd.edu,  
University of California San Diego, CA,  
USA.

© 2020 ACM 0001-0782/20/2 \$15.00

# Computing and the National Science Foundation, 1950-2016

*Building a Foundation for Modern Computing*

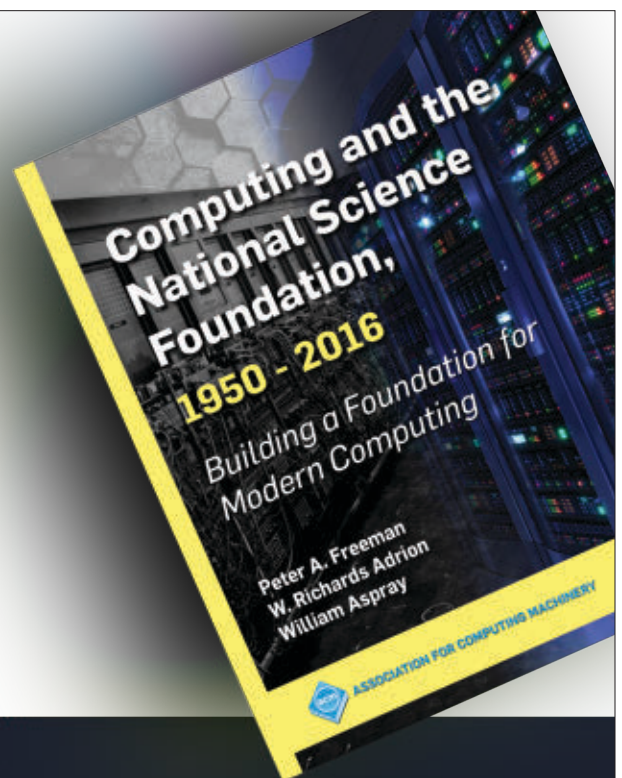
**Peter A. Freeman**  
**W. Richards Adrion**  
**William Aspray**

ISBN: 978-1-4503-7271-8

DOI: 10.1145/3335772

<http://books.acm.org>

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



**ACM BOOKS**  
Collection II

# 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

SPONSORED BY Microsoft

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](mailto:hernandez@hq.acm.org)

## The United States Military Academy (West Point)

### Tenure-Track Assistant Professor Position

The United States Military Academy (West Point) seeks applicants for a tenure track faculty position in the Department of Electrical Engineering and Computer Science.

#### Credentials and desired applicant qualities:

Applicants must be US citizens holding a Ph.D. degree in Computer Science, Computer Engineering, Electrical Engineering, Cybersecurity or related field and must be able to teach at all levels – from introductory courses to senior capstone courses in their discipline. Faculty at West Point develop cadets not only in the intellectual domain, but also as “leaders of character” and future Army officers, so the ideal candidate will have a desire to engage with cadets outside the classroom in activities such as academic clubs, character/ethics education, sports, and other mentorship opportunities. Successful candidates will show a clear vision for a research program that will position her/him for promotion to Associate Professor within 6 years; research typically involves cadets, encourages faculty collaboration, and contributes to the ever-growing reputation of the department and the Academy.

#### About West Point:

West Point, the oldest engineering school in the United States, is a four-year baccalaureate college situated just fifty miles north of New York City in the scenic Hudson River valley. West Point's mission is to educate leaders of character committed to the values of Duty, Honor and Country and devoted to a career of service to the Nation.

Each student (cadet) passes through an intense nomination and appointment process to gain admission. Our acceptance rate is 10%, with the student body consisting of approximately 4,500 cadets. The class of 2023 consists of 1,200 students of which 23% are women, 15% are African Americans, and 12% are Hispanic. West Point is a leader in producing national scholarship winners, and to date has produced 93 Rhodes Scholars, 42 Marshalls Scholars, 31 Fulbright scholars and 39 Hertz scholars. This positions us close to our aspirational schools and ahead of schools such as MIT, Cornell and Duke. Our undergraduate academic program blends the liberal arts with the math and sciences and is consistently ranked highly by U.S. News & World Report, the Princeton Review, Niche and others.

The Department of Electrical Engineering and Computer Science (EECS) is one of the most dynamic and highly regarded departments at West Point and offers ABET accredited majors in Computer Science, Information Technology, and Electrical Engineering. The Department is

in the process of developing the curriculum and associated courses necessary for an accredited Cyber Major, to complement its EE, CS and IT programs. In 2019, the department graduated 78 majors, including 9 cadets who earned national scholarships. EECS has state-of-the-art laboratory facilities and also hosts three centers of excellence: (1) the Cyber Research Center; (2) the Robotics Research Center; and (3) the Photonics Research Center. The department's ties to Army Research Agencies and a unique affiliation with the NSA make it an ideal and unique place to conduct applied research in cyber, photonics, and robotics areas, although faculty research may be within any disciplinary topic desired. EECS is a family-friendly department with values rooted in Trust, Consideration, Expertise, Teamwork, Initiative, Community, Excellence and Balance that make it an ideal place to work, grow and develop.

#### Applicant Selection Factors and Process:

Previous teaching experience is highly desirable, and an initial track record of research and pub-

lications will be integral to the selection process. Applications should highlight any experience in the following areas: cybersecurity, radio frequency electronics, artificial intelligence, and/or machine learning. Salaries are competitive.

Complete applications must arrive in electronic form not later than midnight, 15 January 2020. Applications to be considered complete must include: 1) cover letter; 2) curriculum vitae; 3) statement of research plans; 4) statement describing teaching philosophy/experience; 5) official transcripts; and 6) three letters of reference. Interviews will be conducted in late January of 2020 and a selection made by early February of 2020. Anticipated start date will be on or about the 15th of June 2020. Applications and questions may be directed to Dr. James Loy, United States Military Academy, Department of Electrical Engineering and Computer Science, MADN-EC, West Point, New York 10996, james.loy@westpoint.edu.

The United States Military Academy is an Equal Opportunity, Affirmative Action Employer. Women and Minorities are encouraged to apply.

ZJU-UIUC Institute  
Zhejiang University / University of Illinois at Urbana-Champaign Institute

### Tenure-Track Faculty Positions in Engineering in Zhejiang University

*The Zhejiang University-University of Illinois at Urbana-Champaign Institute (the ZJU-UIUC Institute) invites highly qualified candidates for multiple tenure-track faculty positions at all levels and in areas of engineering and science that match its multidisciplinary mission.*

The ZJU-UIUC Institute is an engineering college on the new Zhejiang University (ZJU) International Campus, China, about 120 km southwest of Shanghai. The ZJU-UIUC Institute conducts teaching and research in broad program themes of engineering sciences for human health, engineering sciences for flexible manufacturing, and engineering sciences at the nexus of energy, environment, and sustainable develop. Undergraduate and graduate degrees are offered in civil engineering, computer engineering, electrical engineering, and mechanical engineering. Applications are welcome from relevant engineering disciplines, computer science, and mathematics. The Institute has interests that address but are not limited to interdisciplinary topics exemplified by data science, artificial intelligence, internets of things, advanced communication, digital manufacturing, systems and networking, transportation electrification, micro-nano-electronics and photonics, Terahertz, smart power, biotechnology, nanotechnology, and atomic-scale materials, intelligent infrastructure. Classes and student activities are conducted in English.

Successful candidates will initiate and lead collaborative research and teaching, and perform academic and professional service duties associated with the ZJU-UIUC Institute. They will be leaders for teaching and research innovation, giving students a meaningful and interactive engineering education.

- Assistant professor candidates must have an earned doctorate, excellent academic credentials, strong research plans, and an outstanding ability to teach effectively. Additional industry, post-doctoral, professional service, or internship experiences are preferred.
- Mid-career candidates must be established leaders in their field; exhibit strong records of teaching, publication, and funded research; and demonstrate participation in interdisciplinary collaborations.
- Senior appointments are available for persons of international stature seeking to build substantial interdisciplinary research and teaching programs.

The search process will continue until positions are filled or through June 30, 2020. Application materials should include a cover letter with current contact information including email address, a complete curriculum vitae, statements of research and teaching goals, and the names of three or more references. Submit applications at <http://zjui.illinois.edu> or to [zjuhr@zju.edu.cn](mailto:zjuhr@zju.edu.cn). For more information, please visit <https://my.zju.illinois.edu/submit/login.asp> or contact the institute human resources office at [zjuhr@zju.edu.cn](mailto:zjuhr@zju.edu.cn) or +86 (571) 8757 2520.

ZJU and UIUC are renowned for their engineering programs, and have a long history of collaboration. The ZJU-UIUC Institute creates a unique student experience of multidisciplinary collaboration, technical leadership, teamwork, and creative excellence. Individuals with diverse backgrounds, experiences, and ideas who embrace and value diversity and inclusivity are encouraged to apply and all qualified applicants will receive consideration without regard to race, national origin, disability, age, or other personal characteristics.

**Contact Person: Chen Zhang | Email: [zjuhr@zju.edu.cn](mailto:zjuhr@zju.edu.cn)**

**Southern University of Science and Technology (SUSTech)**  
**Professor Position in Computer Science and Engineering**

The Department of Computer Science and Engineering (CSE, <http://cse.sustc.edu.cn/en/>), Southern University of Science and Technology (SUSTech) has multiple Tenure-track faculty openings at all ranks, including Professor/Associate Professor/Assistant Professor. We are looking for outstanding candidates with demonstrated research achievements and keen interest in teaching, in the following areas (but are not restricted to):

- ▶ Data Science
- ▶ Artificial Intelligence
- ▶ Computer Systems (including Networks, Cloud Computing, IoT, Software Engineering, etc.)
- ▶ Cognitive Robotics and Autonomous Systems
- ▶ Cybersecurity (including Cryptography)

Applicants should have an earned Ph.D. degree and demonstrated achievements in both research and teaching. The teaching language at SUSTech is bilingual, either English or Putonghua. It is perfectly acceptable to use English in all lectures, assignments, exams. In fact, our existing faculty members include several non-Chinese speaking professors.

As a State-level innovative city, Shenzhen has identified innovation as the key strategy for its development. It is home to some of China's most successful high-tech companies, such as

Huawei and Tencent. SUSTech considers entrepreneurship as one of the main directions of the university. Strong supports will be provided to possible new initiatives. SUSTech encourages candidates with experience in entrepreneurship to apply.

SUSTech is a pioneer in higher education reform in China. The mission of the University is to become a globally recognized research university which emphasizes academic excellence and promotes innovation, creativity and entrepreneurship. Set on five hundred acres of wooded landscape in the picturesque Nanshan (South Mountain) area, the campus offers an ideal environment for learning and research.

SUSTech is committed to increase the diversity of its faculty and has a range of family-friendly policies in place. The university offers competitive salaries and fringe benefits including medical insurance, retirement and housing subsidy, which are among the best in China. Salary and rank will commensurate with qualifications and experience. More information can be found at <http://talent.sustc.edu.cn/en>.

We provide some of the best start-up packages in the sector to our faculty members, including one PhD studentship per year, in addition to a significant amount of start-up funding (which can be used to fund additional PhD students and postdocs, research travels, and research equipments).

To apply, please provide a cover letter identifying the primary area of research, curriculum vitae, and research and teaching statements, and forward them to [cshire@sustc.edu.cn](mailto:cshire@sustc.edu.cn).



**ADVERTISING  
 IN CAREER  
 OPPORTUNITIES**

**How to Submit a Classified Line Ad: Send an e-mail to [acmm mediasales@acm.org](mailto: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](mailto: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](mailto:acmm mediasales@acm.org)**

**ACM Transactions on  
 Computing for Healthcare (HEALTH)**

**Open for  
 Submissions**

*A multidisciplinary journal for high-quality original work  
 on how computing is improving healthcare*



**ACM Transactions on Computing for Healthcare (HEALTH)** is a multidisciplinary journal for the publication of high-quality original research papers, survey papers, and challenge papers that have scientific and technological results pertaining to how computing is improving healthcare.

**For further information and to submit  
 your manuscript, visit [health.acm.org](http://health.acm.org)**



# Introducing ACM Transactions on Data Science (TDS)

A new journal from ACM publishing papers on cross disciplinary innovative research ideas, algorithms, systems, theory and applications for data science

## Now Accepting Submissions

The scope of *ACM Transactions on Data Science* (TDS) includes cross disciplinary innovative research ideas, algorithms, systems, theory and applications for data science. Papers that address challenges at every stage, from acquisition on, through data cleaning, transformation, representation, integration, indexing, modeling, analysis, visualization, and interpretation while retaining privacy, fairness, provenance, transparency, and provision of social benefit, within the context of big data, fall within the scope of the journal.

By its very nature, data science overlaps with many areas of computer science. However, the objective of the journal is to provide a forum for cross-cutting research results that contribute to data science as defined above. Papers that address core technologies without clear evidence that they propose multi/cross-disciplinary technologies and approaches designed for management and processing of large volumes of data, and for data-driven decision making will be out of scope of this journal.

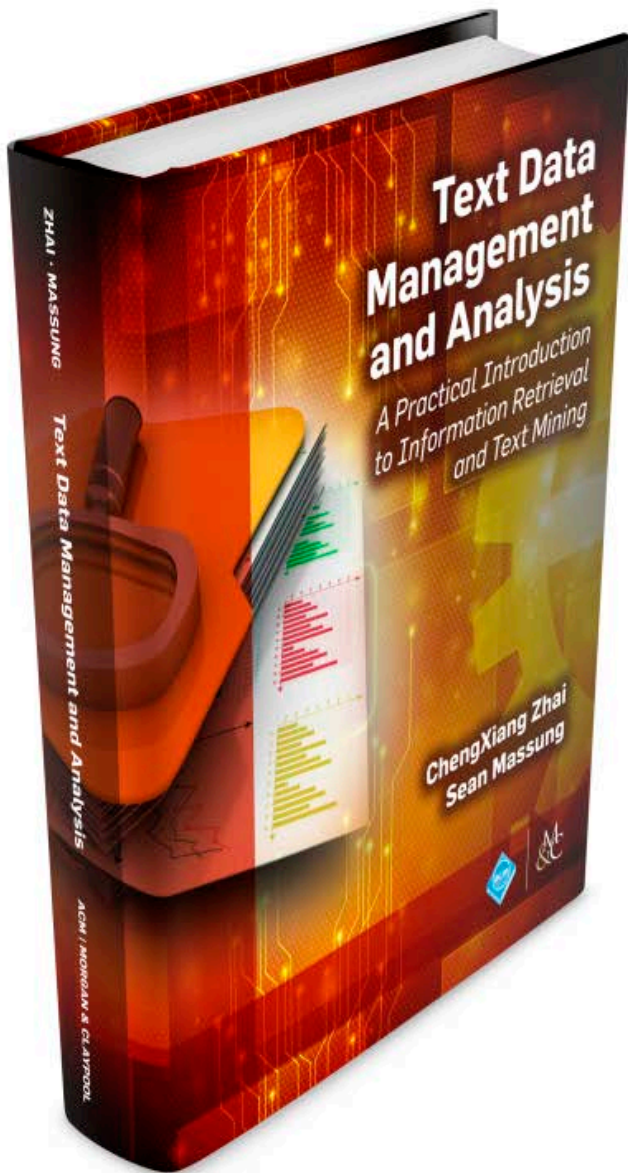


For more information and to submit your work, please visit <https://tds.acm.org>.



Association for  
Computing Machinery

Advancing Computing as a Science & Profession



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

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

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

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



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



[CONTINUED FROM P. 96] **Is there anything you would have done differently, either in terms of developing AWS services or promoting them to customers?**

Well, that's a long list. The good thing is that we have a culture at Amazon that allows you to change your mind. One of the principles we had in the earlier days was that this should all be self-service. Customers should not have to talk to a human; they should be able to do everything by themselves. On the one hand, that removed a number of obstacles for people to start using our technology, because all you needed was a credit card and an email address. But over time, we discovered that enterprises as well as startups really want to have human contact. They need account management, professional services, and architectural support to help them build their systems. It's something we didn't appreciate enough in the early days.

**What about in terms of technology?**

There's lots of things I would have done differently. For example, in the beginning, we combined identity and account. Those are two different things. Identity is a security component, and account is what you bill things to. If we had been smarter, we would have separated them, as we eventually did.

**You have talked about the importance of giving developers operational responsibilities, and the Amazon product definition process of "working backwards" is also designed to incentivize people to take the customer's view. What else have you learned about building and maintaining a product-centered culture?**

The product nature is crucial to how we operate at Amazon. Things that are now common terms—service-oriented architecture, micro-services, DevOps, DevSecOps—were pioneered at Amazon before we had words for them.

That certainly goes for the operational side of things. If you really want your engineers and your product people to be in touch with your customers, there cannot be a layer of operations people between them. Our thinking has always been to make sure our engineers are in contact with their customers so that they can make decisions on their behalf. Take, for example, the AWS data-

**"Over time, we discovered that enterprises as well as startups really want to have human contact. ... It's something we didn't appreciate enough in the early days."**

base groups. We have so many unique, purpose-built databases, and each of those groups is in contact with their most important customers. That creates a feedback loop that I have not seen anywhere else. In fact, more than 90% of all new features and services that we deliver are driven by customer requests.

**At the moment, AWS is quite a bit bigger than your competitors. How will you continue to differentiate yourselves in the future?**

First, let me say that this is not going to be a winner-take-all market. There's going to be a handful of global companies that are going to be really successful in this space, and that's good, because customers need choices. The major thing that separates AWS from most other places is that we don't follow our competitors. We focus on the feedback that our customers are giving us, and we'll continue to go down that path—and at the same time, be a pioneer.

**I'm reminded of the famous and probably apocryphal Henry Ford quote about how, if he'd asked for their feedback, his customers would have simply requested faster horses.**

We are innovators in this space, and we also have to be conscious that we use our inventive brain power not just to fix a particular customer's problem but, whenever possible, solve for issues that apply to many more businesses. That's one of my major tasks, to listen to different types of customers and try to find the bigger patterns.

I imagine that Now Go Build, a video series in which you visit global entrepreneurs who use Amazon's cloud-based technologies to do things like support Indonesian farmers and integrate refugees into the German workforce, has given you a nice window onto that.


Now Go Build is basically a recording of something that I've been doing for the past four or five years—meeting companies all over the world that have tremendous impact that AWS has helped them achieve. As technologists, we sometimes get so wrapped up in technology that we forget about the end-user stories. Most of these young businesses are solving really hard problems. They're not hard problems in the technology sense, but they're hard problems in the human sense.

I'm still so proud of the first episode that we did in Jakarta, where I met the founders of a start-up called Hara Token. These guys provide identity verification for the poorest farmers in Indonesia so they can get loans at regular banks instead of using loan sharks, who charge interest rates of 60%.

**It must be a very interesting exercise—and very humbling.**

The current general view of technology companies is that they focus on uninhibited growth or on profit. But these are entrepreneurs trying to build a business while also solving very difficult human problems.

**In 2006, Jim Gray interviewed you in ACM Queue and noted how much it pains you to hear people describe Amazon as an online retailer, rather than a technology company. Now, I don't think anyone would question your dominance in the tech world.**

Jim was a mentor to me for a long time, and there are so many things in that interview that are still true today. His thinking has had a significant impact on technology architectures around the world, including at Amazon and AWS. I think we should continue to honor his memory and be grateful for all of the things that he impacted. 

Leah Hoffmann is a technology writer based in Piermont, NY, USA.

© 2020 ACM 0001-0782/20/2 \$15.00

## Q&A

# ‘Everything Fails All the Time’

*Werner Vogels, an expert on ultra-scalable systems, talks about listening to customers, reconceptualizing the stack, and building a product-centered culture.*

AMAZON VICE PRESIDENT and chief technology officer Werner Vogels has kept the company at the forefront of scalable systems since he joined in 2004. Vogels was one of the main driving forces behind the architecture of Amazon Web Services (AWS). His insistent reminders that “everything fails all the time” have been hugely influential to the developers of high-availability systems. Here, he talks about continuing to innovate in database technology—and keeping up with customer needs.

**You have followed an unconventional professional path, working at the Netherlands Cancer Institute before returning to school for a Ph.D. in computer science. What drew you to scalable systems?**

Originally, computer science just seemed like a good career. In that era—the mid-1980s—there were many areas that were completely undiscovered, and distributed systems was one of them. I liked research, and it turned out I had a gift for it.

**Let’s talk about Amazon Aurora, which recently won the 2019 ACM SIGMOD Systems Award. When did you first realize the time had come to reconceptualize the stack?**

Our biggest goal with Aurora was to make life easier for our customers, and to do that, the database required a lot of modern innovations. Most database architectures, which



**“If we could just rip the whole architecture apart and redesign it in a reliable, high-scale, cloud-native manner, we would suddenly be able to build all the functionalities that we wanted our customers to have.”**

were developed in the 1990s, don’t offer a good base for that—nor, even, do open-source databases like PostgreSQL and MySQL. We realized that if we could just rip the whole architecture apart and redesign it in a reliable, high-scale, cloud-native manner, we would suddenly be able to build all the functionalities that we wanted our customers to have.

**Such as?**

Modern database users expect everything to be serverless and get seamless replication over at least three data-centers. They expect to be able to have instant backups and to be able to do instant schema changes without having to do complete table copies. We could

only meet those requirements if we revamped how the database was built.

**It must have been exciting to have an opportunity to rethink what a modern database should look like.**

Building upon a db-log-aware storage service was a shift in mind-set, but we knew we had the engineers and the knowledge. Old-style database transactions are quite expensive, easily triggering 10–15 storage operations, because they were developed with an attached disk in mind.

**AWS reinvented how companies access data storage and infrastructure—supported by a lot of education on your part.** [CONTINUED ON P. 95]

## CALL FOR LONG AND SHORT PAPERS

# ICMI 2020 UTRECHT



22nd ACM International Conference on Multimodal Interaction  
Utrecht, The Netherlands, Oct 25-29, 2020

The 22nd International Conference on Multimodal Interaction (ICMI 2020) will be held in Utrecht, the Netherlands. ICMI is the premier international forum for multidisciplinary research on multimodal human-human and human-computer interaction, interfaces, and system development. The conference focuses on theoretical and empirical foundations, component technologies, and combined multimodal processing techniques that define the field of multimodal interaction analysis, interface design, and system development.

We are keen to showcase novel input and output modalities and interactions to the ICMI community. ICMI 2020 will feature a single-track main conference which includes: keynote speakers, technical full and short papers (including oral and poster presentations), demonstrations, exhibits and doctoral spotlight papers. The conference will also feature workshops and grand challenges. The proceedings of ICMI 2020 will be published by ACM as part of their series of International Conference Proceedings and Digital Library.

### Important dates

Paper submission  
May 4, 2020

Reviews to authors  
July 3, 2020

Rebuttal due  
July 10, 2020

Notification of  
acceptance  
July 20, 2020

Camera-ready  
paper  
August 17, 2020



### General Chairs

Khiet Truong, University of Twente, NL  
Dirk Heylen, University of Twente, NL  
Mary Czerwinski, Microsoft Research, USA

More information?  
<http://icmi.acm.org/2020>

  /acmicmi

# 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](mailto:dl-info@hq.acm.org)

ACM

DL

DIGITAL  
LIBRARY