COMMUNICATIONS OF THE ACM

CACM.ACM.ORG

10/2019 VOL.62 NO.10

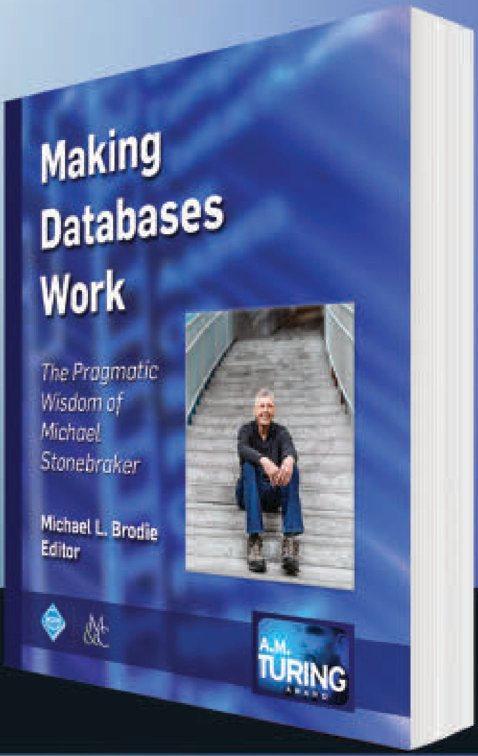
Unlocking Data to Improve Public Policy

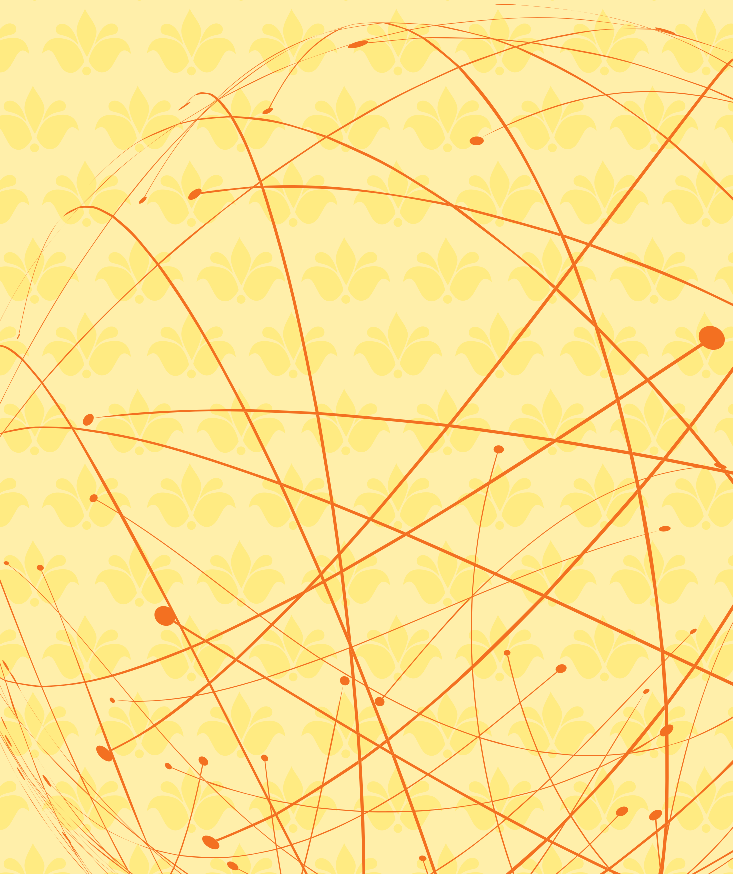
Closing in on Quantum Error Correction

Building Certified Concurrent OS Kernels

Consumer-Grade Fabrication to Revolutionize Accessibility

Association for Computing Machinery


Making Databases Work
The Pragmatic Wisdom of Michael Stonebraker
Michael L. Brodie Editor
A.M. TURING AWARD

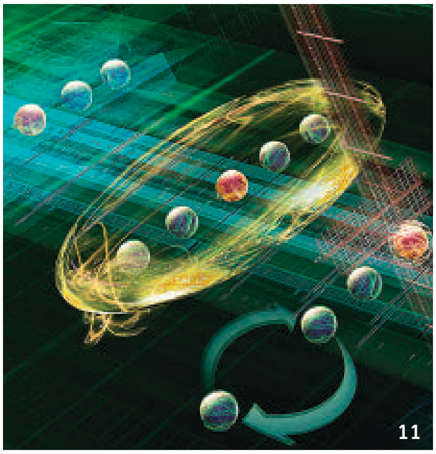


COMMUNICATIONS OF THE ACM

Departments

Last Byte

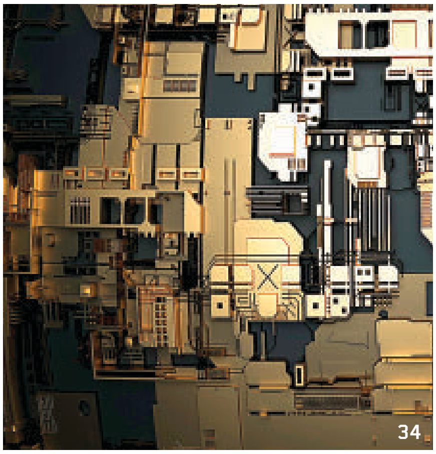
News



Viewpoints

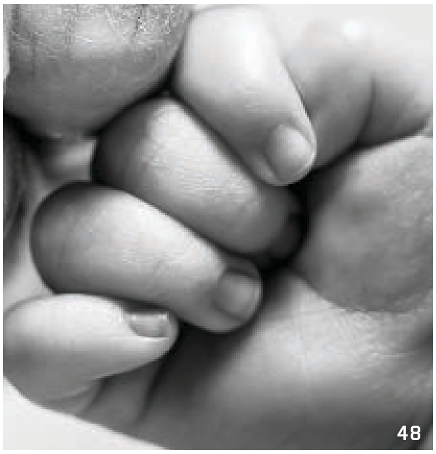
IMAGE: IQOQI INNSBRUCK/HARALD RITSCH

Practice



Articles' development led by acmqueue
queue.acm.org

Contributed Articles

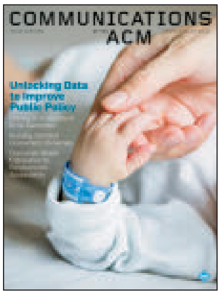


Review Articles

Watch the authors discuss this work in the exclusive *Communications* video. https://cacm.acm.org/videos/protein-design-by-provable-algorithms

Research Highlights

IMAGES: (L) PHILIPP TUR; (R) SIMON DANNHAUER



About the Cover:
This month's cover story explores how the collection of administrative data from multiple sources can be used to inform policymakers. The authors detail their experience with a real-world policy to support low-birthweight newborns and their mothers. Cover photo by Andrii Orlov.

Association for Computing Machinery
Advancing Computing as a Science & Profession

COMMUNICATIONS OF THE ACM

Trusted insights for computing's leading professionals.

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

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

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

WEBSITE
http://cacm.acm.org

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

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

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

Media Kit acmmediasales@acm.org

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

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

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

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

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

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

Printed in the USA.

Association for Computing Machinery

PLEASE RECYCLE THIS MAGAZINE

DOI:10.1145/3357715

Cherri M. Pancake

How ACM Evolves in Response to Community Needs

One question people ask me as President is how ACM—as a global, volunteer-based organization—can evolve over time. We have all seen new publications, conferences, SIGs, and chapters added to the ACM family as technology grows. But what about the organization itself? Does it also evolve to keep up with changes going on in our profession?

The basic structure of ACM doesn't change frequently, but it does get adjusted from time to time in response to new priorities and needs. If you are not familiar with ACM's governance structure, there are four boards that manage our core products and activities: Publications, Education, Practitioners (Lifelong Learning), and SIG Governance. A system of regional councils was created to respond to emerging needs in different geographic areas (currently Europe, China, and India). This year, ACM has added the concept of thematic councils, which leverage products and activities from all the boards and regional councils to address important challenges for our profession.

Ensuring Technology Reflects Its Full Audience

One key to ACM's future health is to ensure its governance and activities involve people from diverse backgrounds. Diversity is defined very generally to encompass not just physical characteristics like gender, age, race, and disabilities, but also characteristics related to a person's career, such as institution type and size, disciplinary domain, workplace role, stage in career, and location where an individual resides. Ideally, ACM would be balanced in terms of the individuals, institutions, and disciplines represented. However, our members, volunteers, and activity participants are self-selecting. We can aim to be proactive—and more effective—in recruiting volunteers from among diverse audiences, but most of our efforts need to focus on inclusion. That is, we must try to become more diverse by creating environments that are welcoming of new and perspectives, and where hostility and other antisocial behaviors are not tolerated.

If you have registered for an ACM conference in the last couple of years, you have seen that we have an organizationwide Policy Against Harassment at ACM Activities (http://bit.ly/2LcV88z), which lays out the expectations for professional behavior at all ACM activities, including committee meetings. In addition, many ACM's SIGs have programs that specifically address diversity. These include awards, travel grants, accessibility reviews, targeted mentoring, fellowships, and competitions to promote broader participation in computing. In July, a new Diversity and Inclusion Council was established to identify "best practices," document them, and encourage their broader adoption across the organization. Led by co-chairs Natalie Enright Jerger and John West, the Council will also reach outside ACM to find new strategies and partnerships that can help improve diversity in the broader community.

Taking Responsibility for Technology We Create

You already know about the ACM Code of Ethics and Professional Conduct (www.acm.org/code-of-ethics), which has received a lot of attention since the updated version was released last year. The changes reflect the fact that for some time now, ACM has been discussing what kinds of ethical and social responsibilities we should take for the technology we create and deploy. You have only to look at past issues of *Communications* to find that the risks and challenges have been discussed in virtually every issue in recent years.

Technology, developed and deployed around the globe, is at the heart of some of the most pressing issues we face as a society. Concerns about data breaches, election security, digital privacy, surveillance, and the future of the Internet stretch beyond national borders and go to the heart of how we live, work, and interact with one another. As a professional society whose members are at the forefront of developing these technologies, ACM has had active policy forums in the U.S. and Europe for many years. These activities have been instrumental in providing technical expertise and advice to policy-makers at a variety of levels. In July, we established a global Technology Policy Council to strengthen and extend ACM's policy efforts even further. Led by Lorraine Kisselburgh, it brings together leading experts to lend a global perspective to the social and ethical challenges posed by technology. One of their first initiatives is to develop a new series of bulletins presenting a balanced perspective on the impact of specific developments or applications of technology, targeted not just at decision-makers but also our general membership and the press.

I hope you are as excited as I am about these two new initiatives. To keep up on what they are doing or to send your own ideas, visit the ACM website.

Cherri M. Pancake is President of ACM, professor emeritus of electrical engineering and computer science, and director of a research center at Oregon State University, Corvallis, OR, USA.

ACM Transactions on
Internet of Things

DOI:10.1145/3359332

Vinton G. Cerf

AI Is Not an Excuse!

I keep hearing excuses for not working on difficult problems: "Eventually AI will solve this so there's no point working on it now." Sorry, wrong answer.

First, we should be cautious about putting too much expectation on artificial intelligence which, by most metrics, is really today's machine learning (ML) and neural networks. There is no doubt these systems have produced truly remarkable results. The GO game story of AlphaGo and AlphaZero from Deep Mind is by now a classic example of the surprisingly powerful results these systems produce. A chess-playing version of AlphaZero learned quickly and demonstrated choices of moves unlike those of traditional chess players. I hesitate to label this *strategy* but the deep neural networks do encode in some deep way *experience* that one could consider a kind of strategic cache.

These systems are also quite brittle and can break in ways that are not always, to my understanding, predictable. Rather, we might predict there will be circumstances in which the ML system will fail but we may not know how. It's a bit like the insurance data that indicates 1% of all males over the age of 85 will die next year—we just don't know who is in the 1%! It seems prudent, then, to anticipate these fragile possibilities and research how they might be characterized and even identified based on the design of the neural network. I think we know, for example, that image classification systems do not work the way human classification works. The systems are much more sensitive to pixel input than humans. We abstract from raw pixel input, recognizing shapes and characteristics ("pointy ears," for instance), while the image recognition ML systems are much more sensitive to pixel-level inputs. Changing a small number of pixels can lead to significant misclassifications.

With these frailties in mind, it seems to me very important not to make too many assumptions about the power of machine learning. I do want to acknowledge, however, that considerable successes have been recorded well beyond playing board games. At Google, a ML system was trained to control the cooling system in its datacenters and saved 40% of the power needed to operate them. Machine speech recognition allows Google to converse via its Assistant and to do automatic language translation. While it can be argued the machine does not *understand* what is said, the ML system can process the input and produce useful output ('What's the weather in Palo Alto?" "It's 78 degrees in Palo Alto with a high of 82 and a low of 68"). There are pathogen and disease image recognition systems helping to identify patients at risk. There is a lot to praise and admire about these applications.

It seems to me very important not to make too many assumptions about the power of machine learning.

In the meantime, however, I think it is not okay to ignore difficult problems on the assumption they will be solved "automagically" by ML tools. We have huge challenges ahead with *ordinary* software that we cannot reasonably assume will be solved by AI. Software analysis for potential mistakes or bugs requires tools I would not identify with traditional AI or ML. Designing systems to be updated reliably with new software doesn't require AI but it does require careful thinking about authentication of the origin of the software update and confirmation it has retained its integrity during its journey from the source to the updated device. Security in general is not solely the purview of AI or ML. Interestingly, some aspects of security will be addressable such as fraud detection. Credit card companies are making good use of modeling to detect unusual card usage and flag anomalous events for further analysis.

Bottom line: Let's enthusiastically explore the uses of machine learning and artificial intelligence but not use their potential to excuse ourselves from crafting high-quality, reliable software that is resistant to abuse!

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

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

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

DOI:10.1145/3358698 http://cacm.acm.org/blogs/blog-cacm

Pinning Down Variables, and Taking an Agile Approach

Robin K. Hill tries to better define variables, while Edwin Torres acknowledges he is his own greatest obstacle.

Robin K. Hill
Variable Vagaries

May 19, 2019
http://bit.ly/2YHWMYc

A few months ago, I wrote in this space some speculation under the question, "What is a Variable?"[3] and would now like to explore that question a bit further. The *Dictionary of Computer Science*[1] says that a variable is:

1. A unit of storage that can be modified during program execution, usually by assignment or read operations. A variable is generally denoted by an identifier or by a name.
2. The name that denotes a modifiable unit of storage.
3. See *parameter*.
4. (in logic) a name that can stand for any of a possibly infinite set of values.

For someone who appreciates a pesky problem, this is great stuff. Alternative 3 passes the buck, alternative 4 doesn't get us very far, and alternatives 1 and 2 are the two of the views we are trying to pin down and reconcile. (The reader should not infer condemnation of this definition or promise of a better one.)

That piece on variables in January garnered some comments. Vincent Di Carlo mentioned the work of Simon Funk, with references, who observes and illustrates some of the difficulties connected to regarding a variable as a container, and the use of predication to alleviate that difficulty.[2]

Peter O'Hearn mentioned Christopher Strachey, for which I commended and thanked him. But on reflection—that wasn't enough. Strachey's remarks on this subject deserve more attention.[5] He brings up the complexity of the L-value (the address, or location; the variable identifier, in other words, on the left side of an assignment). He shows expressions in that place, and explicitly allows functions. We are completely used to the left-hand side of an assignment statement sporting an expression like A[i], an array name with an index to a particular element. This, of course, requires evaluation; in fact, this L-value is a function. Examples of L-values such as A[i+1] make that even more cogent. The variable itself varies, in other words, an affordance that has been played down in our more mechanical and precise development of programming languages.

This might suggest something about the informal view I attempt to exhibit. Here is a lightly fictionalized story of variables from my small town, where everybody knows the local business-people.

I said to a friend, "I made a mistake— the sewing machine repairman's name is not Victor, but I can't remember what it is." She said (now read carefully), "No, it's not not Victor. It's actually not Mercer." Right, of course ... Victor was the repairman for vacuum cleaners, in the same shop, before then. But Mercer was the proprietor of the sewing shop, not the repairman. And she knew I was thinking of Mercer. As for the actual sewing machine repairman who was working for Mercer—let's call him X—he plays no part, and I have no clue as to his real name, except that it was neither Victor nor Mercer.

A gross translation of the pertinent sentences looks like this:

Not(*X*) = *Victor*	We're calling this false!
Not(*X*) = *Mercer*	We're calling this true!

I refrain from expressing those in any notation approaching symbolic logic for fear of inducing some sort of modeling monstrosity. What is the variable at play here? It's X only in the sense that it's not. X is the foil. If X were the principal, we could reason this way:

X != Victor
X != Mercer

Both are true, so that the imposition of standard semantics is not particularly revealing, and does not explain in what sense my friend was right. That's not what I was saying in the story; rather, I was shamelessly assigning to the L-value *Not*(*X*). As I recall, I had grasped the presence of negation when I entered the shop, once or twice, and called the repairman "Mr. Mercer," to which he finally replied, "I'm not Mercer. He owns this place." I had forgotten the details, but recalled the negation, and misdirected it. Does recalling it mean the negation itself is the value of some variable, or is the negation a variable with a name as its value?

I offer no answer to that. It would be interesting as an exercise in formal representation under the laws of some system, perhaps generating a shadow referent for the placeholder that eventually resolves correctly into Mercer on its way to expressing a lack of identifier for X. The predications discussed by Funk might show the way. Yet the remarkable thing is that we tolerate, communicate, exchange, and exploit all those vagaries. We deal with them with no apparent discomfort, but rather recognition of the quirkiness with a laugh. William Kent makes the point that, even though we cannot draw a principled distinction between the attribute and the relationship, the terms are still useful, which might be the most intriguing result of all.[4]

So when our students ask what is a variable, we could state one of the variants given by the *Dictionary of Computer Science*, or something simple like "an expression for a memory location." They hardly ever ask. They learn it by exposure and experience, like we did.

Let's leave the last word to Strachey, in support of the motto he suggests:

> *If we attempt to formalize our ideas before we have really sorted out the important concepts, the result, though possibly rigorous, is of very little value—indeed, it may well do more harm than good by making it harder to discover the really important concepts. Our motto should be 'No axiomatization without insight'.*

References

1. Butterfield., A., and Ngondi, G.E., Editors. *A Dictionary of Computer Science.* Oxford University Press, 7th edition, 2016.
2. Funk, S., *On Knowledge Representation.*, 2013
3. Hill, R.K., January 31, 2019, "What is a Variable?" BLOG@CACM, http://bit.ly/2KdGzCa
4. Kent, W. Data and Reality. North-Holland Publishing Company. Chapter 5.0. Attributes, 1978.
5. Strachey. C.. *Fundamental concepts in programming languages.* These lecture notes for the International Summer School in Computer Programming, 1967, are published in Higher-Order and Symbolic Computation, 13, 11–49, 2000, Kluwer Academic Publishers.

Edwin Torres
An Agile Approach to Learning Programming

http://bit.ly/2NqVQl4
March 29, 2019

I recently read a book called *Rework* (https://amzn.to/2zgVEgc) and had a revelation. I have been an inefficient software engineer and flat-out showstopper at times throughout my career. Like many previous books, *Rework* claims you *can* create your own business. What makes this approach different is that it shows you that most of the obstacles are *you*. To oversimplify, just make progress, inch forward, build something that works and refine later. In other words, take an Agile approach. Rework convinced me my obstacles are not really obstacles. Now my new business venture is almost a reality. Finally.

I quickly realized the lessons I learned from *Rework* are applicable to other areas of my life. One such area is teaching. I teach programming to students and industry professionals. I often see new programmers struggle with basic, fundamental concepts. I have found that taking an Agile approach to teaching these students is extremely effective.

First, most programming books, even beginner books, are overwhelming. New programmers find it intimidating when faced with everything a programming language has to offer. In reality, a programmer only needs a subset of commands to complete a given task. And it is not necessary to learn multiple ways to complete a task, when all you need is one. So I wrote *The Super Simple Programming Book* (https://amzn.to/2zeElMv) to teach basic programming concepts using the Python programming language. The book is a quick read, with lots of short, clear programming examples. Its purpose is to give a brief introduction, teach fundamental programming concepts, and help the programmer quickly move on to more advanced concepts. This is iteration, quick learning, and progress, just like Agile and *Rework*.

Second, my classes incorporate elements of the Agile methodology. A class is like a sprint planning meeting. I describe the content and assignments for the week, and there is a classroom discussion. At the end of the class, students begin the sprint (that is, learning). We even have a Slack (https://slack.com/) team to conduct real-time persistent chats outside of the classroom. Students may submit assignments (such as prototypes) early, get feedback, and resubmit for full credit. At the start of the next class, there is a sprint retrospective; students describe what went well and what was difficult. This is also an opportunity for me to enhance course materials for the next class. And the process repeats (iterative).

My teaching method loosely follows the Agile method, but it is Agile, nonetheless. Ever since I started using this approach, I have noticed a significant improvement in grades and student attitudes. Students often say things like, "your class makes programming easy" or "I never understood programming until now." Agile lets you learn as you go and build on that learning. It has worked well in my classes.

Robin K. Hill is a lecturer in the Department of Computer Science and an affiliate of both the Department of Philosophy and Religious Studies and the Wyoming Institute for Humanities Research at the University of Wyoming. **Edwin Torres** is a full-time software engineer at The MITRE Corporation and an adjunct professor of computer science at Monmouth University.

The Essentials of Modern Software Engineering

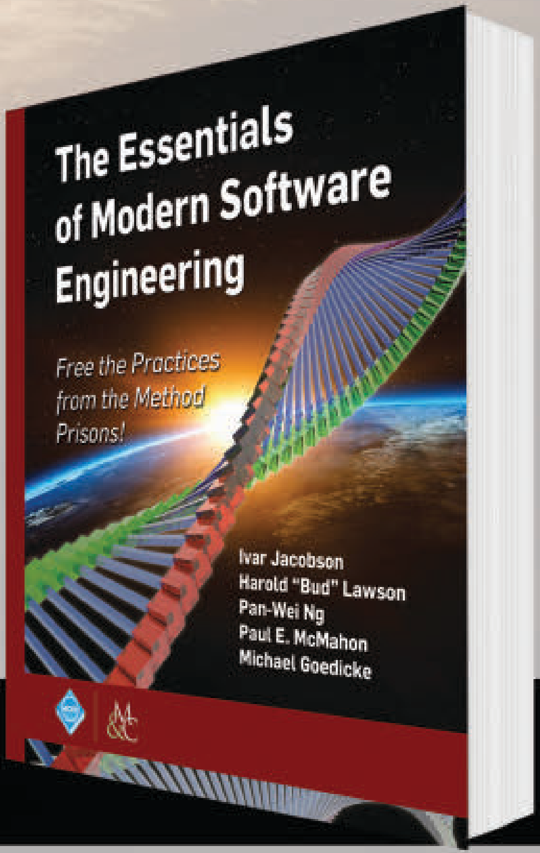
Free the Practices from the Method Prisons!

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


The Essentials of Modern Software Engineering
Free the Practices from the Method Prisons!
Ivar Jacobson
Harold "Bud" Lawson
Pan-Wei Ng
Paul E. McMahon
Michael Goedicke

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

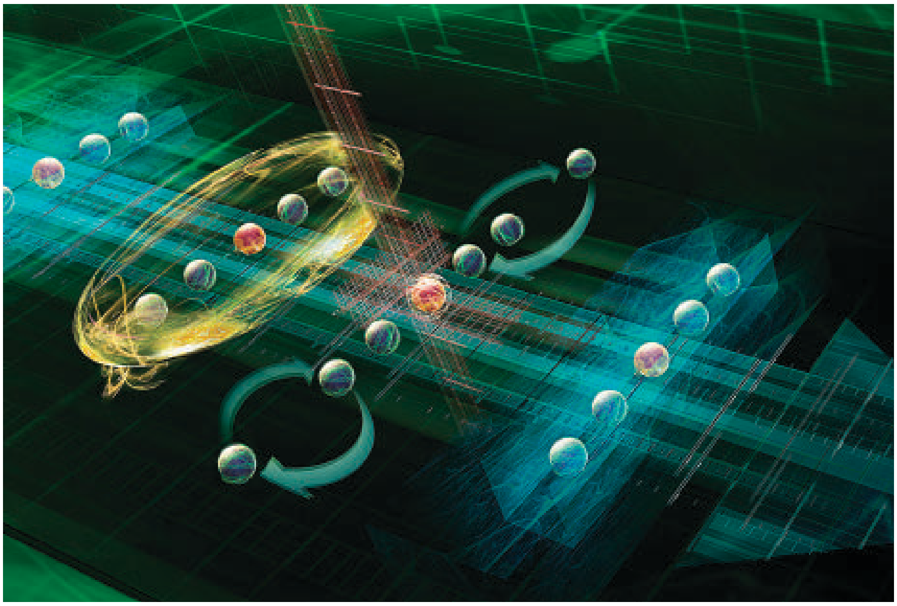
Science | DOI:10.1145/3355371

Don Monroe

Closing in on Quantum Error Correction

Quantum computers will only become practical when they implement quantum error correction.

AFTER DECADES OF research, quantum computers are approaching the scale at which they could outperform their "classical" counterparts on some problems. They will be truly practical, however, only when they implement quantum error correction, which combines many physical quantum bits, or qubits, into a logical qubit that preserves its quantum information even when its constituents are disrupted. Although this task once seemed impossible, theorists have developed multiple techniques for doing so, including "surface codes" that could be implemented in an integrated-circuit-like planar geometry.



"Complicated calculations fail as the systems get out of hand due to perturbations," says Rainer Blatt of the Institute of Experimental Physics at the University of Innsbruck and the Institute of Quantum Optics and Quantum Information. "Using error correction, this process can be contained."

IMAGE: IQOQI INNSBRUCK/HARALD RITSCH

For ordinary binary data, errors can be corrected, for example, using the majority rule: A desired bit, whether 1 or 0, is first triplicated as 111 or 000. Later, even if one of the three bits has been corrupted, the other two "outvote" it and allow recovery of the original data. Unfortunately, the "no-cloning" theorem of quantum mechanics forbids the duplication (or triplication) of a qubit.

Moreover, the power of quantum computing emerges from having arbitrary mixtures of bit values. Since any combination is valid, it would seem impossible to detect a change from the original combination, let alone correct it.

"Most people who were doing quantum information in the '80s and '90s would say we'll never be able to do error correction in these systems, because it's analog error correction," explained Raymond Laflamme of the University of Waterloo and the Perimeter Institute, both in Waterloo, Ontario.

Fortunately, those experts Laflamme spoke of were mistaken.

Quantum Computing

Unlike a classical bit that has a value of either 0 or 1, a single qubit can encode a weighted "superposition" of both values. A set of N qubits can thus represent 2^N different states simultaneously. There is strong mathematical evidence that devices handling qubits could solve some large problems much faster than traditional computers could, even in principle.

In the mid-1990s, mathematician Peter Shor, then at AT&T Bell Laboratories in Murray Hill, NJ, described a powerful example of using such a quantum algorithm to efficiently factor large numbers. Because public-key encryption relies on this task being impractically slow, the result helped transform quantum computing from a physics curiosity to a technology with potential national security implications.

Quantum computations exploit the fact that the ultimate weights assigned to different configurations of qubits are the sum of contributions that can be positive, negative, or even complex numbers. The algorithms combine qubits so that these contributions reinforce one another for the desired solution, but cancel each other out for the many incorrect configurations. A key step in Shor's algorithm, for example, is finding the repetition period of a sequence of numbers generated from the factoring target, essentially by performing a Fourier transform on the entire sequence that produces a peak at the correct periodicity.

The final step is measuring the qubits to see which are 1 and which are 0. The probability of a particular outcome is proportional the square of the corresponding weight. The measurement leaves each qubit unambiguously in the measured state.

Proper cancellation demands that quantum weights be preserved ("coherent") until measurement. However, qubits are highly sensitive to uncontrolled interactions with their environment, which effectively measures them prematurely. Experimenters have been steadily reducing the rate of this decoherence in various candidate systems, but some errors remain inevitable.

Digital Quantum Correction

Not long after introducing the factoring algorithm, Shor and, independently, Andrew Steane of Oxford University showed how these errors could be corrected. Just as "quantum mechanics is both a wave and a particle, quantum computing is both digital and analog," said Shor, now at the Massachusetts Institute of Technology (MIT). "You have to isolate the error correction into the digital piece," protecting the analog piece.

This isolation is achieved by measuring a carefully chosen combination of qubits. Before this measurement, a particular qubit might have been disturbed by its environment so that, for example, a pure 1 gains some small admixture of 0. If the measurement returns 1, however, this admixture is erased. Sometimes, however, the measurement will return 0, meaning that there has been a complete "spin flip."

"The errors live in a continuous space, but when we do quantum error correction the way it's conventionally done, by measuring [particular qubits], that in effect digitizes the errors," said John Preskill of the California Institute of Technology. "Then when you know what the error is, you know what operation to apply to correct it."

Critically, Preskill added, "You don't want to measure the logical qubit because that would disturb it," destroying its usefulness for further calculation. To avoid this, the logical information is encoded not in individual qubits, but in their relationship, known as entanglement.

This procedure can be illustrated with an oversimplified version of Shor's scheme, which superficially resembles the classical majority-rule, with a logical 1 represented by three physical qubits 111 and a logical 0 by 000. If one qubit undergoes a spin flip from 0 to 1 or 1 to 0, the three bits no longer agree, but measuring them all to find out which one flipped would destroy any quantum information.

"Quantum computing is both digital and analog. You have to isolate the error correction into the digital piece," protecting the analog piece.

Instead, one can measure, for example, whether the first two qubits disagree and whether the second two qubits disagree. These two measurements suffice to determine whether one bit was flipped, and which of the three it is, without providing any information about what the actual values are. The disturbed physical qubit can then be flipped back, even without knowing its value. The actual measurement of whether two qubits agree is done by introducing extra "ancilla" qubits in a well-defined state and introducing quantum gates that let the other qubits modify the ancilla, transferring the noise to it.

Generalizing Codes

Shor's scheme actually used nine physical qubits, which also corrects a phase-flip error or combinations of a spin flip and phase flip. Subsequent work by Laflamme and others achieved single-qubit correction with five physical qubits.

In the quarter-century since the first codes, theorists have devised many new schemes as they wait for experimental systems complex enough to test their ideas. To improve coding efficiency and to protect against more complicated errors, researchers developed more complex codes that allow correction of multiple bits.

Many of these schemes are "stabilizer codes" that, like Shor's, measure specific combinations of qubits to force the system to either retreat to its undisturbed state or to reveal specific "error syndromes" that allow the error to be corrected. Importantly, some codes guarantee perfect accuracy, as long as the physical error rate is below some threshold.

Recently, interest has turned to "surface codes," which can be implemented in two-dimensional geometry like computer chips, using only neighboring qubits. Most importantly, said

Preskill, "surface codes can tolerate a higher noise rate than other families of codes that we know about," which will be critical for the error-prone first generations of machines.

However, the surface codes require many physical qubits per logical qubit. Indeed, with current experimental error rates in multi-qubit devices somewhat below 1%, large calculations like those needed for quantum chemistry might require 1,000 physical qubits per logical qubit, Preskill said. "If the error rates were considerably lower, then maybe there would be better things to do than the surface code."

One alternative was found, surprisingly, by physicists studying the nexus of general relativity and quantum mechanics, who determined that an exotic mathematical connection between these fields embodies quantum error-correcting codes. "They are efficient in the sense that they can protect a lot of information without a lot of overhead," said co-discoverer Daniel Harlow, now at MIT. Still, he cautions that, for now, "there are a lot of practical hardware considerations that are probably going to be more important."

Prospects

As theorists explore error-correction schemes, experimentalists have been steadily improving various physical implementations of qubits. Although there is still no clear winning platform, some groups have demonstrated devices with more than 50 qubits.

One such team, led by John Martinis at Google and the University of California, both in Santa Barbara, CA, studies qubits based on superconducting circuits. Their near-term goal is to demonstrate "quantum supremacy" with these modest-size uncorrected devices, and only later to implement error correction and larger assemblies. Nonetheless, Martinis said, "Everyone understands that in the long term we want to do error correction."

The theory underlying error digitization is clear, but it still must be thoroughly tested experimentally, Martinis said. "Maybe there's some new things you don't understand, or maybe the requirements are a lot harder than you think experimentally ... This whole error model has to work so that you can get exponentially small errors" even in complex calculations.

As theorists explore new error-correction schemes, experimentalists have been steadily improving various physical implementations of qubits.

Experiments to date show that "the cross-talk between qubits is more important than what people had thought," Laflamme said, for example because the control circuitry can affect multiple physical qubits. "The biggest issue is how correlated the noise is," Preskill agrees. "It may be that as things advance, the scalability prospects will look better for some platforms than others just on the basis of how effective quantum error correction is."

Error correction will be an important milestone in the field, Laflamme said. "If we would be able to do fully fault-tolerant quantum error correction today, we would have a quantum computer." C

Further Reading

Fowler, A.G., Mariantoni, M., Martinis, J.M., and Cleland, A.N.
Surface codes: Towards practical large-scale quantum computation, *Phys. Rev. A* 86, 032324 (2012)

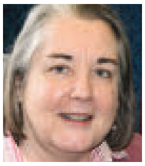
Beale, S.J., Wallman, J.J., Gutierrez, M., Brown, K.R., and Laflamme, R.
Coherence in quantum error-correcting codes, *Phys. Rev. Lett.* 121, 190501 (2018)

Wolchover, N.,
How Space and Time Could Be a Quantum Error-Correcting Code, *Quanta Magazine*, January 19, 2019, http://bit.ly/2Xrnpgg

Don Monroe is a science and technology writer based in Boston, MA, USA.

ACM Member News

MANAGING INFORMATION RETRIEVAL WHERE MEASUREMENTS HAPPEN



"When I was in high school, I took a class in computer science, and the logic behind programming was attractive to me," recalls Ellen Voorhees, a computer scientist at the U.S. National Institute of Standards and Technology (NIST) in Gaithersburg, MD. "It was a puzzle. When you programmed something successfully, it was very tangible that you had succeeded."

Voorhees earned her undergraduate degree in computer science from Pennsylvania State University, and her master's degree and doctorate in computer science from Cornell University. She worked for Siemens Corporate Research in Princeton, NJ, before joining NIST.

"NIST is the place where measurements happen," Voorhees says. The agency serves as the official arbiter of measurements in the U.S.

Voorhees works in the retrieval group at NIST, where her research focus is on document retrieval, information retrieval, and natural language processing.

One of her responsibilities at NIST is to manage the Text REtrieval Conference (TREC) project, a workshop series supporting the information retrieval community by building the infrastructure needed for large-scale evaluation of retrieval technology.

Voorhees says the big push in the information retrieval field right now is artificial intelligence, using deep learning methods. "The real question in information retrieval," Voorhees says, "is whether these deep learning methods are actually better than existing retrieval algorithms."

A future challenge, she adds, will be to properly assess not only information retrieval evaluations, but other types of natural language processing assessments as well.

—*John Delaney*

Technology | DOI:10.1145/3355377

Keith Kirkpatrick

Protecting Industrial Control Systems

Finding, and plugging, the security holes in SCADA.

While most commercial and government organizations have a corporate network to handle administrative, sales, and other back- or front-office data, a growing number of organizations also have implemented one or more supervisory control and data acquisition (SCADA) systems. These systems incorporate software and hardware elements that allow industrial organizations, utility companies, and power generators to monitor and control industrial processes and devices, including sensors, valves, pumps, and motors. Today's SCADA systems also allow organizations to harvest data from these devices, and then to analyze and make adjustments to their operational infrastructure to improve efficiency, make smarter decisions, and quickly address system issues to help mitigate downtime.

Real-world attacks over the past decade have sought to exploit the vulnerabilities in supervisory control and data acquisition (SCADA) systems.

IMAGE COURTESY OF SIEMENS

A typical SCADA architecture consists of programmable logic controllers (PLCs) or remote terminal units (RTUs), which are small computers used to communicate with manufacturing equipment, human-machine interfaces (HMIs), sensors, and other end devices, and then route the information from those objects to computers equipped with SCADA software. The SCADA software collects, processes, distributes, and displays this data, helping operators and other employees analyze the data and make important decisions.

Because SCADA systems are designed to connect and control a huge amount of industrial equipment, malevolent actors see significant value to infiltrating or controlling these systems and the operational technology (OT) networks through which they send and receive data. Data collected and compiled by X-Force Red, an autonomous team of hackers within IBM Security that was hired to uncover security vulnerabilities, illustrated the number of vulnerabilities exposing industrial control systems has increased 83% from 2011 to 2018. Moreover, over the past decade, there have been a number of real-world examples of attackers targeting SCADA systems:

- In 2010, malware created by the intelligence forces of the U.S. and Israel, known as Stuxnet, was used to destroy centrifuges used in the enrichment of uranium at a facility in Iran, thereby delaying the development of that country's nuclear weapons.
- In 2015, BlackEnergy, a Trojan Horse virus (which sits undetected until the attacker decides to activate it), was adapted by Russian hackers to infiltrate several Ukrainian power companies, with the malware used to gather intelligence about the power companies' systems, and to steal login credentials from employees.
- In 2016, malware known as CrashOverride or Industroyer was deployed by Russian cybercriminals to attack a part of Ukraine's electrical grid. CrashOverride replicated the communication languages, or protocols, that are used by different elements of an electrical grid to talk to one another, which allowed the hackers to strike at an electrical transmission substation in Kiev, resulting in a short blackout of part of that city.
- In the summer of 2017, hackers deployed malware known as Triton, which was named for the Triconex safety controller model that it targeted, against a petrochemical plant in Saudi Arabia. The malware allowed the hackers to take over the plant's

safety systems remotely, though a flaw in the code allowed the plant to respond before any damage occurred.

These attacks, along with numerous others, highlight the vulnerability of SCADA systems and industrial networks.

In 2018, the International Society of Automation (ISA) helped to develop a series of industrial cybersecurity standards designated ISA/IEC 62443, which were designed to protect the industrial automation and control systems (IACS) and networks that operate OT machinery and associated devices within critical infrastructure. The ISA/IEC 62443 standards also serve as key components of the U.S. Framework for Improving Critical Infrastructure Cybersecurity (released in April 2018), a how-to guide developed through the National Institute of Standards and Technology (NIST) in support of U.S. cyber defenses.

NIST also released last year Internal Report 8219, "Securing Manufacturing Industrial Control Systems: Behavioral Anomaly Detection." which notes that NIST is trying to add anomaly and malicious user detection to OT networks, and has documented the use of behavioral anomaly detection (BAD) systems in two demonstration environments, including one that mimicks a process control system that resembles what is being used by chemical manufacturing industries.

Industrial operators would be wise to follow NIST's recommendations, as SCADA systems that have been compromised could be hijacked by hostile actors, with potentially serious outcomes. For example, organizations can have their operations halted until a ransom payment is made, or physical systems or processes can be sabotaged, resulting in significant damage. Perhaps the most dangerous scenario is that bad actors, likely cyber teams from or working on behalf of foreign governments, could plant malware that lies dormant in a power plant, electrical distribution grid, or municipal water supply plant, so it may used as a point of leverage at a future date.

"What we are seeing in the field is that there's an increase, not just of cyberattacks, but what we call cyberattack readiness, or red button capability," says Barak Perelman, CEO of Indegy, a New York City-based industrial cybersecurity firm that improves operational safety and reliability for industrial control networks. Perelman highlights a U.S. Federal Bureau of Investigation (FBI) and Department of Homeland Security (DHS) account of a Russian cyber campaign to infiltrate critical infrastructure in the U.S., which occurred within the last two years.

Industrial operators should heed NIST's recommendations, as compromised SCADA systems could be hijacked by hostile actors, with potentially serious outcomes.

"'Why didn't I see anything blow up?'" Perelman says. "And the answer is that the Russians are not stupid; what they probably are after is to have a grip around critical segments of the critical infrastructures, and then when they need it most, as part of an act of war, or as part of leveraging negotiations with the U.S. next time, then they will [push the red button]."

With so much at stake, it may be difficult to believe that industrial control systems are so vulnerable to attack. However, most industrial control systems were developed before the use of the Internet had become commonplace, so they were intentionally designed to be simple, and to work in a closed system that was "air-gapped," or unconnected to the outside world. Further, many SCADA systems and industrial networks typically were built using devices that were designed and manufactured without even basic security protocols or features.

As industrial companies and utilities have sought to connect their infrastructure to their corporate systems, or to enable greater interoperability or communication capabilities between devices, hackers have taken notice.

"When we take a look at some of the devices that are being used to program logic controllers or other types of data systems that are being controlled with various thin protocols, whether it's Modbus, PnP3, Hard IP, they're very thin protocols," explains Don Arnold, a security engineer with L Squared LLC, a Greenwood Village, CO-based information security consulting firm. Arnold says these devices do not have any security protocols built into them, making them easy targets for criminals. "Therefore, any kind of SCADA or industrial control system that is sending packets inbound and outbound without some sort of a protective security environment is at risk."

The lack of modern security protocols and tools, however, does not address perhaps the biggest security risk to industrial control systems: the propensity of humans to ignore common sense and inadvertently expose the network to malware through their own behavior.

"There have always been other ways to get malware to the network, the most notorious example being when malware is brought in by USB drives," says Phil Neray, vice president of Industrial Cybersecurity and Marketing at Cyber X, a Waltham, MA-based Industrial Internet of Things (IIoT) and Industrial Control System (ICS) cybersecurity platform provider. In this scenario, attackers may scatter or distribute USB thumb drives outside the facility, in the hope that an employee of the industrial plant or utility will pick one up and plug the drive into a computer within the facility, allowing malware to be automatically uploaded onto the network. This technique was used in the U.S./Israeli Stuxnet cyberattack on Iran's nuclear facilities in 2010, and likely in other instances of industrial sabotage.

"Social engineering is starting to become much more popular today, because the technical security has increased," Arnold says. "Attackers are starting to go towards the weakest link, and that's people. The majority of attacks occur from inside the network, whether they are socially engineered, or if somebody [clicks on] an email and they launch something into the network. The end user is still the absolute weakest link in the network."

For industrial or utility companies, there are several protective measures that should be taken to harden or protect SCADA systems from those with malicious intent, but the process should begin with a sober risk assessment.

"The first thing I would do is sit down and say, 'where's the keys to the kingdom and what's the value of the keys to the kingdom'," Arnold says. "What's the most important asset that you have to protect, and then design a security system around that."

Joe Morgan, business development manager for critical infrastructure for Sweden-based Axis Communications, provides a concrete list of steps that should be taken by industrial and utility organizations, including using certificate control on each IP device, and cloaking IP addresses within a network so potential hackers will not have an IP trail to follow in their attempts to take over the control operations or extract data from the network or present a false narrative to invoke an unneeded response, propagating a dangerous chain of events.

Video surveillance may be linked to analytics software to distinguish between cyber threats and other abnormal, but important, operational situations.

Further, Morgan says video surveillance or thermal cameras can be linked to analytics software to help distinguish between cyber threats and other abnormal, but still important, operational situations, such as a blown gasket on a pressure line in a manufacturing plant. The cameras and the analytics engines can be used to verify whether a SCADA alarm is tipping personnel off to an actual issue, or to a false alarm caused by a breach.

The greatest challenge, of course, is that deploying new or augmented security controls is expensive, and many organizations simply will not see the return on investment or value of taking proactive security steps.

"Somebody will get in," says Mike Trojecky, vice president of IoT and Analytics for U.K.-based IT solutions company Logicalis. "It's about identifying when it happens and being able to respond to it and basically remediate as quickly as possible. But the cost to implement the security, in a lot of cases, is greater than the immediate cost to recover from an attack. [Organizations] don't look at the long-term, corporate espionage piece, but instead will think, 'Well it's going to cost me $1 million to do [implement security], but I was attacked and compromised, and it only cost me $250,000." C

Further Reading

What is SCADA?, Inductive Automation, September 12, 2018
https://inductiveautomation.com/resources/article/what-is-scada

International Society of Automation ISA/IEC 62443
http://bit.ly/2VCmmgQ

Framework for Improving Critical Infrastructure Cybersecurity, National Institute of Standards and Technology, April 16, 2018

The Virus That Saved The World From Nuclear Iran? STUXNET, *The Infographics Show*, June 3, 2018
https://www.youtube.com/watch?v=J07N1KXOyfk

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

Milestones

Fernando, Smith Receive George Michael Memorial HPC Fellowships

Milinda Shayamal Fernando of the University of Utah, and Staci A. Smith of the University of Arizona have been named to receive 2019 ACM-IEEE CS George Michael Memorial HPC Fellowships, created to honor Ph.D. students whose research focuses on high-performance computing applications, networking, storage, large-scale data analytics using the most powerful computers available.

The award of a fellowship to Fernando is in recognition of his work on high-performance algorithms for applications in relativity, geosciences, and computational fluid dynamics.

Fernando's research focuses on developing algorithms and computational codes that enable the effective use of modern supercomputers. His key objectives include making computer simulations on high-performance computers easy to use, portable. high-performing, and scalable.

The results of Fernando's work have improved applications in areas of computational relativity and gravitational wave astronomy. In the universe, when two supermassive black holes merge, they bring along clouds of stars, gas, and dark matter; modeling these events requires powerful computational tools that consider all the physical effects of such a merger. Recently developed algorithms and codes to create simulations of black hole mergers were limited because they could only handle simulations when the masses of the two black holes were comparable. Fernando developed algorithms and code capable of modeling mergers of black holes, or neutron stars, of vastly different mass ratios. These computational simulations help scientists understand the early universe, as well as what is going on at the heart of galaxies.

Smith was named to receive a fellowship in recognition of her work developing a novel dynamic rerouting algorithm on fat-tree interconnects.

A general problem in high-performance computing is that multiple jobs running on supercomputers send messages at the same time, and these messages interfere with each other, potentially degrading a computer's performance. Smith's first research paper in this area had two goals: to explore the causes of network interference between jobs (in order to model that interference), and to develop a mitigation strategy to alleviate the interference.

As a result of this work, Smith recently developed a new routing algorithm for fat-tree interconnects called Adaptive Flow-Aware Routing (AFAR), which improves execution time up to 46% when compared to other default routing algorithms. As part of her ongoing Ph.D. research, Smith continues to develop algorithms to improve the performance and efficiency of HPC workloads.

The ACM-IEEE CS George Michael Memorial HPC Fellowships are endowed in memory of George Michael, one of the founding fathers of the SC Conference series. The Fellowships include a $5,000 honorarium and travel expenses to attend SC19 in Denver, CO, next month, where the Fellowships will be presented.

Society | DOI:10.1145/3355375

Esther Shein

The CS Teacher Shortage

How can we fill more computer science classrooms when there just aren't enough teachers to go around?

THE ONLY EXPOSURE Yancarlos Diaz had to computer science during his high school years in New York City was when he used a computer to write essays. When it came time to apply to college, Diaz, who says he was good in math, "blindly signed up" for the computer science program at the Rochester Institute of Technology (RIT), figuring it was a major that would help him easily find a job when he graduated.

That decision already is paying off.

Now a fourth-year student at RIT, Diaz expects to graduate in 2021 with dual bachelor and master of science degrees in computer science (CS). He then plans to work in the private sector as a software engineer "mainly to pay the loans,'' he says.

Diaz is not alone. Colleges are not producing large numbers of CS majors, and many of those who graduate with a CS degree are opting to go into industry rather than academia, which can pay twice as much as what professors earn. This is causing a perfect storm: a shortage of computer science teachers is making it harder for many students majoring in the discipline to get into the classes they need to graduate.

Finding enough qualified computer science teachers is also an issue in secondary education. Only 36 teachers graduated from universities with computer science degrees in 2017, compared with 11,157 math teachers and 11,905 science teachers, according to the nonprofit Code.org. In 2016, 75 teachers graduated from universities equipped to teach the subject, the organization reports.

IMAGE FROM SHUTTERSTOCK.COM

"I can say at the K–12 level there's a dramatic shortage'' of computer science teachers, says Jake Baskin, executive director of the Computer Science Teachers Association (CSTA), which worked with Code.org to produce a report in 2018 on the state of computer science education policy in the U.S. Surprisingly, the study revealed that only 35% of public high schools in 24 states offer computer science courses.

However, 33 U.S. states now offer teacher certification in computer science, up from 27 last year. "Overall, the theme of the report is very much that we're moving in the right direction and adopting policies … to increase participation in computer science in the K–12 space,'' Baskin says.

The increase is also trickling up, making computer science a far more popular college major. The average number of undergraduate computer science majors per department at U.S. doctoral institutions grew from 818 in 2016 to 900 in 2017, according to the Computing Research Association (CRA) annual Taulbee Survey. This has proven to be a mixed blessing, says Elizabeth Bizot, director of statistics and evaluation at the CRA.

"There's a lot of demand for students with those skills, and in that sense, increases are a good thing,'' she says. However, the average number of CS majors per department has increased 368% from 2006 to 2017, according to Bizot, "and that puts a lot of strain on departments in terms of teaching resources, classroom space, etc., as they seek to serve students well."

That has been the issue at The University of Texas at Austin (UT Austin), where demand for the CS major is rising and the faculty is unable to keep pace, says Donald Fussell, chairman of the university's computer science department.

"As enrollments keep growing without bounds, it's very hard for anyone to keep their faculty size growing as fast as the demand for computer science majors,'' Fussell says.

Two ways to respond to that in the short term are to "start trying to hire like crazy," Fussell says, and compromise your quality standards, or leave enrollments open and have very large programs, which translates into increased competition for courses. At UT Austin, the solution has been to put a cap on the number of computer science majors, Fussell says.

"It's not so much a shortage (of faculty), it's just when you grow that quickly, there's no quick way" to find enough qualified teachers to meet the demand for courses, he says.

Fussell says his department is hiring new faculty members on a non-tenured track "much more aggressively," which works out well, "because these folks are really good teachers. But the pay scales are such that it's hard to offer compelling salaries" at a state-run university.

A typical non-tenured position at UT Austin commands under $100,000 for a six-course, nine-month teaching load, he says. "Anyone with reasonable experience in the [computer science] industry is going to be looking at something twice that much. So that's the problem."

The problem also exists at private institutions, not only because industry pays so much better, but also because there is a shortage of CS Ph.D. candidates, and a doctorate is a requirement for tenure-track positions. RIT, which has over 4,200 students enrolled in graduate and undergraduate computing degree programs and is one of the largest colleges of computing in the country, has been able to address the problem with increased class sizes and by using current Ph.D. students to assist with teaching, says Mohan Kumar, chair of the Computer Science Department.

Having doctoral students help with the teaching is alleviating the problem of students getting shut out of computer science classes, he says. "We do have a large number of non-tenure-track faculty to do the bulk of teaching,'' Kumar says. He adds that his department also has some people who work in industry teaching part-time, "but the number is very small."

RIT recently hired four tenure-track faculty with Ph.D.'s as well, he adds.

The University of Illinois at Urbana-Champaign (UIUC) has nearly doubled its number of undergraduate computer science majors in the past seven to 10 years, according to Leonard Pitt, associate head of the Computer Science Department. To cope with an ongoing shortage of computer science professors, UIUC created the CS + X program, which blends computer science with a discipline in the arts and sciences. "That's allowed us to expand the number of programs without taxing the upper-level computer science courses and, hence, the faculty,'' says Pitt.

Right now, UIUC has about 1,100 computer science engineering majors, and another 700 in the CS + X program. "We saw skyrocketing demand for computer science offerings, so this was a 'build it and they will come' solution," Pitt says. In the CS + X program, students take the same foundational courses as computer science and engineering majors, amounting to about 30 hours of course work. The main difference is that they also take courses in disciplines such as advertising, astronomy, crop sciences—and even music.

At the same time, hiring both tenure-track and non-tenure-track computer science faculty remains a challenge, Pitt says. "We are trying to grow like everybody else and have a number of open slots, and the limitation on our growth is … the rate at which we can hire for faculty, both teaching faculty as well as tenure-track faculty."

Pitt recalls that last year, the department hired a teaching faculty professor to teach a 400-person math class. "The week before classes began, he came and said, 'I hate to do this, but I had applied for a position at Cisco and thought it was dead, but they came to me and offered me three times the salary.' So how do you compete with the incredible demand for Ph.D.'s in industry and the amount of money they're being paid?"

Not only are such candidates being offered top salaries, but if they are interested in conducting research, many companies also offer the autonomy to do that, Pitt adds. "So we struggle with that and also, there's a limited pool of top talent [among] computer science graduates with Ph.D.'s and we're competing with all of our peer schools,'' he says. "It's a sellers' market if you're a graduating Ph.D. computer science student."

These dynamics are causing some students to get shut out of classes, Pitt says. The department used to open computer sciences classes to all students until a few years ago, when "it was pretty clear we had to restrict registration to computer science students initially, until all [of them] had a chance to register."

Normally, priority is given to seniors first; the problem, he says, was that seniors from different disciplines were taking spots away from computer science majors. "We have an obligation to help them graduate in four years," and the department didn't want students not to be able to graduate because they couldn't get into a computer science class. But it continues to be "very competitive to get into one of these classes," he acknowledges.

UIUC is addressing the issue with larger class sizes and by offering classes online, in recognition of the fact that "some students prefer to stay home and watch a lecture in their pajamas,'' Pitt notes. "So by creating online resources for classes, we've been able to expand our offering."

The university has also started using some adjunct faculty, "but more often than not, they'll be someone who is maybe at the post-doctoral level who is passing through on their way somewhere and just serendipitously happens to be" in the area, he says. But that doesn't happen often, since "we're on Silicon Prairie, not Silicon Valley, so the number of professionals in town is nowhere near as large as what you'd see in the Bay area."

Pitt believes one way that universities can prepare more students to become computer science professors is to increase interest among women and underrepresented minorities. Another way is to get undergraduates to see the value of research early on.

UIUC has a program that matches freshmen and sophomores with graduate students to work in faculty research labs for credit. Pitt says it's a win-win because it gives the graduate students the ability to see what it's like to mentor undergraduates and lead research, while the undergrads are exposed to research work early.

RIT's Diaz says eventually, he would like to go into academia and teach at the high school level, even though it means he will have to take a pay cut. "I've always had passion for teaching,'' he says. "I've been a tutor since my second year here and I like it, and people say I'm good at it. And it's my way of giving back, after a few years."

Further Reading

Flaherty, C.
System Crash, *Inside Higher Ed*, May 9, 2018 https://www.insidehighered.com/news/2018/05/09/no-clear-solution-nationwide-shortage-computer-science-professors

Terdoslavich, W.
Tech Industry Really Needs Professors and Teaching Talent, *Dice*, April 30, 2018 https://insights.dice.com/2018/04/30/tech-industry-really-needs-professors-teaching-talent/

Wills, C.E.
Outcomes of Advertised Computer Science Faculty Searches for 2017, *Computing Research News*, https://cra.org/crn/2017/11/outcomes-advertised-computer-science-faculty-searches-2017/

Assessing and Responding to the Growth of Computer Science Undergraduate Enrollments, The National Academies of Sciences, Engineering, Medicine. 2017 https://cs.stanford.edu/people/eroberts/ResourcesForTheCSCapacityCrisis/files/NationalAcademiesReport-Prepublication.pdf

2017–2018 Taulbee Survey Computing Research Association https://cra.org/wp-content/uploads/2019/05/2018_Taulbee_Survey.pdf

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

Association for Computing Machinery
STEM
Author's Rights
SIGs
Science
Open
Copyright
TechNews
Publishing
Articles
eLearn
IT
Author-Izer
CHORUS
HCI
AI
OpenTOC
Ubiquity
Software
Computer Science
Scholarly
Journals
IoT
Technology
Machine Learning
Emerging Technologies
CACM
Digital Library
OpenSurround Service
Articles
Research
Transactions
Gold Open Access
Magazines
Cybersecurity
Proceedings
3D Printing
Community
Big Data
Robotics
Cloud
Networks
Access
HPC
Conferences
Computing
Green Open Access
ACM
Hybrid Open Access
Communications
XRDS
Inroads
Computers in Entertainment
Queue

DOI:10.1145/3357222

Michael A. Cusumano

Technology Strategy and Management

The Cloud as an Innovation Platform for Software Development

How cloud computing became a platform.

Since the early 2000s, advances in networks and virtualization technology have made Web delivery of software applications possible on different types of hardware and software. As a result, we have seen increasing use of Internet or "cloud-based" servers as the primary way organizations and individuals use software applications (see "Cloud Computing and SaaS as New Computing Platforms," *Communications*, April 2010). Many software developers have continued to build applications using hardware-based operating systems. Today, however, the trend is clear that cloud-based services have become a new platform not only for using software applications but also for building them.

In addition to hardware-based operating systems such as Google Android, Microsoft Windows, Apple's iOS, and Linux, now we must include Amazon Web Services (launched in 2006, initially to sell Amazon's excess computing and storage capacity), the Google Cloud and App Engine (launched in 2008), and Microsoft's Azure (launched in 2010) as popular platforms for applications development. But cloud computing as an applications development platform did not occur overnight. The evolution occurred gradually as the cloud service vendors began opening up their proprietary infrastructures to third-party software engineers who wanted to build new applications and use tools or services specific to those hosting environments.[a]

The trend is clear that cloud-based services have become a new platform not only for using software applications but also for building them.

There were also network effects associated with this gradual transition that made the new development environments increasingly popular and valuable: As more tools and services as well as third-party applications be-

[a] This column extends an earlier discussion in Michael A. Cusumano, "The Evolution of Cloud Computing into a New Type of Innovation Platform," in A. Boes and B. Langes, *Die Cloud und der digitale Umbruch in Wirtshaft und Arbeit* [*Cloud and the Digital Revolution of Work and Economy*], Haufe Group, Germany, 2019.

IMAGE BY ANDRIJ BORYS ASSOCIATES, USING SHUTTERSTOCK

came available, more application developers chose to build new software for specific Web environments, which led the platform owners and third-parties to make more tools, services, and applications available. Much like the app stores for smartphone software, Amazon, Microsoft, and Google, as well as other firms, now boast app stores for their cloud-based applications and development tools.[b]

An early movement toward using Web environments for applications development came from companies that delivered software products as a service. In particular, Salesforce.com, founded in 1999, created a customer relationship management (CRM) product configured not as packaged software but as software delivered over its own servers and accessed through a customer's browser. In 2005, Salesforce launched a website called AppExchange that functioned as a transaction platform and online store for companies to share, buy, and sell applications or tools that work particularly well with the Salesforce CRM product. Salesforce then created an innovation platform when it launched Force.com in 2008 as a development and deployment environment using Salesforce's SaaS server infrastructure and growing set of software engineering tools.

Various companies would go on to offer other public hosting services for SaaS products but Amazon quickly became the market leader for hosting as well as Web-based applications development. It had over 400,000 developers registered as early as 2008 to use Amazon Web Services (AWS).[6] AWS was particularly attractive because it was not tied to a single product and was especially good for developing online retailing applications. Amazon also made several critical services available to its cloud users, including data storage, computing infrastructure, messaging, content management, and customer billing. In recent years, AWS has expanded to include computing resources (EC2 for virtual servers and Lamda for running code); database, data storage, and content delivery services; networking administration and security services; code deployment facilities; analytics, application, and mobile support services; and a growing number of enterprise applications.[c]

Microsoft in 2019 offered a similar collection of tools and services for its

[b] See https://aws.amazon.com/marketplace/help; https://azuremarketplace.microsoft.com/en-us/marketplace/; and https://cloud.google.com/marketplace/.

[c] See https://aws.amazon.com/products/?nc2=h_ql_prod.

hosting service users and application developers. Its shift to the cloud began in 2010, when Microsoft began to offer Azure as an online service consisting mainly of Windows Live and Office Live. Azure initially appeared to be a weak competitor to Amazon and Google because of Microsoft's strong preference for the packaged software business model. That preference gradually changed as increasing numbers of Microsoft customers asked for software delivered as a service and preferred to pay via short-term licenses and annual subscriptions.[d]

Google was an early innovator in Web services but lost ground to Amazon and Microsoft probably because it lacked the experience, and the strategy, to sell services to enterprises. Most of what Google offers its users (except advertisers) it offers for free. However, this may change in the future. Google is now spending billions of dollars to upgrade its cloud infrastructure and marketing efforts, with the intention of attracting more enterprise users for its services.[1]

As we look back, though, it is Microsoft that most stands out for its successful shift in strategy under CEO Satya Nadella that led the company to repackage many of its key software products and development tools as online services.[3] For example, Azure made available all the services that had been ported to the online versions of Windows and Office as well as Microsoft SQL Server, Microsoft CRM, .NET services, and Sharepoint services. These Web offerings made it possible for customers to continue using Microsoft's products as Web services rather than buy new versions of the packaged software. At the same time, Microsoft enabled its customers to integrate Microsoft services with products from other vendors, thereby positioning Azure as a relatively neutral hosting environment and innovation platform. On Azure, software engineers can use various programming languages and not just Microsoft's proprietary .NET environment.

We can expect further growth but also intensifying vendor competition between Microsoft and Amazon, especially for enterprise users.

Microsoft's cloud revenues have continued to rise as customers built new applications using the Azure Web services and ran them on the Azure platform rather than on Windows. Microsoft lumps several cloud services together, but one estimate is that, in fiscal 2018, Microsoft had revenues of approximately $23 billion from Azure out of total company revenues of $110 billion. The Azure business in 2018 and 2019 was also growing at annual rates between 70% and 90%.[5,8] This growth has helped propel Microsoft beyond Apple and Amazon to become the most valuable publicly listed company in the world, with a market value over $1 trillion.

For the past 10 years or so, industry analysts have been tracking cloud software revenues in increasing detail, with annual reports on the different market segments. "Software as a Service" refers to software products delivered via the Web and priced generally on a subscription or on-demand basis, with the full applications stack and data running on the software company's cloud servers. "Platform as a Service" (PaaS) refers to features that users access via the cloud while managing their own software applications and data on internal (on-premises) company servers. These features include middleware applications, the operating system and updates, data storage, networking, and tools for developing new applications. PaaS covers the new cloud-based innovation platforms. There is also "Infrastructure as a Service" (IaaS), which generally refers to when customers manage their own applications, databases, middleware, and operating systems while receiving some basic services via the cloud. These other services include virtualization (the ability to run different applications in a software environment that emulates different hardware platforms), data storage, and networking.[9]

The total size of the information technology market in 2018, including software, hardware, and services, was estimated to be approximately $3 trillion.[2] Cloud revenues remained a relatively small part of this total number but were growing 20% to 30% annually: SaaS revenues in 2018 were estimated at over $55 billion; IaaS revenues approximately $45 billion; and PaaS revenues just under $11 billion. Public cloud services cut across all three categories, with AWS by far the market leader in recent years. Gartner and Goldman Sachs put AWS's mid-2019 market share at 47%, followed by Microsoft (22%), Alibaba (8%), and Google (7%).[8]

Among these companies, Microsoft's Web business was growing the fastest. We can expect further growth but also intensifying vendor competition between Microsoft and Amazon, especially for enterprise users. Both companies offer the most popular cloud platforms and marketplaces for software development as well as powerful IaaS platforms for running those applications. There is now little doubt that Microsoft—which made its fortune from packaged versions of Windows and Office—has successfully made a transition in its business model from desktop software to the cloud. C

References

1. Amin, A. Google's Market Share in the Cloud Space Has Increased. *Marketrealist.com*, February 2019.
2. Bartels, A. Global Tech Market Will Grow By 4% in 2018, Reaching $3 Trillion. *Forbes.com*, October 18, 2017.
3. Carr, A. and Bass, D. The Nadellaissance. *Bloomberg Businessweek*, May 6, 2019.
4. Coles, C. AWS vs. Azure vs. Google Cloud Market Share 2017. *Skyhighnetworks.com*, March 5, 2018.
5. Dignan, L. Microsoft Q4 Strong as Commercial Cloud Revenue Hits $6.9 Billion. *ZDNet.com*, July 19, 2018.
6. Malik, O. Amazon Cuts Prices on S3. *Gigaom.com*. October 9, 2008.
7. Novenet, J. Amazon lost cloud market share to Microsoft in the fourth quarter: Keybanc. *CNBC.com*, January 12, 2018.
8. Stalcup, K. AWS vs. Azure vs. Google Cloud Market Share 2019: What the Latest Data Shows, *ParkMyCloud.com*, April 30, 2019.
9. Watts, S. SaaS vs. PaaS vs. Iaas: What's the difference and how to choose. *BMC.com*, September 22, 2017.

Michael A. Cusumano (cusumano@mit.edu) is a professor at the MIT Sloan School of Management doing research on computing platforms for business and coauthor of *The Business of Platforms: Strategy in the Age of Digital Competition, Innovation, and Power* (Harper Business, 2019).

[d] See https://azure.microsoft.com/en-us/overview/what-is-azure/.

DOI:10.1145/3357225

Peter G. Neumann

Inside Risks

How Might We Increase System Trustworthiness?

Summarizing some of the changes that seem increasingly necessary to address known system and network deficiencies and anticipate currently unknown vulnerabilities.

IMAGE FROM SHUTTERSTOCK.COM

THE ACM RISKS Forum (risks.org) is now in its 35th year, the *Communications* Inside Risks series is in its 30th year, and the book they spawned—Computer-Related Risks[7]—went to press 25 years ago. Unfortunately, the types of problems discussed in these sources are still recurring in one form or another today, in many different application areas, with new ones continually cropping up.

This seems to be an appropriate time to revisit some of the relevant underlying history, and to reflect on how we might reduce the risks for everyone involved, in part by significantly increasing the trustworthiness of our systems and networks, and also by having a better understanding of the causes of the problems. In this context, 'trustworthy' means having some reasonably well thought-out assurance that something is worthy of being trusted to satisfy certain well-specified system requirements (such as human safety, security, reliability, robustness and resilience, ease of use and ease of system administration, and predictable behavior in the face of adversities—such as high-probability real-time performance).

The most recent Inside Risks discussion of trustworthiness appeared in the November 2018 *Communications* column.[2] This column takes a different view of the problems, with a specific conclusion that we need some fundamental changes in the state of the art and practice of developing and using computer systems, rather than trying to continually make small incremental improvements on baselines that may be unworthy.

The pithy wisdom of Albert Einstein—"Everything should be made as simple as possible—*but not simpler*"—is particularly relevant in the design, modification, and configuration of computer systems and networks. Oversimplification is often a cause of failure to satisfy expectations. Indeed, this column seriously violates that wisdom: each bullet item significantly oversimplifies the point it is intended to make. Thus, each item should be considered as a guideline that must be applied with considerable care, experience, and detailed elaboration. Consequently, given that achieving trustworthiness is inherently complex and there are typically no easy answers or quick fixes, it is with some trepidation that I offer the following ideas that might help enhance trustworthiness:

- Accept that we cannot build adequately trustworthy applications on top of compromisable hardware and flawed systems of today, particularly for those with life-critical requirements. (The Common Vulnerabilities Enumerators list—cve.mitre.org—now includes over 120,000 vulnerabilities!) For example, the best cryptography and useful artificial intelligence can be completely subverted by low-level attacks, insider misuse, and hardware failures, whereas applications are still a huge source of vulnerabilities even in the presence of stronger operating-system security. Vulnerabilities tend to be pervasive. On the other hand, building new systems or cryptography from scratch is likely to be riskful. Thus, having a set of trustworthy basic system and cryptographic (for example, EverCrypt) components would be a highly desirable starting point.
- Establish a corpus of theoretical and practical approaches for predictable composition of such components—addressing both composability (requiring the preservation of local properties) and compositionality (requiring the analysis of emergent properties of compositions, some of which are vital, as in safety and security—but some of which may be dangerous or otherwise failure-prone, as in exposed crypto keys and privacy violations). Composition itself can often introduce new vulnerabilities.
- Develop and systematically use more ways to reliably increase trustworthiness through composition. Desirable approaches might include (for example) the use of error-correcting codes, cryptography, redundancy, cross-checks, architectural minimization of what has to be trusted, strict encapsulation, and hierarchical layering that avoids adverse dependencies on less-trustworthy components.
- Whenever a technology or a component might be potentially unsound, trustworthiness (for both composition and compositionality) must be independently evaluated—for example, when using machine learning in life-critical applications.
- Adopt and honor underlying principles of computer systems, especially with respect to total-system trustworthiness for safety and security.
- Eschew the idea of inserting back doors (or not patching existing ones) in computer and communication systems.[1] It should be intuitively obvious that if back doors in systems have exploitable vulnerabilities, they would be exploited by people and programs supposedly not authorized to use them. Nevertheless, governments repeatedly fantasize that there can be bypasses that would be securely accessible only to 'authorized' entities. (Consider again the first bullet item in this column.)

We need some fundamental changes in the state of the art and practice of developing and using computer systems.

- Recognize that trustworthiness in the "Internet of Things" may always be suspect (for example, regarding security, integrity, human safety, and privacy). Various hardware-software and operational approaches must be developed (perhaps easily securable and locally maintainable firewalls?) that can help control and monitor how various classes of devices can more soundly be connected to the Internet. Self-driving vehicles, fully autonomous highways, and totally interconnected smart cities imply the components and the total systems must be significantly more trustworthy. However, the risks of ubiquitously putting your crown jewels on the IoT would seem to be excessively unwise.
- Accept the fact that the use of remote processors and storage (for example, cloud computing) will not necessarily make your computer systems more trustworthy, although there are clearly considerable cost and operational savings that can result from not having to manage local hardware and software. Nevertheless, trusting trustworthy third-party cloud providers would be more desirable than attempting to create one's own. As in other cases, there are many trade-offs to be considered.
- Accept the reality that we cannot build operational election systems that are trustworthy enough to withstand hacking of registration databases, insider misuse, rampant disinformation and disruptions—especially using components without substantive audit trails or paper records that should be able to make forensics-worthy analysis possible. Although greater system trustworthiness would be helpful, many of the existing problems are not technological and must also be addressed.
- Recast software engineering and system engineering as engineering disciplines, with more focus on hardware and software vulnerabilities, aspects of system trustworthiness, importance of well-defined system requirements, proactive design, system usability, risk assessment, computer-science theory and practice.
- Revamp software-engineering educational programs to ensure graduates have the necessary abilities and resources.[3]
- Recognize there are no one-size-fits-all solutions, and that many potential trade-offs must be considered. Furthermore, technology by itself is not enough, and many other factors must be considered—especially critical systems.
- Stress learning, not just teaching, to instill an awareness of the issues discussed here from elementary school on, dealing with complexity, principles, abstraction, respecting holistic long-term thinking rather than just short-term premature optimization, logical reasoning, altruism, and much more. Encourage rational and logical thinking from the outset, and later on, the use of practical formal methods to improve the quality of our computer systems. Formal methods have come a long way in recent years (for example, DeepSpec) and are increasingly finding their way into practice.
- Pervasively respect the importance of human issues (for example, with greater emphasis on usability, personal privacy, and people-tolerant interfaces) as well as issues that are less technological (for example, compromises of supply-chain integrity, environmental hazards, and disinformation). Also, independent oversight is often desirable, as for example is the case in aircraft safety, business accountability, and elections.
- Respect history, study the literature, learn from past mistakes, and benefit from constructive experiences of yours and others.
- Recognize this list is incomplete and only a beginning. For example, I have not even mentioned the risks of side channels, speculative execution, direct-mem-

ory access from embedded microcontrollers and input-output, and tampering.

As a reminder, some important mantras have been repeated in the Inside Risks archives. Here are just a few:

- Characterizing potential vulnerabilities inherent in various types of computer-related systems and operational environments, such as automation,[11] clouds,[9] IoT,[5] and AI.[16]
- System engineering and software engineering as a discipline.[2,15]
- Theoretically based practice.[4,14,15]
- Foresighted planning for achieving long-term benefits rather than just short-term gains.[6,8,10]

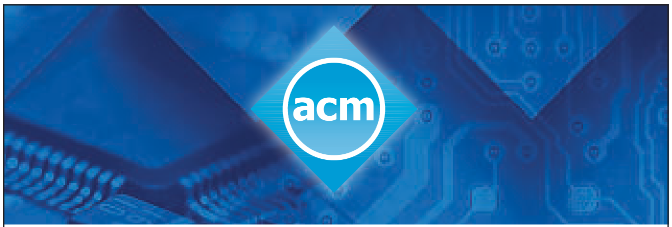
Note that old wisdom may still be very relevant and insightful, as in the Einstein quote, Norbert Wiener's prescient *Human Use of Human Beings*,[18] and Don Norman's *The Design of Everyday Things*.[13] *Computer-Related Risks*[7] is no exception. Furthermore, there is considerable hope in some recent advances. For example, the CHERI hardware-software architecture and its intra-process compartmentalization[17]—together with its ongoing formal analysis of the hardware specifications—can provide some guidance on how many of the aforementioned desiderata and principles[12] can actually be constructively applied in practice. CertiKos, seL4, and the Green Hills separation kernel are other examples of formal analysis of real system components—albeit just for operating-system microkernels.

Each bulleted item is oversimplified, and the problems that must be faced are complex and far-reaching. System engineers, academics, computer users, and others might wish to reflect on the history of how we reached where we are today, and how theoretical and practical research and development experience might help achieve the desired goals, as well as avoiding known shortcomings and as-yet-unrecognized vulnerabilities. However, the bottom line is that we still have a long way to go toward achieving trustworthy systems.

References

1. Abelson, H. et al. Keys Under Doormats: Mandating Insecurity by Requiring Government Access to All Data and Communications. July 6, 2015. https://dspace.mit.edu/handle/1721.1/97690
2. Bellovin, S.M. and Neumann, P.G. The big picture. *Commun. ACM 61*, 11 (Nov. 2018).
3. Landwehr, C.J. et al. Software systems engineering programmes: A capability approach. In *Journal of Systems and Software 125* (Mar. 2017), 354–364; Article: JSS9898 doi 10.1016/j.jss.2016.12.016
4. Leveson, N. and Young, W. An integrated approach to safety and security based on system theory. *Commun. ACM 57*, 2 (Feb. 2014).
5. Lindqvist, U. and Neumann, P.G. Risks in the emerging Internet of Things. *Commun. ACM 55*, 2 (Feb. 2017).
6. Neumann, P.G. The foresight saga, redux. *Commun. ACM 55*, 10 (Oct. 2012).
7. Neumann, P.G. *Computer-Related Risks.* Addison-Wesley and ACM Press, 1995.
8. Neumann, P.G. More sight on foresight. *Commun. ACM 56*, 2 (Feb. 2013).
9. Neumann, P.G. Risks and myths of cloud computing and cloud storage. *Commun. ACM 57*, 10 (Oct. 2014).
10. Neumann, P.G. Far-sighted planning for deleterious computer-related events. *Commun. ACM 58*, 2 (Feb. 2015).
11. Neumann, P.G. Risks of automation. *Commun. ACM 59*, 10 (Oct. 2016).
12. Neumann, P.G. Fundamental trustworthiness principles in CHERI. A. Shrobe, D. Shrier, and A. Pentland, Eds. In *New Solutions for Cybersecurity*, MIT Press/Connection Science, 2018, chapter 6.
13. Norman, D. *The Design of Everyday Things*, 2002; revised and expanded edition, 2013.
14. Parnas, D.L. Software engineering: An unconsummated marriage. *Commun. ACM 40*, 9 (Sept. 1997).
15. Parnas, D.L. Risks of undisciplined development. *Commun. ACM 53*, 10 (Oct. 2010).
16. Parnas, D.L. The real risks of artificial intelligence. *Commun. ACM 60*, 10 (Oct. 2017).
17. Watson, R.N.M. Capability Hardware Enhanced RISC Instructions: CHERI Instruction-Set Architecture, Version 7, University of Cambridge, June 2019; https://www.cl.cam.ac.uk/research/security/ctsrd/cheri/
18. Wiener, N. *The Human Use of Human Beings.* Houghton Mifflin, 1950, revised 1954.

Peter G. Neumann (neumann@csl.sri.com) is Chief Scientist of the SRI International Computer Science Lab, and moderator of the ACM Risks Forum.



DOI:10.1145/3357227

George V. Neville-Neil

Article development led by acmqueue
queue.acm.org

Kode Vicious
What Is a Chief Security Officer Good For?

Security requires more than an off-the-shelf solution.

IMAGE BY LIGHTSPRING

Dear KV,
The little startup I am working for must be getting bigger because we just hired someone to be our "chief security officer," which I place in quotes because I am not quite sure what that actually means. Most of the developers I work with seem to write good code, which, if I understand some of your previous columns, means we should also have relatively good security.

What confuses me about the CSO is that whenever our chief architect—my boss—tries to talk to him about how our systems function, I get the feeling the CSO is not listening. In fact, much of what the CSO has done since joining our company has not focused on the security of our software. Instead, he buys third-party security products and then pushes them on the development groups and the rest of the company. Often these systems get in the way of getting work done, and from time to time they just fail, which means we either stop using them or find ways to bypass them.

Is this normal? I like working at startups, and this is the first time I have been at one that has become big enough to hire such a person, so maybe this is just how big companies work and it is time to move to yet another startup, where security is part of our work rather than something that is bought for us.

Bought and Paid For

Dear Bought,
Asking "What is a CSO good for?" is like asking "What is any executive good for?" This is a topic that is probably too meaty for me to address in a single column, but let's see if I can at least partially answer your question here. CSOs are like snowflakes; no two are alike. Actually, the snowflake theory of any group is completely incorrect; there are definitely distinct categories you find in any role, whether it is a developer, marketer, or C-level executive. Like any executive, a CSO is supposed to be a leader with a concentration in security, someone who can: survey and understand the threats against the company on many levels; describe those threats to various groups within the organization; and then develop plans to protect the company, its people, and its assets against those threats.

The CSO is *not* a security engineer, so let's contrast the two jobs to create a picture of what we should and should not see.

The CSO thinks about (actually, the good ones have nightmares about) various security threats and then ranks them in various orders. One possible ordering is based on the likelihood of the threat being realistically carried out. Another ordering is based on the downside risk of the threat actually coming to fruition. A good example is an attack on a single system versus one that takes out a whole set of systems.

Imagine you are building an app that runs on someone's phone—a very common job. There is some nonzero probability that someone will attack the app. The downside risks of a successful attack on a single instance of the app (say, where the attacker can get at some data but must have physical possession of the person's phone) versus the one where the attacker can remotely get data from many—or all—instances of the app are very different. In the former case, you have failed one customer, and in the latter, you have failed your entire user base. These mental calculations, writ large, are what a CSO spends time thinking about.

A security engineer, on the other hand, builds systems such as software, network architectures, or other artifacts that implement a particular security feature against an identified threat. Using the same threat-model map, a security engineer works to prevent a successful attack on the system.

The case of the phone application remains illustrative. A security engineer will work on the application code to ensure it stores any data that must remain secret—for example, keys used to carry out secure network communications—in a secure place such as a TPM (Trusted Platform Module), a hardware security module that is commonly provided in modern, mobile hardware. Of course, the security engineer knows why this is necessary, but is not going to simultaneously worry about how the company's network routers are protected from attack.

Good security is not a one-size-fits-all situation.

Once CSOs have developed a threat-model map, they must determine if it is correct and applies to the systems being developed. Good security is not a one-size-fits-all situation. The fact that you think your CSO is not listening to your chief architect should give you pause. I would expect their discussions would be quite intense, and I have worked at one startup where no such conversation was carried out without a lot of yelling. If CSOs do not understand what they are trying to help protect, how can they protect it?

This brings me to one of the least-understood parts of security work, both by its practitioners and by those upon whom it is practiced. The security role is always a helping role: that person, or, more often, group of people, must be there to help everyone around them understand the threats and be able to point them to resources that help them solve their problems.

Too much of the security industry is full of people with military backgrounds or military frames of mind, where one can command and compel people to act in certain ways under harsh penalties. Most software companies are not military units, and most engineers laugh at this type of command and control. You pointed out that you and your colleagues have started to work against the security systems being foisted upon you, and this is actually the worst possible outcome, because it makes systems far less secure than if the security system was not put in place at all.

The other issue you described, the CSO's penchant for buying systems of sometimes dubious quality, has worsened with the spread of the Internet and the need to secure increasing numbers of systems. Before the Internet, you had to secure only your computer, the hulking thing in the basement, and a few dialup modems against insiders, which was bad enough. Now, your systems and software can be attacked from anywhere and everywhere, and if you look at your SSH (Secure Shell) logs, you will see they are.

As any industry grows, it inevitably draws a percentage of people and companies who are there "just to make a buck," and that makes careful and deliberate decision making even more important. There is plenty of fear, uncertainty, and doubt sown by the security industry, which you can see in their advertising in pretty much any airport: *Spammers are out to get you and there are two viruses in every laptop!* There is definitely a nasty threat landscape, and though there continues to be interesting work in mitigations, countermeasures, and overall development practices, security will remain an arms race, at least for the foreseeable future.

What your CSO is currently practicing is called "checkbook security," a particularly dangerous way to deal with threats. While there are definitely good security products on the market, the fact is that without a careful plan and careful deliberation, you cannot simply achieve security by buying a product or a suite of products. You must think about how to use the product, if it addresses an identified threat, and if it integrates with your company's work. A failing in any of these three areas means you are sending good money down a drain.

KV

Related articles on queue.acm.org

Pointless PKI
A koder with attitude, KV answers your questions. Miss Manners he ain't.
https://queue.acm.org/detail.cfm?id=1147526

Browser Security: Appearances Can Be Deceiving
A discussion with Jeremiah Grossman, Ben Livshits, Rebecca Bace, and George Neville-Neil
https://queue.acm.org/detail.cfm?id=2399757

CTO Roundtable: Malware Defense
The battle is bigger than most of us realize.
https://queue.acm.org/detail.cfm?id=1731902

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

DOI:10.1145/3357159

Ryen W. White et al.

Viewpoint
Multi-Device Digital Assistance

Increased availability of cloud services and ownership of multiple digital devices create unique opportunities for digital assistants to provide guidance across a range of tasks and scenarios.

THE USE OF multiple digital devices to support people's daily activities has long been discussed.[11] The majority of U.S. residents own multiple electronic devices, such as smartphones, smart wearable devices, tablets, and desktop, or laptop computers. Multi-device experiences (MDXs) spanning multiple devices simultaneously are viable for many individuals. Each device has unique strengths in aspects such as display, compute, portability, sensing, communications, and input. Despite the potential to utilize the portfolio of devices at their disposal, people typically use just one device per task; meaning they may need to make compromises in the tasks they attempt or may underperform at the task at hand. It also means the support that digital assistants such as Amazon Alexa, Google Assistant, or Microsoft Cortana can offer is limited to what is possible on the current device. The rise of cloud services, coupled with increased ownership of multiple devices, creates opportunities for digital assistants to provide improved task completion guidance.

PHOTO BY ALICIA KUBISTA/ANDRIJ BORYS ASSOCIATES

Case for Multi-Device Digital Assistance

Arguments in favor of multi-device support are not new. Cross-device experiences (CDXs) and MDXs have been discussed in the literature on interaction design, human factors, and pervasive and ubiquitous computing.[2,8] CDXs have focused on scenarios such as commanding (remote control), casting (displaying content from one device on another device), and task continuation (pausing and resuming tasks over time). In CDXs, devices are often used sequentially (that is, device A *then* device B) and are chosen based on their suitability and availability. Tools such as the Alexa Presentation Language (APL) enable developers to create experiences for different device types (that is, A *or* B). In MDXs, different devices are used simultaneously for task completion (that is, A *and* B). Digital assistants can capitalize on the complementary input and output capabilities of multiple devices for new "better together" experiences.

Although the majority of sold smart speakers lack screens, their capabilities continue to expand, for example, through the introduction of "smart displays" that combine smart speakers and screens in a single device. Despite the value of screens housed on speaker hardware, other devices (such as tablets, smartphones) often have higher-resolution displays, have more powerful processors, and are interchangeable, making them more versatile across tasks and scenarios. These devices may also already be used for the current task, providing valuable contextual information. For example, a recent study of tablet use found that 40% of participants reviewed recipes on their tablet before cooking.[6] This contextual information also helps to address challenges such as deciding when to trigger recommendations about invoking MDXs or ground question-answering in a specific context (for example, a user inquiring "how many cups of sugar?" while on a recipe website). Integrating with existing usage and information access practices also means users need not learn a new device or system to capitalize on multi-device support.

MDXs primarily address capability shortfalls in what individual devices can do, rather than there being anything inherent in the type of tasks that requires physical separation of devices. While a tablet device with a sufficiently powerful speaker and microphone would be perfectly capable of serving the exact same purpose as a speaker plus tablet with just one device, such devices could still be a long way off and require that people purchase a new device. A significant advantage of MDXs is that people can get support now, by pulling together devices they already own. Even new devices will have weaknesses that could well be addressed in combination with existing hardware.

CDX scenarios, such as reviewing background material while engaging with another device illustrate the benefits of multi-device support. Microsoft SmartGlass, a companion application for Xbox that supplements the core gaming console experience, acts as a remote control, and provides additional functionality such as recommendations, profile access, and leaderboards. Companion applications for Netflix (netflix.com) and Hulu (hulu.com) are similar. Conversely, in MDXs, all devices are integral, and contribute in important and complementary ways to the user experience and to task completion.

Figure 1a. Illustrative schematic of the *Ask Chef* MDX, combining tablet and smart speaker, mediated by cloud services, to help with recipe preparation.

In ①, the Web page URL is passed to the service (by an instrumented Website in this case) where the microdata (JSON) is parsed to extract ingredients and preparation steps. The preparation instructions are chunked for audio output and an enhanced version of the Website is displayed, with the steps enumerated and the current step highlighted. In ②, the user requests the next step via voice. The system responds in ③ with two actions *simultaneously*: (a) highlighting the next step on the tablet, and (b) vocalizing the step through the speaker.

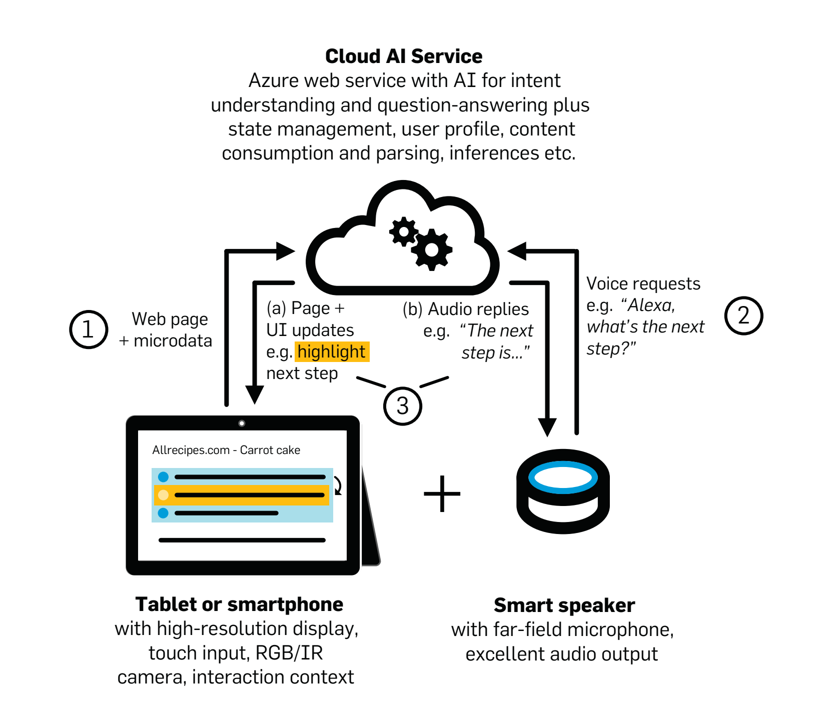
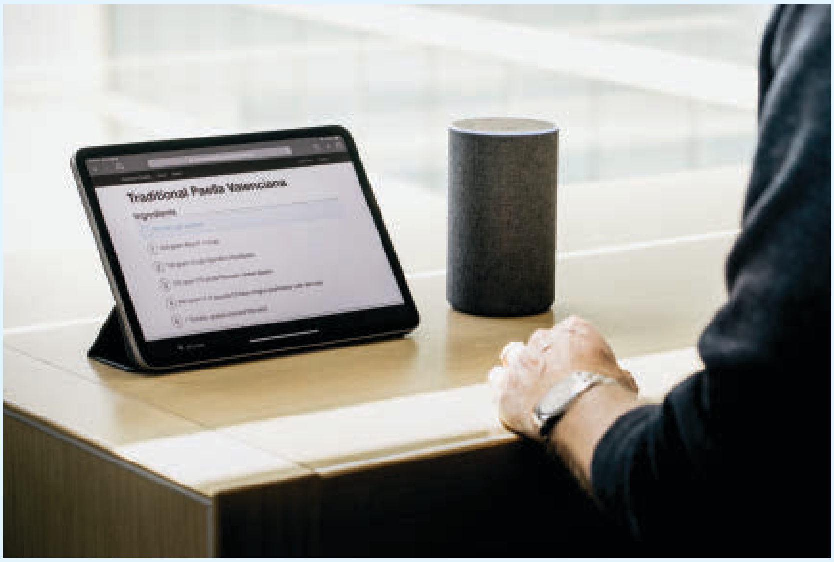

Cloud AI Service
Azure web service with AI for intent understanding and question-answering plus state management, user profile, content consumption and parsing, inferences etc.
1
Web page + microdata
(a) Page + UI updates e.g. highlight next step
(b) Audio replies e.g. "The next step is..."
Voice requests e.g. "Alexa, what's the next step?"
2
3
Allrecipes.com - Carrot cake
+
Tablet or smartphone
with high-resolution display, touch input, RGB/IR camera, interaction context
Smart speaker
with far-field microphone, excellent audio output

Figure 1b. Still image of the *Ask Chef* MDX, using an Apple iPad and an Amazon Echo smart speaker. Interactions are coordinated via a Website communicating with the cloud as in Figure 1a.



Potential for Multi-Device Digital Assistance

Usage of digital assistants on multiple devices can be low, primarily due to a lack of compelling MDXs. Support for multiple devices in digital assistants is typically limited to CDXs, mostly to assist with task continuation, including quickly resuming edits on a tablet for a document started on a desktop computer (for example, the "Pick Up Where I Left Off" feature in Cortana in Windows 10) or system-initiated pointers from smart speakers to screen devices to view lists or tables.

More than 60 million smart speakers have been sold in the U.S. alone and co-ownership of screen devices by smart speaker users is high. The benefits of building digital-assistant experiences combining screen and non-screen devices are bidirectional: screens offer visual output to augment audio plus (often) support for touch/gestural interactions, smart speakers offer hands-free capabilities through far-field microphones plus high-quality audio output. Smart speakers can also serve as the hub for Internet-of-Things (IoT) smart home devices (for example, Amazon Echo Plus has a Zigbee[a] radio to support device connectivity) and are often situated where they can easily be accessed and are proximal to task hubs (for example, in the kitchen for cooking support or in the family room to control entertainment devices). Many of these IoT devices already use conversational interfaces tailored to the device. Although engaging with devices might have once required immediate physical proximity, conversational interfaces and improved technical capabilities in areas such as far-field speech recognition have alleviated this need. A successful multi-device digital assistant would manage and leverage multi-modal input/output and conversational interfaces to create seamless simultaneous interactions with multiple smart devices, irrespective of the device manufacturer.

Digital assistants can capitalize on the complementary input and output capabilities of multiple devices for new "better together" experiences.

Guided Task Completion

At Microsoft, we have developed an MDX application called *Ask Chef* as part of a larger effort to build an extensible MDX platform for any application or website. *Ask Chef* focuses on using screen and non-screen devices for recipe preparation assistance. Cooking is a common, time-consuming task that requires state and ingredient tracking, and involves multiple steps. People routinely bring smartphones or tablets into their kitchens to help manage these processes and to access step-by-step instructions via Web browsers. Here, there is often a need for hands-free, far-field interaction.[4] Likewise, smart speaker devices such as the Amazon Echo or Google Home are frequently placed in the kitchen and are used at mealtime to set timers or manage short cooking-related processes.[b] There is an opportunity for digital assistants to help people prepare recipes more effectively by providing support including coaching,[6] status tracking, and coordinating multiple courses of a meal.

Figure 1 illustrates multi-device digital assistance in *Ask Chef*, spanning two devices: a tablet (for example, Microsoft Surface, Apple iPad) and a smart speaker (such as Amazon Echo, Google Home), and mediated by a cloud service, which orchestrates the experience to establish and maintain session state, apply artificial intelligence (for example, for intent understanding and contextual question answering), and handle events and interface updates across different devices.

Powering MDXs via cloud services means system designers do not need to rely on over-the-air updates to client-

[a] See https://bit.ly/1KfBrJS

[b] See https://bit.ly/2KZdA5q

side code to make experience and/or algorithmic modifications, that usage can be easily aggregated and analyzed to improve assistance offered, and that assistants can run on third-party hardware, enabling scaling via Amazon[c] and Google[d] skills kits. Skill availability is only part of the challenge for digital assistants; *discoverability* of skills remains an issue on devices that lack displays.[12] Screen devices can surface skill recommendations and support recall of prior skills on headless devices.

The current implementation of *Ask Chef* relies on schema.org microdata for the Web page being accessed. This markup is used to extract ingredients and preparation instructions. Important extensions include generalizing to content that is not enriched with such data and integrating additional content to augment the recipe (for example, refinements from user comments[3]). Recommending assistance for the current step in the task (including instructional content: videos, Web pages, and so forth), while also considering the previous steps, assistance already offered, and future steps. Determining how to utilize wait times between steps in recipe preparation (for example, "bake for 20 minutes") can be challenging, and users may elect to spend that time in different ways (from food preparation to unrelated tasks [such as email, social media, and other activities]).[6] Beyond task guidance, digital assistants could also provide pre-preparation support (for example, add items to a shared grocery list) and post-preparation support (for example, help revisit favorite recipes).

Looking Ahead

MDXs enable many new scenarios. Support for guided task completion can extend beyond cooking to include procedural tasks such as home improvement, shopping, and travel planning. Other scenarios such as homework assistance, games, puzzles, or calendar management could be enhanced via MDXs. Support for these scenarios could be authored by third parties. For example, educators could compose or flag visual/multimedia content to accompany tutorial/quiz materials, to help students learn more effectively than with text or audio only.[1]

A significant advantage of MDXs is that people can get support now, by pulling together devices they already own.

As experience with devices such as the Amazon Echo Show has demonstrated, augmenting voice-based digital assistants with a screen can also enable new scenarios (for example, "drop ins"—impromptu video calls). This adds value even though the screen is small; more would be possible with a larger, higher-resolution display that could be located as far from the smart speaker as needed. The user-facing camera (webcam, infrared camera) on many laptops and tablets can add vision-based skills such as emotion detection and face recognition to smart speakers. Powerful processors in tablets and laptops enable on-device computation to help address privacy concerns associated with handling sensitive image and video data.

Multi-device digital assistance is not limited to a single, static device pairing. For example, it includes scenarios such as dynamically connecting a smartphone and any one of many low-cost smart speakers as users move around a physical space; imbuing, say, any Amazon Echo Dot with the capabilities of an Echo Show. Although we targeted MDXs comprising two devices, there are situations where three or more could be used (for example, adding a smartphone to *Ask Chef* for timer tracking and alerting); these experiences must be carefully designed to avoid overwhelming users. Multi-device interactions can also help correct errors in speech recognition and yield useful data to improve voice interfaces.[9]

In sum, MDXs unlock a broad range of more sophisticated digital assistant scenarios than are possible with a single device or via CDXs. Utilizing complementary devices *simultaneously* could lead to more efficient task completion on current tasks, cost savings for device consumers, and unlock new classes of digital assistant skills to help people better perform a broader range of activities.

[c] See https://amzn.to/2cDSN3K

[d] See https://bit.ly/2NC7VnF

References

1. Carney, R.N. and Levin, J.R. Pictorial illustrations still improve students' learning from text. *Educational Psychology Review 14*, 1 (Jan. 2002), 5–26.
2. Dong, T., Churchill, E.F., and Nichols, J. Understanding the challenges of designing and developing multi-device experiences. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems* (2016), 62–72.
3. Druck, G. and Pang, B. Spice it up? Mining refinements to online instructions from user generated content. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, (2012), 545–553.
4. Jokela, T., Ojala, J. and Olsson, T. A diary study on combining multiple information devices in everyday activities and tasks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, (2015), 3903–3912.
5. Kiddon, C. et al. Mise en Place: Unsupervised interpretation of instructional recipes. In *Proceedings of Empirical Methods on Natural Language Processing*, (2015), 982–992.
6. Müller, H., Gove, J., and Webb, J. Understanding tablet use: A multi-method exploration. In *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services* (2012), 1–10.
7. Pardal J.P. and Mamede N.J. Starting to cook a coaching dialogue system in the Olympus framework. In *Proceedings of the Paralinguistic Information and Its Integration in Spoken Dialogue Systems Workshop* (2011).
8. Segerståhl, K. Crossmedia systems constructed around human activities: A field study and implications for design. In *Proceedings of the IFIP Conference on Human-Computer Interaction* (2009), 354–367.
9. Springer, A. and Cramer, H. Play PRBLMS: Identifying and correcting less accessible content in voice interfaces. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems* (2018), 296–305.
10. Sørensen, H. et al. The 4C framework: Principles of interaction in digital ecosystems. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, (2014), 87–97.
11. Weiser, M. The computer for the 21st century. *Scientific American* Special Issue on Communications, Computers and Networks, (1991), 94–104.
12. White, R.W. Skill discovery in virtual assistants. *Commun. ACM 61*, 11 (Nov. 2018), 106–113.

Ryen W. White (ryenw@microsoft.com) is Partner Research Manager at Microsoft Research AI, Redmond, WA, USA.

Adam Fourney (adamfo@microsoft.com) is Senior Researcher at Microsoft Research AI, Redmond, WA, USA.

Allen Herring (allenh@microsoft.com) is Principal Research Engineer at Microsoft Research AI, Redmond, WA, USA.

Paul N. Bennett (pauben@microsoft.com) is Senior Principal Research Manager at Microsoft Research AI, Redmond, WA, USA.

Nirupama Chandrasekaran (niruc@microsoft.com) is Principal Research Engineer at Microsoft Research AI, Redmond, WA, USA.

Robert Sim (rsim@microsoft.com) is Principal Applied Science Manager at Microsoft Research AI, Redmond, WA, USA.

Elnaz Nouri (elnouri@microsoft.com) is Senior Applied Scientist at Microsoft Research AI, Redmond, WA, USA.

Mark J. Encarnación (markenc@microsoft.com) is Principal Development Manager at Microsoft Research AI, Redmond, WA, USA.

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

acm
Association for Computing Machinery

Advancing Computing as a Science & Profession

DOI:10.1145/3343042

Article development led by acmqueue
queue.acm.org

Step into the world behind the kernel.

BY JESSE FRAZELLE

Open Source Firmware

OPERATING SYSTEMS SUCH as Windows, Linux, and macOS have kernels. The kernel controls access to system resources. It contains the logic for allowing multiple processes to share hardware mechanisms such as CPU, memory, disk I/O, and networking.

When a computer boots, the main interface for initializing the DRAM, silicon, and devices is the firmware. The firmware initializes the operating system with a bootloader. You might have heard of GRUB (derived from Grand Unified Bootloader), a common bootloader for Linux distros.

Every computer or server typically comes with firmware produced by the vendor that manufactured it. Firmware lives in the SSD/HD (solid state drive/hard drive), keyboard, mouse, CPU, network card, and other devices.

Exploits in firmware can cause a lot of harm because of the many privileged operations for which firmware is responsible. For example, consider the hack on SoftLayer,[3] a bare-metal cloud, where the base management controller (BMC) was hacked to leave a backdoor so when a server was reprovisioned after a customer used it, the hacker could still have access to that server. The minimum bar for any cloud provider is to provide a machine for a user that gets wiped cleanly and completely after use. This is a clear violation of that promise.

Making matters worse, most firmware is proprietary. The code that runs with the most privilege has the least visibility. This leads to breaches and incidents that have the capacity to affect users on multiple platforms simultaneously. To hackers this is like catnip.

Open source firmware can help bring computing to a more secure place by making the actions of firmware more visible and less likely to do harm. The goal of this article is to make readers feel empowered to demand more from vendors who can help drive this change.

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 open source firmware.

Privilege Levels

Computers today have various levels of privileges.

- Ring 3—*Userspace.* This ring has the fewest privileges. This is where user programs run. Userspace sandboxes can restrict privileges further.
- Ring 0—*Kernel.* This is the operating-system kernel; open source operating systems allow visibility into the code behind the kernel.
- Ring −1—*Hypervisor.* This VMM (virtual machine monitor) creates and runs virtual machines. Open source hypervisors such as Xen, KVM, bhyve, among others, provide visibility into the code behind this ring.

IMAGE BY PHILIPP TUR

- Ring −2—System management mode (SMM), unified extensible firmware interface (UEFI) kernel. This is proprietary code that controls all CPU resources (more on this later).
- Ring −3—*Management engine.* This is proprietary code that runs as long as the motherboard is receiving power, even if it is off (more on this later).

This summary makes clear that rings −1 to 3 have the option to use open source software and have a large amount of visibility and control over the software. The privilege levels under ring -1 allow less control, but the situation is improving with the open source firmware community and projects.

It's counterintuitive that the code with the least visibility has the most privilege. This is what open source firmware is aiming to fix. The ecosystem's goals are focused on making firmware less capable of doing harm and making its actions more visible.

Ring −2. SMM, UEFI kernel. This ring controls all CPU resources. SMM is invisible to the rest of the stack on top of it. It was originally used for power management and system hardware control. It handles system events such as memory or chipset errors.

UEFI is the interface between the operating system and the BIOS firmware. EFI, the predecessor of UEFI, was made to solve BIOS bit and address limitations. Since then, more functionality has been added to the UEFI spec, including cryptography, networking, and authentication. The UEFI kernel is extremely complex and has millions of lines of code. It consists of boot services and runtime services. The specification (https://uefi.org/specifications) is quite verbose if you want to dig in. UEFI applications such as the UEFI shell, GRUB, Gummiboot, or Windows Boot Manager have the option of being active after boot.

The UEFI kernel is a common vector for many vulnerabilities since it has some of the same proprietary code used on many different platforms. Bootloaders such as GRUB and Windows Boot Manager are platform specific. The UEFI kernel is shared on multiple platforms, making it a great target for attackers.

Additionally, since only UEFI can rewrite itself, exploits can be made persistent. This is because UEFI lives in the processor's firmware, typically stored in the Serial Peripheral Interface (SPI) flash. Even if a user were to wipe the entire operating system or install a new hard drive, an attack would persist in the SPI flash.

Ring −3. Management engine. In the case of Intel (x86), Ring −3 is the Intel Management Engine.[7] It can turn on nodes and reimage disks invisibly. It has a kernel that runs Minix,[11] as well as a web server and entire networking stack. Because of this, Minix is the world's most widely used operating system. There is a lot of functionality in the Management Engine; it could take all day to list it all, but many resources are available for digging into more detail.[16]

Between Ring −2 and Ring −3 there are at least two and a half other kernels in our stack that have many capabilities. Each of these kernels has its own networking stacks and web servers, which is unnecessary and potentially dangerous, especially if you do not want these rings reaching out over the network to update themselves. The code can also modify itself and persist across power cycles and reinstalls. There is very little visibility into what the code in these rings is actually doing, which is horrifying, considering these rings have the most privileges.

They all have exploits. It should be of no surprise to anyone that Rings −2 and −3 have their fair share of vulnerabilities. Exploits here have a huge impact radius when they happen. For example, there was a bug in the Web server of the Intel Management Engine.[14] No one realized the bug existed for *seven* years.

How can we make it better?

Firmware Projects

Firmware projects are typically stored in SPI flash.

u-boot (https://www.chromium.org/developers/u-boot) and **coreboot** (https://www.coreboot.org/) are open source firmware. They handle silicon and DRAM initialization. Google Chromebooks use both: coreboot on x86 and u-boot for the rest. This is one part of how Google verifies boot.[2] Verified boot reduces the risk of malware, permits safe software updates, and ensures the integrity of the software on the device.

Coreboot's design philosophy is to "do the bare minimum necessary to ensure hardware is usable and then pass control to a different program called the payload" (https://doc.coreboot.org). The payload in this case is LinuxBoot.

LinuxBoot (https://www.linuxboot.org/), formerly known as Non-extensible Reduced Firmware, or NERF (https://trmm.net/NERF), handles device drivers, manages the network stack, and supplies a multiuser, multitasking environment. It is built with Linux so a single kernel can work for several boards. It is arguably better to use an open source kernel with lots of eyes on it, rather than the two and a half other kernels that are all different and closed off. This means that you are lessening the attack surface by using fewer variations of code, and you are making an effort to rely on code that is open source. Linux improves boot reliability by replacing minimally tested firmware drivers with hardened Linux drivers. (Linux is significantly more vetted than most proprietary systems are; it has lots of eyes on it, since it is used quite extensively.)

By using a kernel that already has tooling, firmware devs can build using tools they already know. When they need to write logic for signature verification, disk decryption, and the like, they can use a language that is modern, easily auditable, maintainable, and readable.

Runtimes

Runtimes enable systems to use open source firmware and run custom programming logic.

Heads (http://osresearch.net/) is a configuration of coreboot that has a securely configured Linux kernel as the coreboot payload. It works on servers and laptops. The project, started by Trammel Hudson, is influenced by several years of firmware vulnerability research (Thunderstrike; https://trmm.net/Thunderstrike; and Thunderstrike 2; https://trmm.net/Thunderstrike_2).

u-root (https://github.com/u-root/u-root) is a set of Golang userspace tools and bootloader. It is used as the initramfs for the Linux kernel from LinuxBoot.

By being open source, this new firmware stack helps improve the visibility into many of the components that were previously very proprietary. Using LinuxBoot makes boot times 20 times faster.[12] Booting an open compute node to a Linux shell went from 8 minutes to 17 seconds, a speed improvement of 32 times.

What About All the Other Firmware?

Open source firmware is needed for a plethora of other devices, too. These include the following:

- EC (embedded controller)/SIO (super I/O). This is for mobile devices and desk-based platforms. It controls keyboards, temperature monitoring, etc.
- TPM (trusted platform module). This is a secure home for cryptographic keys.
- BMC (baseboard management controller)/ME (management engine). A BMC is associated with server platforms while an ME is typically associated with client platforms. For an open source BMC, there are two projects: OpenBMC (https://github.com/openbmc/openbmc) and u-bmc (https://github.com/u-root/u-bmc). me_cleaner (https://github.com/corna/me_cleaner) is the project used to clean the Intel Management Engine to the smallest necessary capabilities.
- NIC (network interface controller). Work is being done in the open compute project on NIC 3.0,[13] a spec for a NIC.
- GPU (graphics processing unit).
- HDD/SSD.
- eMMC (embedded MultiMediaCard (eMMC)/UFS (universal flash storage). Storage devices for mobile systems.
- Power supply.
- CPLDs (complex programmable logic devices), FPGAs (field-programmable gate arrays). The programmable logic components.
- Fans.

Open source firmware is necessary not only to provide visibility into the stack, but also to verify the state of software on a machine.

Intel's Boot Guard

Boot Guard is supposed to verify the firmware signatures for the processor. The problem with this, in the case of Intel processors, is that only Intel has the keys for signing firmware packages. This makes it impossible to use coreboot and LinuxBoot or their equivalents as firmware on those processors. If you tried, the firmware would not be signed with Intel's key, and the failed attempt to boot would brick the board.

A post by Matthew Garrett about Boot Guard highlights the importance of user freedom when it comes to firmware.[1] The owner of the hardware has a right to own the firmware as well. Boot Guard prevents this. In the security keynote at the 2018 Open Source Firmware Conference,[5] Trammel Hudson described how he found a vulnerability to bypass Boot Guard (http://bit.ly/2S6oGrd); the bugzilla details can be found at http://bit.ly/2XVdAKU. The bug allows an attacker to use unsigned firmware and boot normally, completely negating the purpose of Boot Guard.

Open source firmware can help bring computing to a more secure place by making the actions of firmware more visible and less likely to do harm.

Root of trust. The goal of the root of trust should be to verify that the software installed in every component of the hardware is the software that was intended. This way you can know without a doubt and verify if hardware has been hacked. Since you have little to no visibility into the code running in a lot of places in your hardware, it is currently difficult to do this. How do you really know the firmware in a component is not vulnerable or that it doesn't have any backdoors? You cannot know without a firm root of trust.

Every cloud and vendor seem to have its own way of implementing a root of trust. Microsoft has Cerberus,[15] Google has Titan,[18] and Amazon has Nitro.[4]

Paul McMillan and Matt King gave a presentation in 2018 on securing hardware at scale.[8] It covers in great detail how to secure bare metal, while also giving customers access to the bare metal. When customers return hardware to them, they need to ensure with consistency and reliability that nothing from the customer is hiding in any component of the hardware.

All clouds must ensure the hardware they are running has not been compromised after a customer has used compute resources.

Platform firmware resiliency. Hip vendors are investing in platform firmware resiliency (PFR) based on NIST guidelines.[17] These guidelines focus on ensuring the firmware remains in a state of integrity, detecting when it has been corrupted, and recovering the pieces of firmware back to a state of integrity.

Vendors have been building features around the NIST guidelines for PFR. Intel[6] and Lattice semiconductors[10] each have a version. The Open Compute Project (OCP) talk on Intel's firmware innovations[9] states that Intel is using PFR to deliver Microsoft's Cerberus' attestation principles.

Challenges

One challenge of open source firmware involves the threat model. Whether you have a root of trust, and how that root of trust operates, depends on the threat model. Let's dive in a bit with an example. If you are an enterprise with your own cloud, your threat model would prevent you from using any firmware that might contain vulnerabilities or backdoors that would threaten your business or customer data. In this case, you would ideally want an entirely open source root of trust, as well as open source firmware for each of the devices in your server or laptop, with reproducible builds to ensure integrity. This would give you the most visibility into the firmware that is running and the logic it is composed of.

Another challenge is writing the firmware for all the devices. There are a lot of device options for vendors to use in their systems, so supporting many of those will be difficult without the device vendors helping out. For example, consider that many different vendors manufacture DRAM or SSDs.

How to help. The goal of this article is to provide some insight into what is being built with open source firmware and why making firmware open source is so important. To help with this effort, please help spread the word. Try to use platforms that value open source firmware components. Chromebooks are a great example of this, as are Purism (https://puri.sm/) computers. Ask your providers what they are doing to further the cause of open source firmware or ensuring hardware security with roots of trust.

Acknowledgments. Huge thanks to the open source firmware community and a shout out to Ron Minnich, Trammel Hudson, Chris Koch, Rick Altherr, and Zaolin for helping me along this journey.

All clouds must ensure the hardware they are running has not been compromised after a customer has used compute resources.

Related articles on queue.acm.org

Continuous Delivery Sounds Great, but Will It Work Here?
Jez Humble
https://queue.acm.org/detail.cfm?id=3190610

Toward Higher Precision
Rick Ratzel and Rodney Greenstreet
https://queue.acm.org/detail.cfm?id=2354406

Simulators: Virtual Machines of the Past (and Future)
Bob Supnik
https://queue.acm.org/detail.cfm?id=1017002

References

1. Garrett, M. Intel Boot Guard, Coreboot and user freedom, 2015; https://mjg59.dreamwidth.org/33981.html.
2. Glass, S. Verified boot in Chrome OS and how to make it work for you. Embedded Linux Conference Europe, 2013; https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42038.pdf.
3. Goodin, D. Supermicro hardware weaknesses let researchers backdoor an IBM cloud server. arsTechnica, 2019; https://arstechnica.com/information-technology/2019/02/supermicro-hardware-weaknesses-let-researchers-backdoor-an-ibm-cloud-server/.
4. Hamilton, J. AWS Nitro System. Perspectives, 2019; https://perspectives.mvdirona.com/2019/02/aws-nitro-system/.
5. Hudson, T. Open Source Firmware Conference Security Keynote, 2018; https://trmm.net/OSFC_2018_Security_keynote#Boot_Guard.
6. Intel. Intel Data Center Block with Firmware Resilience. Solution Brief, 2017; https://www.intel.com/content/dam/www/public/us/en/documents/solution-briefs/firmware-resilience-blocks-solution-brief.pdf.
7. Intel. What is Intel Management Engine? Intel, 2017; https://www.intel.com/content/www/us/en/support/articles/000008927/software/chipset-software.html.
8. King, M., McMillan, P. Securing bare metal hardware at scale. BSides PDX, 2018; https://www.youtube.com/watch?v=PEVVRkd-wPM
9. Kumar, M. J. OCP initiatives and Intel implementations, 2018; https://www.opencompute.org/files/Intel-System-Firmware-InnovationsMohanKumar-OCP18.pdf.
10. Lattice Semiconductors. Universal Platform Firmware Resiliency (PFR) – Servers, 2018; http://www.latticesemi.com/en/Solutions/Solutions/SolutionsDetails02/PFR.
11. Leroux, S. The truth about the Intel's hidden Minix OS and security concerns. It's FOSS, 2017; https://itsfoss.com/fact-intel-minix-case/.
12. Minnich, R. et al. Replace your exploit-ridden firmware with a Linux kernel, 2017; https://schd.ws/hosted_files/osseu17/84/Replace%20UEFI%20with%20Linux.pdf.
13. OCP Server Workgroup. OCP NIC subgroup. Open Compute Project OCP NIC 3.0 Design Specification Version 0.85b, 2018 https://www.opencompute.org/wiki/Server/Mezz
14. Newman, L. H. Hack brief: Intel fixes a critical bug that lingered for 7 dang years. *Wired*, 2017; https://www.wired.com/2017/05/hack-brief-intel-fixes-critical-bug-lingered-7-dang-years/.
15. Open Compute Project. Project Cerberus. GitHub, 2018; https://github.com/opencomputeproject/Project_Olympus/tree/master/Project_Cerberus.
16. Pataky, D. Intel Management Engine. Technische Universität Dresden, 2017; https://files.bitkeks.eu/docs/intelme-report.pdf.
17. Regenscheid, A. Platform firmware resiliency guidelines. NIST Special Publication 800-193, 2018; https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-193.pdf.
18. Savagaonkar, U. et al. Titan in depth: Security in plaintext. Google Cloud, 2017; https://cloud.google.com/blog/products/gcp/titan-in-depth-security-in-plaintext.

Jessie Frazelle is an independent contractor. She has worked as an engineer at GitHub, Microsoft, Google, Docker, and several startups.

DOI:10.1145/3343052

Article development led by acmqueue
queue.acm.org

Automation helps collaboration.

BY THOMAS A. LIMONCELLI

Demo Data as Code

ENGINEERS ARE OFTEN asked to generate demo data for various reasons. It may seem like a one-time task that can be done manually and forgotten. Automating the process, however, has many benefits, and supports the inevitable need for iteration, collaboration, and future updates. When data is treated as code, you can leverage techniques from modern software engineering practices.

Many years ago I was at a company that needed to produce a demo version of its software. The demo would essentially be the company's software preloaded with fictional data. Salespeople would follow a script that would walk the customer through the features of the product. The script involved finding various problems and resolving them with the ease that only this product could provide.

Marketing would create the script, and engineering would create a dataset that would support the story.

Using live customer data in the demo was not an option because that would be a privacy violation. Even so, no one customer dataset could support the entire demo script.

This project had many red flags. Engineers were expected to work on it "in their spare time." That misunderstands and devalues engineering work. When nontechnical managers don't understand something, they often assume it is easy to do and, thus, obviously should not take very long.

More worrisome was the fact this "spare time" theory was supported by the incorrect assumption that the project was a one-time thing. That is, the data would be generated once and be perfect on the first try; the engineers could then wash their hands of it and return to their regularly scheduled work.

This assumption was intended to be a compliment to the engineers, but, "Oh, please, this will just take an afternoon!" is not a tenet of good project management.

I don't know about you, but I have never produced something for marketing without being asked for at least one revision or adjustment. This is a creative collaboration between two groups of people. Any such project requires many iterations and experiments before the results are good or good enough.

Marketing believed that by keeping the requirements vague, it would be easier for the engineers to produce the perfect dataset on the first try. This is the opposite of reality. By doing this, marketing unknowingly requested a waterfall approach, thinking that a one-and-done approach would be less wasteful of the engineers' time. The reality is that a big-bang, get-it-all-right-the-first-time approach always fails.

The primary engineer assigned to the project quickly spotted these red flags and realized that to make this project a success, he needed an approach that would allow for iteration now and provide the ability to efficiently update the project months later when version 2.0 of the software would necessitate an updated demo.

To fix this, the engineer created a system to generate the demo data from other data. It would programmatically modify the data as needed. Thus, future updates could simply regenerate the data from scratch, with slightly different operations performed on the data.

The system he created was basically a tiny language for extracting and modifying data in a repeatable way. Some of the features included:

- The ability to import data from various sources.
- The ability to insert predefined (static) data examples.
- Functions to extract data from one database, with or without clipping or filtering.
- Synthesizing fake data by calling function *f*.
- Transforming data using function *g*.
- Various anonymization methods.

The data was generated with a "program" illustrated in the accompanying figure.

This is not so much a new language as it is a library of reusable functions.

Pseudo-code for generating the demo data.

```
# Salespeople need to be able to show "problem X".
# We found this data in customer1's dataset, but we
# only need the first 200 rows:
AnonymizeAndInject("customer1.data", 200)
# NB: Approval to use customer1's data is in ticket #12345.
# NB: Anonymization technique signed-off in ticket #45678.

# The next thing sales will demonstrate is what it
# looks like when Problem X is fixed.
# Function X generates data that looks that way.
# It bases this off dataset2.data, provided by marketing.
GenerateAndInject(X, "dataset2.data")

# There is a requirement that at least one "problem Y"
# will be seen in the data. We hand-created that data.
Include("problem-y.csv")
```

New features were added on demand, adding functions as needed.

Because the demo data was being generated this way, it was easy to regenerate and iterate. For example, the marketing manager would come to us and say, "More cowbell!" and we could add a statement such as `GenerateAndInject(cowbell)`. The next day we would be told, "The cowbell looks too blue. Can it be red instead?" and we would add code to turn it red. Rerun the code and we were ready to show the next iteration.

Anonymization is particularly difficult to get right on the first try. People are very bad at anonymizing data. Algorithms are not always that much better. There will be many attempts to get this right. Once it is deemed "good enough," invariably the source data will change. Having the process automated is a blessing.

Notice the example code includes comments to record the provenance of the data and various approvals. We will be very glad these were recorded if there are ever questions, complaints, audits, or legal issues.

This was so much better than hand-editing the data.

This approach really paid off a few months later when it was time to update the demo. Version 2.0 of the software was about to ship, and the marketing managers wanted three changes. First, they wanted data that was more up to date. That was no problem. We added a function that moved all dates in the data forward by three months, thus providing a fresher look. Next, the script now included a story arc to show off a new feature, and we needed to supply data to accomplish that. That was easy, too, as we could generate appropriate data and integrate it into the database. Lastly, the new demo needed to use the newest version of the software, which had a different database schema. The code was updated as appropriate.

Oh, and it still needed to do all the things the old demo did.

If the demo data had been handcrafted, these changes would have been nearly impossible. We would have had to reproduce every single manual change and update. Who the heck could remember every little change?

Luckily, we did not have to remember. The code told us every decision we had made. What about the time one data value was cut in half so it displayed better? Nobody had to remember that. There was even a comment in the code explaining why we did it. The time we changed every data point labeled "Boise" to read "Paris?" Nobody had to remember that either. Heck, the Makefile encoded exactly how the raw customer data was extracted and cleaned.

We were able to make the requested changes easily. Even the change in database schema was not a big problem because the generator used the same library as the product. It just worked.

Yes, we did manually go over the sales script and make sure that we did not break any of the stories told during the demo. We probably could have implemented unit tests to make sure we did not break or lose them, but in this case manual testing was OK.

Creating the little language took longer than the initial "just an afternoon" estimate itself. It may have looked like a gratuitous delay to outsiders. There was pressure to "just get it done" and not invest in making a reusable framework. However, by resisting that pressure we were able to rapidly turn around change requests, deliver the final demo on time, and save time in the future.

Another benefit of this approach was that it distributed the work. Automation enables delegation. Small changes could be done by anyone; thus, the primary engineer was not a single point of failure for updates and revisions. Junior engineers were able to build experience by being involved.

I highly recommend this kind of technique any time you need to make a synthetic dataset. This is commonly needed for sales demos, developer test data, functional test data, load testing data, and many other situations.

The tools for making such a system are much better than they used to be. The project described here happened many years ago when the available tools were Perl, awk, and sed. Modern tools make this much easier. Python and Ruby make it easy to create little languages. R has many libraries specifically for importing, cleaning, and manipulating data. By storing the code and other source materials in a version-control system such as Git, you get the benefit of change history and collaboration through pull requests (PRs). Modern CI/CD (continuous integration/continuous delivery) systems can be used to provide data that is always fresh and relevant.

Anonymization is particularly difficult to get right on the first try. People are very bad at anonymizing data. Algorithms are not always that much better.

Ideally the demo data should be part of the release cycle, not an afterthought. Feature requests would include the sales narrative and supporting sample data. The feature and the corresponding demo elements would be developed concurrently and delivered at the same time.

Conclusion

A casual request for a demo dataset may seem like a one-time thing that does not need to be automated, but the reality is this is a collaborative process requiring multiple iterations and experimentation. There will undoubtedly be requests for revisions big and small, the need to match changing software, and to support new and revised demo stories. All of this makes automating the process worthwhile. Modern scripting languages make it easy to create ad hoc functions that act like a little language. A repeatable process helps collaboration, enables delegation, and saves time now and in the future.

Acknowledgments

Thanks to George Reilly (Stripe) and the many anonymous reviewers for their helpful suggestions.

Related articles on queue.acm.org

Data Sketching
Graham Cormode
https://queue.acm.org/detail.cfm?id=3104030

Automating Software Failure Reporting
Brendan Murphy
https://queue.acm.org/detail.cfm?id=1036498

Going with the Flow
Peter de Jong
https://queue.acm.org/detail.cfm?id=1122686

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

DOI:10.1145/3343048

Article development led by acmqueue
queue.acm.org

Transitioning up the ladder.

BY KATE MATSUDAIRA

The Evolution of Management

I HAVE BEEN thinking a lot about the different transitions I have made as I have been promoted to different levels of management, from individual contributor to manager to organization leader in charge of hundreds of people.

With each step up, the job changes—but not all of the changes are obvious. You must shift your mindset and focus on building new skills that are often very different from the skills that made you successful in your previous role.

There are lots of great resources for first-time managers, and many books designed for CEOs or top-level executives—but there are fewer resources specifically for the people in the middle.

Some ideas translate well from the CEO/executive content (such as establishing a team culture), but very little of the available content translates to running technical software teams at scale.

This in-between space is where I have spent almost all of my career—somewhere between individual contributor (abbreviated as IC here) and CEO. Many people (even if they are not yet managers) might be interested in practical advice for managing these transitions, so I have compiled everything I possibly could on the topic for this article.

Individual Contributor to Entry-Level Manager

Every time you move up as a leader you go through a set of changes. One of the biggest transitions occurs when you first move from an IC role into a management position.

Your impact becomes difficult to measure. As an IC you are hands-on and doing things yourself. You have a direct line between your daily tasks and the results: You write code for a feature for your team's product, and you can see the feature right before your eyes once you are finished. Every time your team reaches a milestone, you know exactly what you contributed to that success (and you can even quantify these contributions if you choose to).

When you move into management, you step away from that direct line. It is no longer your job to do the work yourself; instead, your role is to mentor, motivate, and guide your team to do the work, while you maintain the connection to the big-picture vision/strategy and make it easier for your team to get things done.

This can be one of the most difficult parts of the job for new managers to get used to. They just want to jump in and solve problems themselves, but (as anyone who has had a manager do this knows), this actually tends to do more harm than good. Those who try to do the work themselves can end up micromanaging or becoming a bottleneck on the project.

Your new job is to solve problems by removing roadblocks (including yourself), streamlining processes, and helping others be productive. *You don't solve the problem yourself now; you create an environment where other people*

IMAGE BY BERTOLD WERKMANN

can solve the problem. This is how you add value.

This is a big shift in mindset: how you think about yourself and how you define your success.

Measurements of success become lagging indicators. Unfortunately, there are not the same accolades for making that shift as there were in your previous role as an IC. Streamlining processes and mentoring your team are essential tasks but less immediately rewarding; it can sometimes take a while for the effects of your work to be truly felt and appreciated.

Understanding your impact can be elusive when the work is being done by others. And when you have a strong team, the value that you, as a leader, are bringing to the table can be hard to judge or see (this becomes more and more true as you go further up the ladder).

It is up to you to define what success means to you and your team. You achieve this by being the connection between the parent organization and your team.

Communication becomes a prized skill. Leadership is based on two-way communication (between you and your leaders, and then between you and your team), whereas before communication was more of a one-way street between your manager and you.

You must be in frequent communication with your own manager and leadership in order to understand the bigger vision for the organization, and then you can drill down on what your team needs to accomplish and why. You must work closely with your staff to make sure the best possible work gets done by the best possible people; a large part of this is helping them to understand the big-picture impact of what they are doing.

You must look further into the future than you have probably ever before to see not just the project your team is working on today, but how it will connect to projects that will be done one to two years in the future.

In summary, these are the biggest transitions that occur when moving from IC to entry-level manager:

- Let go of the immediate/quick sense of gratification that comes from doing/building/creating.
- Accolades and recognition become less frequent as you move up.
- You derive your sense of accomplishment from mentoring, growing, and furthering the work of your team and those around you.
- Add value by removing roadblocks, streamlining processes, and helping others be productive.
- Think one to two years out for your project and roadmap.
- Help people connect their work to the parent organization or company, and help them see their individual impact and value.

Entry-Level Manager to Manager of Managers

By the time you are promoted to become a manager of managers, you have some established management skills and experience under your belt. This means a less shocking transition than the one from IC to manager, but there are still plenty of changes to adapt to.

For example, you should be well practiced at letting go of micromanaging and instead trusting the people on your team to do good work (that is, delegation). As a manager of managers, however, the stakes of the work your direct reports are doing increase.

Trusting your leaders. Now, instead of your reports writing code that could be fixed in a day or two, they are making hiring decisions, managing performance, and driving strategy. A mistake here could have long-range and costly consequences, so you must learn how to balance trusting your team with avoiding a disaster. If you override your leads, it can erode trust in your team—and you quickly become a micromanager rather than a boss who empowers the team.

This ultimately comes down to knowing what calls really matter. Where do you need to be right, and where can you trust that your reports will make the right call or be able to course-correct from a bad call? Overriding your direct reports can erode their trust in you.

Knowing what calls matter comes from understanding what is most important, and how these decisions influence the overall strategy. To put things in perspective, zoom out two to three years into the future and ask yourself the following questions:

- How do all of your teams fit together?
- How should resources be distributed?
- Which projects and people are critical to the organization's most important goals?
- What lessons do you need your managers to learn?
- Where can you allow them to take control and make mistakes?
- What areas cannot fail and therefore need your oversight?

Once you have criteria for what matters, the key is implementing the right checks and balances so you can feel confident in the decisions being made (and have enough time to make adjustments when things go astray).

Your most important job becomes picking your leads. As the manager of one team, it's still possible to keep tabs on every single person. Not only do you know their names, but you probably have at least a semi-clear idea of what they are spending their time on day to day.

As a manager of managers, you have not only your direct reports but also their teams under you. That is a lot to keep track of. In fact, it is more than you *can* keep track of.

This is why it becomes critically important to get the right people into a few key roles. In engineering teams, culture can be established at the manager/product level, but it can also come in the functional unit. As such, having a few really good senior people (technologists, project managers, product managers and UX leads) can help you maintain quality, excellence, and progress. Connect with those people and make sure they know how important they are.

The way you manage people will also change. The people you are managing in these roles have been working for a while; they know how the game is played. They are there for performance ratings, promotions, and compensation. As a result, you can be more transparent in your discussions with them.

That does not mean you can completely forget about coaching and mentoring, though. As a manager, it is still your job to help your direct reports achieve their career objectives.

Planning for the future. As you move up through the ranks at your organization, so do the people underneath you. If you are an entry-level manager who loses an engineer, that won't sink the ship. As the people who report to you take on increasingly important and hard-to-replace roles, however, you have to prepare for succession.

- What will you do if your best [fill-in-the-blank] leaves?
- What can you do to help make your best people want to stay with your team?
- Which resources do you need today, and what will you need a year from now?
- Who is on your team right now who could move up in the future?
- Which jobs don't exist today that you will need filled in the future?
- Have any team members outgrown their roles, or have any of the roles changed enough that they are no longer filled by the right people?

You need to be continually looking into the future—not only in managing your people, but also in managing your team's work and objectives.

You need to know where you are going—again, you are looking at a two- to three-year time horizon—and then it's your job to set up systems that will allow you to get there. When you manage so many teams, you need to find ways to make it easy to know what is going on:

- What metrics do you need to measure and pay attention to? Why?
- How do you set up structures for visibility into progress?

This can be done in a variety of ways, from having skip-level meetings (those that involve managers and employees more than one level apart in the chain of command) to setting up reporting systems that automatically filter key data points up to you. Don't rely on just one method. Be creative, and make sure you are not getting a one-sided view.

Finally, how do you communicate those key metrics to your leadership? How will you communicate about success and failure?

Your success as a manager is now even harder to define, because what you do all day probably just looks like going to lots of meetings. There is almost no immediate, concrete output. It becomes even more critical that you get clarity from your leadership on what successful outcomes will look like for you and your team, and that you do the hard work today for big-picture results tomorrow.

Here is an overview of the transitions involved in becoming a manager of managers:

- Continue learning to let go of control and allow people to make mistakes. Balance the importance of getting your way with the risk of undermining your people. Focus on the calls that really matter.
- People management is still about mentoring, but there is a new level of transparency with your direct reports about the rewards of their work.
- Your job is to think into the future. How do all of your teams fit together? How do changes in priorities affect the way people and resources are distributed? It is always better to be the person who can do more with less.
- Succession planning comes into play. Make sure you have a solid plan to grow your leadership bench and maintain successors (or a plan) for all critical roles.
- You are responsible for progress and

execution. It's your job to make systems that work, track the right metrics, and share those results with your leadership.

Manager of Managers to Organization Leader

When moving into a role that is several layers above the engineers, your role changes again. Now you are managing organizations consisting of unique teams that may each have their own culture, process, priorities, and mode of operation.

You have to decide where the teams should have similarities and what most benefits the organization as a whole. Each structure is different, so it helps to understand the principles that hold everything together: Is it about an aligned strategy, decision making, sharing resources, or something else?

Moving from manager to executive is a huge shift in your career. Probably the biggest change you will deal with is this: *Your job is no longer the "what." Your job is now the "how."*

This shift will define almost everything you do in this leadership role. When you manage hundreds of people, you become too far removed from the people doing the "what." Instead, the way you add value is by defining and streamlining the "how."

Your job is to make an entire organization successful. Here are some of the strategies that have worked well for me in practice.

Establishing your team culture. Give up now on the idea that you will be able to stay aware of everything going on within the teams you manage. If you try to make that part of your job, you will become the thing that holds back the talented people you have in place—which is no way to retain top talent.

Instead, it is your job to establish a culture for your organization. Once you have the right people in the right places, you need to step back and let them do the work.

What does it mean to establish a culture and values? Ask yourself these questions:

- What does it mean to be in [team name]?
- What do you stand for?
- How should decisions be made?
- How should issues be escalated?
- What are the principles used to make tough calls?

Every time you move up as a leader you go through a set of changes. One of the biggest transitions occurs when you first move from an individual contributor role into a management position.

Start by defining the values that you feel are most important to your organization. This will inform everything else. You want to start at the heart of the matter—what does it mean to be on this team and what do we stand for?—and that will inform the rest of your decisions.

For example, an Amazon manager told me he never does pre-reads for the meetings he runs. Instead, he sets aside the first few minutes of every meeting for all participants to read whatever document is being discussed in this meeting. In his view, very few people pre-read, and this ensures the remaining meeting time is effective.

This might seem like a small detail, but it is indicative of someone who has a clear understanding of his own values and how he translates those values to his team. It is one small way that he executes his vision for his team culture.

You will find that a large part of your role is creating and maintaining a structure for work to be done within and communicated about. Which meetings need to happen every week? Which happen monthly, quarterly, and so on? How often do you meet with the entire organization? How do you share the roadmap, and how often?

It is also critical to identify the culture-keepers in your organization. Are they the managers? Your senior engineers? Make sure you know who they are, but also that they know who *they* are. Show them you understand their value. Help them see themselves as leaders and mentors.

Every decision you make at this level must be deliberate and reflect your organizational values. Once you make a decision, be open to reviewing it. If it's not working, iterate until it does.

Growth, mentoring, and people management. How do you connect with your team when there are too many people to count?

The answer is, you try different things until you see what works best. And then you keep trying new things to avoid falling into a rut.

Here are a few ways to make yourself available to connect with the people who report to you:

- Office hours
- Group lunches with different teams, randomly selected groups, and so on
- Sporadic one-on-ones

- Skip-level meetings
- Large-group all-hands meetings
- Regular outbound communication via email, videos, among others.

You now manage more people than you could ever possibly keep track of in your head. You may work with teams distributed around the country or around the world. Does it matter if every single engineer working on a team within a team that you manage feels like he or she knows you? That might feel like an impossible challenge if you are managing hundreds of people. But do not forget, your role is to *be* the organization at this level. You represent the company and its goals.

Feeling a connection with you is a way for the people on your team to feel connected with the company they work for. No, you won't be friends or even on a first-name basis with everyone under you but making an effort to be accessible will help people want to follow your lead over time. Based on my experience, the more people spend time with you and get to know you, the more effective you can be as a leader.

The name of the game now is trust. Your people need to trust you, and you need to trust your people.

Trust, but verify. But there is a caveat to that trust: You must also audit. Yes, you trust your managers, but you also do the work to check in.

This can take a number of forms; for example, you might do a really deep dive into one or two programs on some regular cadence. You will get a presentation from the key players and have time to ask questions about the work, timeline, and goals. You will get the chance to review architecture, call out red flags, and confirm that progress is what is expected.

Just as startups have board meetings, you can also set up monthly updates or quarterly business reviews. My favorite format involves having highlights, lowlights, and sections for special topics.

These forums are a chance for you to answer questions for the people doing (or, more likely, leading) the work, clarify your vision or goals for the project, and make sure your team is aligned. This is a much more sustainable way to work than just assuming you know what's going well and what's not, based on a bunch of status reports.

You cannot keep your eye on every single project and person every single day, but you can set up systems that allow you to keep in touch with the work being done on your team so you can communicate effectively with your peers and your leadership.

You must look further into the future than you have probably ever before to see not just the project your team is working on today, but how it will connect to projects that will be done one to two years in the future.

If you work with a manager over time and don't feel that sense of trust, it is critical to get that person out of that role or into another position as soon as possible. Your job just doesn't work without being able to trust your key leads.

Oversight and project reporting. Your managers are essentially acting as "you" (and, in turn, the organization) when they speak to their teams. You need managers in place who are in line with your vision and who will accurately present to their teams what you tell them.

If there is dissonance between what you say and what the managers say, this will gro w into a bad situation. Likewise, if what you say is different from what your peers or CEO say, that is also an opportunity to breed mistrust.

If there are managers who are not representing you accurately, then you need to correct them or replace them as soon as possible.

You should be upfront and clear with your managers about your expectations: What is their job? What is your job?

Their job is to deeply understand the company goals, and how their team fits within those goals, and to communicate to you where things are working well and where they need help.

Your job is to support their decisions (which is possible when you know you have managers who innately understand the company goals) and to communicate clearly on behalf of the company's leadership.

With so many people and projects in your purview, it is important you get regular status reports that are presented in the language of your organization (for example, OKRs at Google, Red/Yellow/Green at Nordstrom, and others). This will enable you to share progress upward as efficiently as possible.

These reports, however, will give you only the simplest view of the work being done. To truly understand what is happening on your teams, you must maintain high-quality communication with the managers working for you.

One trick another executive told me about was to select several projects to review (not exactly a deep dive but devoting more time than just reading a report) every week. Look at key metrics, milestones, deadlines, and overall status. This is a good way to quickly and randomly audit the work being done, but it is an even better way to get insight into early red flags: Are there risks? Dependencies? Changes that need to be made?

Create a spreadsheet or other structured format for reviewing projects, so that each one can be judged and discussed using the same language.

Talent review. You are now managing high-level people in important roles. It is now even more important to think about the future of those people and those roles, and to have a plan should someone need to be replaced.

Make a point of identifying the high-potential people on your team. Who are they? Are they getting what they need? What will their path forward be? You can ask your managers for their recommendations, but it is likely some standouts will be clear to you even from your relative distance.

You should also identify people who are critical—perhaps too critical—to the team's success. There should never be a single point of failure. If you would be in danger if a key person left, then you are in a really bad position that needs to be fixed right now. Find ways to create backups and have redundancy for your most critical people.

Ensure you are involved in the hiring and evaluation process for your teams, even being part of interview loops for certain positions. Sit in and hear how your managers are giving feedback to employees and how managers are interviewing potential new hires. This could teach you a lot about your team and how the culture is being broadcast. It's an opportunity to catch big red flags that could have really long-term results.

When people do leave, always take the time to do exit interviews. Uncovering the flaws in your people-management system is just as important as finding the flaws in your programs and projects. There is always a push and a pull when someone leaves, so try to uncover both and identify a way to improve.

One friend told me, always look for and check in with the smartest and most dissatisfied people on your team. They will alert you to problems within the team that might be small now but will be big problems for you in the future.

Managing your own time and resources. In this executive role, the work never stops. There will always be more than you can do. It is up to you to decide what is the most valuable use of your time and get out of the way of everything else. Hire people you trust who will help you be more effective by keeping the wheels turning while you focus on your most important tasks.

One important way you can do this is to make sure you have an exceptional assistant or other support staff. This should be someone who can read and respond to your email, because you will have a lot and most of it will not be worth your time. Get someone who is really invested in the job and who you can spend significant time with, allowing this person to understand how to make your life easier and more efficient.

You could hire a chief of staff who acts as your number two and can speak on your behalf to make decisions for the team when you are not available. This is a person who you will spend a lot of time with, working through ideas and making sure you are always on the same page. This can be a formal or informal role, but it is essential to have someone you can count on in this role.

Talk to your CEO about the best uses of your time. Talk about this with your peers, too. How do they spend their time? What do they delegate?

Make sure that, even as you delegate many tasks, you continue to put a priority on time with your customers. With all the internal work you do, it can be easy to forget about the people outside of your organization who really matter. If you lose touch with your customers, you lose touch with your goals. You cannot lead your team effectively without knowing your customers, so it is worth your while to carve out regular time in your schedule to get in tune with customers.

Working on a five-plus-year timeline. When you think about concerns, you should be thinking further out. It is not uncommon to be focused on problems two to five years in the future. It can be a distraction to get mired in the details and tactical work, but when you can, you should delegate that to your very capable leads.

One way to help this process is simply to write it all down. Create a living document that defines your priorities and high-level goals. Write out your aspirational timeline and outline your strategy for achieving it. Update your document every year and refer to it often. This will be your guide.

Maintain an open communication loop from your team, to you, to your CEO, then back to you, and then your team. You are the in-between, and you will learn how to be successful by listening.

Make sure you stay knowledgeable about your industry. You are no longer solving the problem in front of you; you are looking years into the future. Where is your industry going? What is your competition working toward? Is your company on track to fundraise, go public, multiply in size, completely change direction … ?

Set goals that are continually growing. Make sure every person on your team has goals. Some should be stretch goals, and some should be practical. Create a culture that places value on doing work that matters and gets the team's goals done, not on being the busiest or smartest or loudest person in the room.

Finally, create a roadmap and share it. You do not have to share every detail of what the next five years are going to look like, but do share the vision of what you have been working on with your team and with your CEO. Get people on board with your vision and with you, so you will have their trust and enthusiasm on the journey.

This job isn't easy, but the rewards of executing amazing work on such a huge scale are some of the best you will ever experience. C

Related articles on queue.acm.org

Evolution of the Product Manager
Ellen Chisa
https://queue.acm.org/detail.cfm?id=2683579

Getting What You Measure
Eric Bouwers, Joost Visser, and Arie van Deursen
https://queue.acm.org/detail.cfm?id=2229115

Mal Managerium: A Field Guide
Phillip Laplante
https://queue.acm.org/detail.cfm?id=1066076

Kate Matsudaira (katemats.com) is an experienced technology leader. She has worked at Microsoft and Amazon and successful startups before starting her own company, Popforms, which was acquired by Safari Books.

DOI:10.1145/3335150

When properly secured, anonymized, and optimized for research, administrative data can be put to work to help government programs better serve those in need.

BY JUSTINE S. HASTINGS, MARK HOWISON, TED LAWLESS, JOHN UCLES, AND PRESTON WHITE

Unlocking Data to Improve Public Policy

THERE IS A growing consensus among policymakers that bringing high-quality evidence to bear on public policy decisions is essential to supporting the effective and efficient government their constituencies want and need. At the U.S. federal level, this view is reflected in a recent Congressional report by the Commission on Evidence-Based Policymaking, which recommends creating a data infrastructure that enables "a future in which rigorous evidence is created efficiently, as a routine part of government operations, and used to construct effective public policy."[4]

This article describes a new approach to data infrastructure for fact-based policy, developed through a partnership between our interdisciplinary organization Research Improving People's Lives[a] and the State of Rhode Island.[13] Together, we constructed *RI 360*, an anonymized database that integrates administrative records from siloed databases across nearly every Rhode Island state agency. The comprehensive scope of *RI 360* has enabled new insights across a wide range of policy areas, and supports ongoing research into improving policies to alleviate poverty and increase economic opportunity for all Rhode Island residents (see the sidebar "Policy Areas in which *RI 360* Has Contributed Insights"). Our approach can guide other policymakers and researchers seeking to similarly transform and integrate administrative data to guide and improve policy.

The role of administrative data in policymaking. Administrative data can be collected from the computer systems used by government agencies to run their programs. When transformed into databases that are more suitable for insights, these anonymized records provide new sources of facts for policymakers to benchmark goals and measure the successes and shortcomings of existing and future programs. Often classified

[a] https://ripl.org

» key insights

- Fact-based policymaking—the practice of using data and research to guide policy decisions—is a promising solution to improving the effectiveness and efficiency of government programs.
- Administrative data can provide new facts to guide policymakers. However, understanding the quality of administrative records, and integrating, transforming, and optimizing them for policy insights present many challenges.
- To overcome these challenges, we developed an integrated database of administrative records from multiple Rhode Island state agencies with over 800 tables and 2.7 billion records related to over 4 million anonymous individuals. This data supports econometric and machine-learning research into policies with promise to deliver higher impact per dollar and better serve Rhode Island families.

PHOTO BY SIMON DANNHAUER

as "big data"[10] due to their volume, variety, and availability, administrative records are also an increasingly valuable source for empirical social science research.[5] Research with administrative records can contribute new data-driven insights to inform important policy decisions (see the sidebar "Recent Data-Driven Insights from Administrative Records"), and add objectivity and scientific rigor to measuring program impact and designing effective program changes. Moreover, scientists can inform how data from administrative systems, which are primarily designed around operational needs and often not suitable for analysis, can be transformed effectively to support research and insights.

Although the idea of guiding policy with data dates back to the 1970s and 1980s, early studies only considered isolated data sources and come from a time when data was scarce. It was not until recently that advances in data collection, storage, and scale provided the opportunity to integrate data across nearly every facet of government. Early case studies and survey studies highlight how the process of data modeling can facilitate negotiation and consensus-building among policymakers,[8] but also how the unmet promises of new information technologies prompted frustration among government leaders at that time.[9]

An important lesson is to engage policymakers and leaders to fully understand their needs, which is why we formed extensive partnerships with state government leaders while building *RI 360*. Integrated administrative data can support not only academic research, but also the analytics requirements of government itself. Like researchers, government analysts need access to data that has been transformed to provide insights and integrated across programs that serve what are often overlapping populations. For these reasons, *RI 360* was selected as the primary data source for the Rhode Island Executive Office of Health and Human Service's Data Ecosystem project, to empower its data analysts and partners with data optimized for insights.

An example policy for low-birth-weight newborns. Throughout this

Policy Areas in which *RI 360* Has Contributed Insights

- Lowering non-urgent emergency health care costs
- Curbing the opioid epidemic
- Improving worker training programs
- Creating tools to connect dislocated workers to benefits
- Helping families become more food secure
- Optimizing energy policy for low-income families
- Helping children reach proficiency on reading and math tests
- Closing the college achievement gap

Recent Data-Driven Insights from Administrative Records

- Records from the New York City criminal justice system show how judges often mispredict risk when making bail decisions.[15] Judges identify and release many defendants who have a low flight risk, but also release nearly half of the defendants with the highest flight risk. In simulations, replacing judges' decisions with a machine learning prediction can reduce either crime rates (at a fixed jailing rate) or jailing rates (at a fixed crime rate), and in both cases can reduce racial disparities in outcomes.

- Transaction data from a private grocery retailer and data from the Supplemental Nutrition Assistance Program in Rhode Island show that households treat their nutrition benefits as if they were earmarked for food expenses, even when they could be substituted for cash.[14] This finding contradicts traditional economics theory that predicts nutrition benefits should be fungible (that is, substitutable for cash), and instead supports an alternative economics hypothesis called mental accounting. Results suggest that Supplemental Nutrition Assistance Program impact on spending and nutrition can be influenced by policies governing when and how benefits are distributed.

- Federal income tax records show there are growing inequalities in life expectancy in the U.S. across socioeconomic factors.[2] The breadth and scale of this administrative data (with over 1.4 billion person-year observations) reveals that geographic factors like government expenditure and fraction of immigrants and college graduates are positively correlated with life expectancy at the bottom of the income distribution.

- Randomized field experiments in Chicago combined school and unemployment insurance records with arrest records to evaluate the impact of a summer job support program for youth.[6] By using this integrated administrative data, the study found the program caused declines in violent-crime arrests even though there were no significant effects on school or employment outcomes, which are the more typically studied effects of youth job programs.

article, we will describe our process for building *RI 360* in the context of a specific policy: determining the optimal weight threshold for providing additional medical care and resources to low-birthweight newborns and their mothers.[3] Children born with low birthweight tend to have more health difficulties and worse outcomes later in life compared to their peers. They also tend to be at higher risk, coming from disadvantaged backgrounds where mothers are more likely to be teen mothers or have reported alcohol or drug abuse. Programs to support these infants and mothers may increase equity of opportunity and reduce state and federal expenditures for support programs and anti-poverty programs later in life. Currently, the threshold for additional resources is set at 1,500 grams.[1] We use this threshold to measure the causal impact of these additional resources to determine if increasing this threshold could be a low-cost, high-return policy change that could improve lives, increase equity of opportunity, and save state and federal funds in the long run.

Using integrated data from *RI 360*, we can examine a wide range of outcomes, including educational test scores, college enrollment, use of social programs and Medicaid, and maternal care and stress. The data allows for a holistic view of policy impact; measuring gains to education and well-being from the immediate to the longer-term, and also measuring expenditure savings to government-funded social safety-net programs from early-life investments so that government can incorporate concepts of return on investment when considering how to get the most impact per dollar spent.

Our study finds that newborns just below the threshold who receive additional medical care fare significantly better later in life compared to those just above the threshold. Crossing the threshold is associated with increases in standardized test scores in elementary and middle school of 0.34 standard deviations, increases in college enrollment rates by 17.1 percentage points of a base rate of 53.6%, and decreases in social program expenditures of $27,291 by age 10 and $66,997 by age 14. Because the average cost of the additional medical services provided in the hospital at birth is approximately $4,000,[1] this study provides new facts to help policymakers evaluate the educational impact and potential financial returns of adjusting the threshold. We conclude that moving the threshold is a potential low-cost, high-impact policy lever for helping children at the margin to achieve better outcomes later in life.

To conduct this comprehensive study of outcomes for low-birthweight newborns, we access data in *RI 360* that originates from several Rhode Island agencies. Three decades of birth records from the RI Department of Health define the study population of newborns with low birthweight. The RI Department of Education provides test scores from third-, fifth-, and eighth-grade standardized tests, the PSAT, the SAT, and Advanced Placement exams; records of grade repetition, Individualized Education Programs, and disciplinary actions; and college enrollment records from the National Student Clearinghouse. The RI Department of Human Services provides enrollment and benefit payment records for Supplemental Security Income, the Supplemental Nutrition Assistance Program, Medicaid, and Temporary Assistance for Needy Families. The RI Department of Labor and Training provides quarterly wage records that measure maternal employment rates and earnings following birth. The Centers for Disease Control provide survey responses from the Pregnancy Risk Assessment Monitoring System that measure maternal attitudes and experiences following birth.

Securing the data. Figure 1 summarizes our approach and highlights the first challenge when working with administrative records: deploying security controls that protect the data. Security is our first and foremost concern because the risks of improperly securing administrative data is great. Unauthorized access or data leakage have the potential for invasions of individual's privacy, identity theft, financial fraud, or even interference with our democratic institutions, including elections. Moreover, irresponsible handling of data can have spillover effects that hinder scientific progress and policy improvement, as data owners perceive great risks of using data and partnering with scientists, even if the uses and partnerships are legitimate and secure.

We mitigate these risks by isolating all data ingest and processing within an encrypted tank (Figure 1a) inside a secure computing environment called a *data enclave*.[16] The enclave's key features are that it is physically secure and isolated from the Internet, data transfers in and out are restricted and subject to a documented approval process, all access is comprehensively audited, and access is granted to only a limited group of approved researchers. These security controls protect against unauthorized access and ensure researchers access the data in compliance with the data-sharing agreements governing its use.

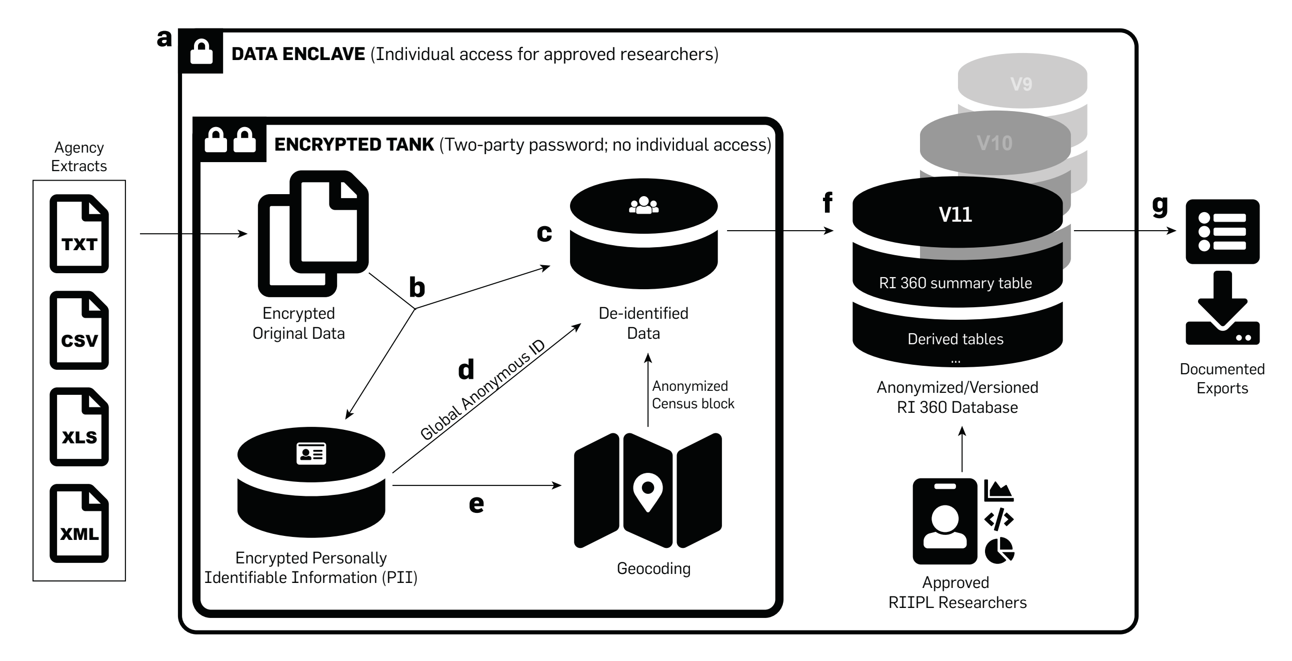
Our implementation of the data enclave uses a locally hosted system. However, modern cloud computing can help governments implement similar data enclaves using best practices for security and compliance. An additional benefit of a cloud solution is that government can own and operate the enclave, retain possession of the administrative data, and directly manage researchers' access, which removes the need for data transfers and data sharing agreements.

As an additional security measure, we restrict access to the encrypted tank using a two-party password, known only by senior leadership. A two-party password means two people each know a different half of the password, and both of the senior parties must be present and consent to access the encrypted tank. This ensures no individual researcher can access data that may reveal personally identifiable information.

Once the original data has been successfully transferred into the encrypted tank, we run an automated pipeline to separate out personally identifiable information (Figure 1b). Sensitive identification numbers—such as Social Security numbers or other identifiers deemed sensitive

Figure 1. Overview of the processing steps to secure, integrate, and conduct anonymized research with administrative data.

Agencies securely transfer data extracts to an encrypted tank inside the data enclave **(a)**. This data is split **(b)** into personally identifiable information and de-identified data **(c)**. Personally identifiable information is used to construct an anonymized global identified **(d)** and to geocode home addresses to construct an anonymized neighborhood identifier **(e)**. De-identified data is used to construct research versions of the *RI 360* database **(f)**, which can be accessed by approved researchers from inside the data enclave. Research findings can be exported from the data enclave through a documented review process **(g)**.


a
DATA ENCLAVE (Individual access for approved researchers)
Agency Extracts
TXT
CSV
XLS
XML
ENCRYPTED TANK (Two-party password; no individual access)
Encrypted Original Data
b
c
De-identified Data
d
Global Anonymous ID
Anonymized Census block
Encrypted Personally Identifiable Information (PII)
e
Geocoding
f
V9
V10
V11
RI 360 summary table
Derived tables
...
Anonymized/Versioned RI 360 Database
Approved RIIPL Researchers
g
Documented Exports

by the agency—are flagged ahead of time and automatically replaced with irreversible hashes, a technique that is widely used for protecting passwords.[11] Following this separation, the remaining data contains no personally identifiable information and is de-identified (Figure 1c).

Anonymizing the data. Once the data is secured, the next challenge is developing a method for identifying the same individual across datasets, while also preserving their anonymity so researchers cannot discover their identity, even inadvertently. Although many of the data sources for the birthweight study identify records by Social Security number, an exception is the RI Department of Education, which identifies students by name and an internal identification number. Therefore, we require an automated method to find matches among individual records based on hashed Social Security number when available, or else based on other fields like name and date of birth—all without revealing these fields to the researcher.

Our solution is to assign a global anonymous identifier (Figure 1d) to records right after separating out personally identifiable information. An automated script identifies matches among all hashed social security numbers, phonetic representations of names (using the Soundex algorithm[18]), and dates of birth. Using the global identifier, we can join information on outcomes to low-birthweight newborns and their parents in the birth records without knowing any personally identifiable information for any of the individuals.

Our deterministic algorithm is designed to minimize false matches (incorrectly matching two different individuals) at the expense of having more missed-matches (in which two records of the same individual are not matched). Some records are missing too many fields and are considered too ambiguous to assign a global identifier, but this occurs for only 3.9% of records. As an alternative to the deterministic approach, the identifier could be constructed with probabilistic record-linkage methods that would likely have fewer missed-matches, but would also carry higher costs for computation and manual curation, as well as a higher likelihood of false-matches.[12]

Integrated administrative data can support not only academic research, but also the analytics requirements of government itself.

Integrating the data. We receive data extracts from administrative systems in various formats. The raw records used in the birthweight study arrive in the encrypted tank as comma-separated text (with varying delimiters and quoting conventions), fixed-width text, XML, and Excel files. Our approach has been to meet government data partners where they are, and to accommodate data extracts in the format they can most easily produce. Most agencies have perpetual operational demands on their administrative systems, and they are not resourced to support additional development for data warehousing or analytics.

Since there is no universal format or data dictionary across agencies, we normalize the data into a consistent format and typing structure with a lightweight and open source integration tool called Secure Infrastructure for Research with Administrative Data. We developed this tool using an agile approach to meet the evolving needs of researchers and analysts as we built *RI 360*. Our GitHub repository[b] provides additional technical detail about our integration methods, as well as a worked example based on simulated data.

We chose an Extract Load Transform approach over the more typical Extract Transform Load approach.[7] In practice, this means the de-identified data is loaded into *RI 360* in as close to its original format as possible. The majority of transformations are added later after researchers have a chance to perform preliminary analyses to assess data quality and understand the data-generating processes underlying the administrative systems.

As an example, birthweight is an essential variable for defining our study population. However, it has been measured in different units (grams and ounces) over the three decades of birth records. Therefore, we construct a *birth derived* table that normalizes weight, as well as several other categorical variables measured at birth that switch from using numeric to character codes

[b] https://github.com/ripl-org/sirad-example

in the records over time. A derived table is a materialized view that aggregates, normalizes, and/or combines data from multiple original tables in *RI 360* into a single table that facilitates a specific analysis need—in this case, determining birthweight in a consistent way for all births.

A more complex example in *RI 360* is the Supplemental Nutrition Assistance Program-derived table. It combines records on applications, eligibility, benefit payments, and household structure to determine all individuals enrolled in the program at a given month and their household-level benefits.

At the highest level, we roll up all the derived tables into a single *RI 360* summary table, which spans 20 years of history for the state's most important programs and outcomes, as well as demographic information about anonymized individuals (for example, age, race, ethnicity, and sex). Most of the outcomes in the birthweight study, including educational outcomes and benefit payments, are found in the *RI 360* summary table, which reduced the effort needed to launch the study. Creating derived tables also ensures all studies using *RI 360* draw from common variable constructions and definitions that are robust and reproducible.

Supporting research integrity. A fundamental requirement of scientific findings is that they can be independently replicated by other investigators.[17] Similarly, fact-based policy should be based on robust findings that are peer-reviewed and replicable. To facilitate future replication, we update and snapshot *RI 360* approximately three times a year, creating what we call a *research version*. (Figure 1f). The research versions are de-identified data and become the permanent archive of *RI 360*. We have generated 11 such versions. Once a research version has been validated, the encrypted original data used to create that version is wiped from the encrypted tank and destroyed. Every analysis is tied to a fixed research version of the database, and can be rerun against the research version at a later time to replicate the results. Additionally, to encourage reproducibility, analysis projects use a common project template to organize code and research results in a standardized way.[c]

Even through *RI 360* has been de-identified, our data-sharing agreements restrict all research with anonymized individual-level records to the data enclave. Only aggregated or statistical results such as summary tables, plots, and regression coefficients can be exported from the enclave. All statistics must be aggregated such that they represent 11 or more distinct individuals. To ensure compliance with these agreements, no individual researcher has the ability to export files from the enclave. Copy and paste functionality has also been disabled within the enclave's user interface. Exports are subject to review and documentation to ensure exported results conform to usage agreements (Figure 1g), and they trigger real-time alerting to senior leadership. A read-only snapshot of each export is archived in the enclave to facilitate future audits.

Conclusion

The insights gained from research with administrative data have the potential to transform the way policymakers approach some of society's most important policy decisions. Robust evidence on previous policy outcomes and predictive modeling of future outcomes can guide policymakers to smarter policies with greater benefits at lower cost. We have described a comprehensive approach to overcoming the many challenges faced when integrating siloed statewide databases into a data infrastructure for fact-based policy, which is the first system of its kind in the U.S. In the future, we hope more systems of this kind will provide policymakers at all levels of government—and in many countries across the world—with a rich ecosystem and evidence base for the important decisions they make on behalf of their constituents.

Acknowledgments. This work was supported by the Smith Richardson Foundation and Laura & John Arnold Foundation.

[c] https://github.com/ripl-org/predictive-template

References

1. Almond, D., Doyle, J.J., Kowalski, A.E. and Williams, H. Estimating marginal returns to medical care: Evidence from at-risk newborn. *Quarterly J. Economics 125*, 2 (May 2010), 591–634; https://doi.org/10.1162/qjec.2010.125.2.591
2. Chetty, R., Stepner, M., Abraham, S., Lin, S., Scuderi, B., Turner, N., Bergeron, A. and Cutler, D. The association between income and life expectancy in the United States, 2001–2014. *JAMA 315*, 16 (Apr. 2016), 1750–1766; https://doi.org/10.1001/jama.2016.4226
3. Chyn, E., Gold, S. and Hastings, J. The Returns to Early-life Interventions for Very Low Birth Weight Children. Working Paper No. 25753. National Bureau of Economic Research, Cambridge, MA, 2019; https://doi.org/10.3386/w25753
4. Commission on Evidence-Based Policymaking. The Promise of Evidence-Based Policymaking (2017); https://www.cep.gov/cep-final-report.html.
5. Connelly, R., Playford, C.J., Gayle, V. and Dibben, C. The role of administrative data in the big data revolution in social science research. *Social Science Research 59*, (Sept. 2016), 1–12; https://doi.org/10.1016/j.ssresearch.2016.04.015
6. Davis, J.M.V. and Heller, S. *Rethinking the Benefits of Youth Employment Programs: The Heterogeneous Effects of Summer Jobs.* Working Paper No. 23443. National Bureau of Economic Research, Cambridge, MA, 2017; https://doi.org/10.3386/w23443
7. Dayal, U., Castellanos, M., Simitsis, A. and Wilkinson, K. Data integration flows for business intelligence. In *Proceedings of the 12th Intern. Conference on Extending Database Technology: Advances in Database Technology*. (Saint Petersburg, Russia, Mar. 24–26, 2009); https://doi.org/10.1145/1516360.1516362
8. Dutton, W.H. and Kraemer, K.L. *Modeling as Negotiating: The Political Dynamics of Computer Models in the Policy Process.* Ablex Publishing Corporation, Norwood, NJ, 1985.
9. Danziger, J. Computers and the frustrated chief executive. *Management Information Systems Quarterly 1*, 2 (June 1977), 43–53.
10. Einav, L. and Levin, J. Economics in the age of big data. *Science 346*, 6210 (2014), 1243089.
11. Gauravaram, P. Security analysis of salt||password hashes. In *Proceedings of the 2012 Intern. Conference on Advanced Computer Science Applications and Technologies*, 25–30; https://doi.org/10.1109/ACSAT.2012.49
12. Harron, K., Dibben, C., Boyd, J., Hjern, A., Azimaee, M., Barreto, M.L. and Goldstein, H. Challenges in administrative data linkage for research. *Big Data & Society 4*, 2 (Dec. 2017); https://doi.org/10.1177/2053951717745678
13. Hastings, J.S. Fact-Based Policy: How Do State and Local Governments Accomplish It? The Hamilton Project (Brookings Institution), Policy Proposal 2019-01; https://bit.ly/2VFK3og
14. Hastings, J. and Shapiro, J.M. How are SNAP benefits spent? Evidence from a retail panel. *American Economic Review 108*, 12 (Dec. 2018), 3493–3540; https://doi.org/10.1257/aer.20170866
15. Kleinberg, J., Lakkaraju, H., Leskovec, J., Ludwig, J. and Mullainathan, S. Human decisions and machine predictions. *Q J Econ 133*, 1 (Feb. 2018), 237–293; https://doi.org/10.1093/qje/qjx032
16. Lane, J. and Shipp, S. Using a remote access data enclave for data dissemination. *Intern. Journal of Digital Curation 2*, 1 (2007), 128–134; https://doi.org/10.2218/ijdc.v2i1.20
17. Peng, R.D. Reproducible research in computational science. *Science 334*, 6060 (Dec. 2011), 1226–1227; https://doi.org/10.1126/science.1213847
18. Robert, C.R. The Soundex coding system. Patent No. US1261167. 1918.

Justine S. Hastings is a professor of economics and international and public affairs at Brown University, founding director of Research Improving People's Lives, Providence, RI, and a research associate at the National Bureau of Economic Research, Cambridge, MA, USA.

Mark Howison is director of research and technology at Research Improving People's Lives, and a research associate at the Watson Institute for International and Public Affairs at Brown University, Providence, RI, USA.

Ted Lawless is a research associate at Research Improving People's Lives, Providence, RI, USA.

John Ucles is a research associate at Research Improving People's Lives, Providence, RI, USA.

Preston White is a research associate at Research Improving People's Lives, Providence, RI, USA.

DOI:10.1145/3318164

To address the computational challenges that arise when planning for robotic systems, traditional CS algorithms, tools, and paradigms must be revisited.

BY OREN SALZMAN

Sampling-Based Robot Motion Planning

IN RECENT YEARS, robots play an active role in everyday life: medical robots assist in complex surgeries; search-and-rescue robots are employed in mining accidents; and low-cost commercial robots clean houses. There is a growing need for sophisticated algorithmic tools enabling stronger capabilities for these robots. One fundamental problem that robotic researchers grapple with is *motion planning*—which deals with planning a collision-free path for a moving system in an environment cluttered with obstacles.[13,29]

TO A LAYMAN, it may seem the wide use of robots in modern life implies that the motion-planning problem has already been solved. This is far from true. There is little to no autonomy in surgical robots and every owner of a house-cleaning robot has experienced the highly simplistic (and often puzzling) routes taken by the robot.

Roughly speaking, the complexity[a] of a motion-planning problem is primarily governed by two factors: The *dimension* of the configuration space (*C*-space)—a space defined by the parameters needed to describe the robot's position and orientation, and the *tightness* of the environment—informally, an environment is said to be tight if the robot is required to move with little or no clearance[b] from the obstacles.

State-of-the-art motion-planning algorithms can efficiently construct paths for low-complexity problems (that is, either problems whose *C*-space is low-dimensional or problems that do not contain narrow passages). However, as the complexity increases, their running time may grow in an exponential fashion (exponential in the dimension of the *C*-space or in the clearance of the path that the robot needs to move along[13,25]).

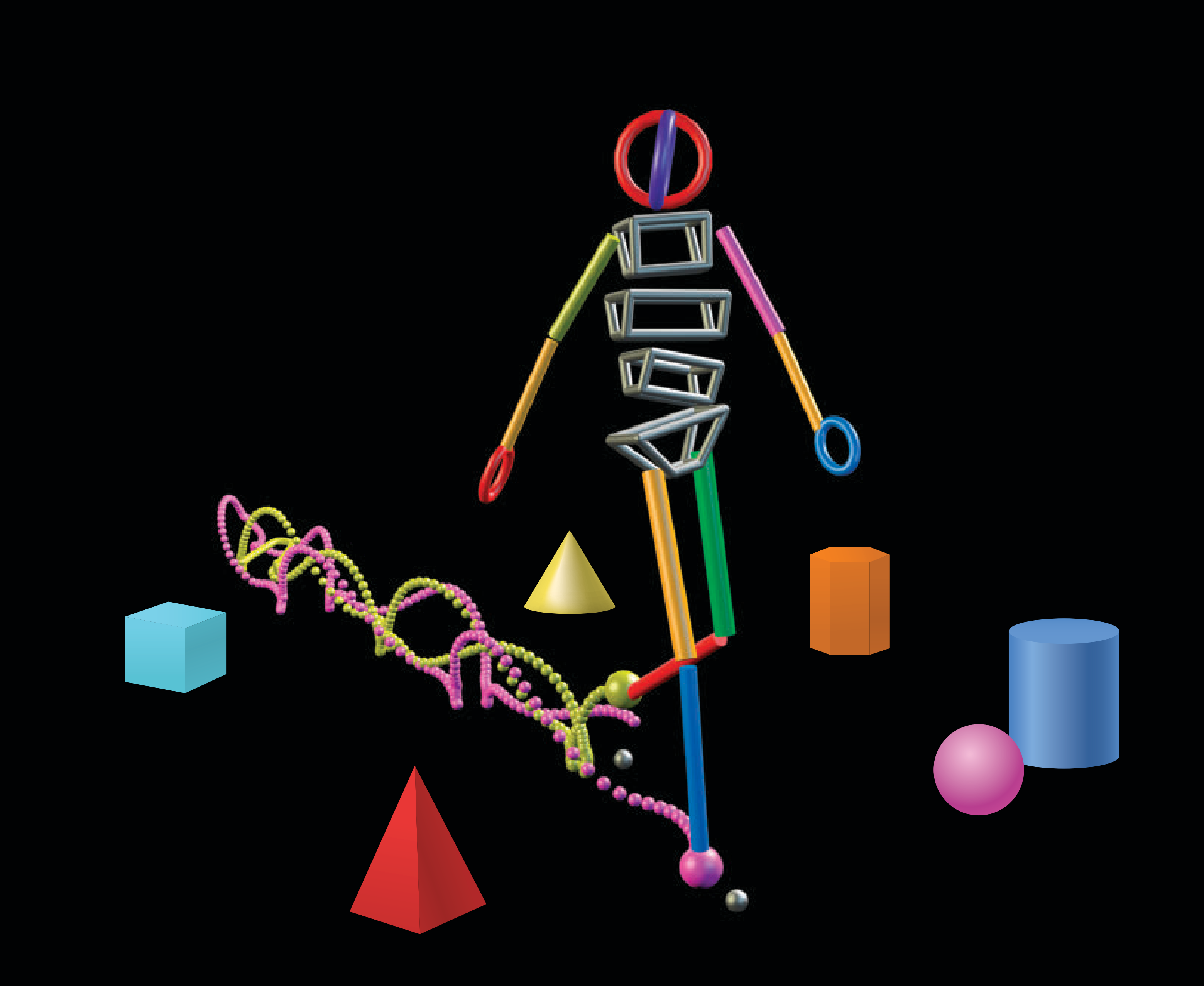
Moreover, algorithms that produce high-quality paths[c] with optimality guarantees[21] require additional overhead both in terms of running time and memory consumption when compared to the basic version of these algorithms.

In this article, I provide insight on *why* planning (high-quality) paths for complex robotic systems is computationally challenging. Specifically, after providing algorithmic background, we examine general computational challenges that arise in motion planning. This is done by examining sampling-based methods, a common approach to address the

[a] Here, the term "complexity" is used loosely to describe where state-of-the-art planners struggle—the PSPACE-Hardness proof of the motion-planning problem[13] actually argues about the complexity of obstacles. However, in practice, we can deal with a large number of obstacles using efficient data structures.

[b] The clearance of the robot is its distance from the nearest obstacle. The clearance of a robot's path is the minimal distance attained over all points along the path.

[c] In motion-planning applications, the quality of a path can be measured in terms of, for example, path length, clearance, smoothness or energy consumption along the path, to mention a few criteria.



IMAGERY FROM SHUTTERSTOCK.COM

motion-planning problem, and considering the different algorithmic building blocks that are used to design such planners and their unique computational challenges. This article focuses on the simple problem of rigid-body planning, but highlights challenges and approaches that occur when extending such algorithms to more complex settings.

There have been several earlier overviews of the robot motion-planning problem (for example, LaValle[26]). However, these were written before the seminal work of Karaman and Frazzoli on asymptotically optimal motion planning.[21] While there are some overlaps between this article and the aforementioned papers, there is a large focus on the implications of asymptotical-optimal motion planning to fundamental computational questions. In this article, I highlight new problems (such as those introduced by human-robot interaction) as well as reach out to other communities (such as the machine learning community and the computer-hardware community) for potential ways to revolutionize the field.

Algorithmic Background

Problem statement. In its basic form, the motion-planning problem is to find a collision-free path for a robot or a moving object R in a workspace W cluttered with static obstacles. The spatial pose of R, or the *configuration* of R, is uniquely defined by some set of parameters, the degrees of freedom (DOFs) of R. The set of all robot configurations X is termed the *C-space* of the robot, and decomposes into the disjoint sets of free and forbidden configurations, namely X_{free} and X_{forb}, respectively. It is common to rephrase the motion-planning problem as the problem of moving R from a start configuration to a target configuration in a path fully contained within X_{free}.

To better understand the notion of *C*-spaces, consider a polygonal robot moving in the plane. If the robot is only allowed to translate, then it suffices to use a two-dimensional reference point to describe the location of the robot. Thus, both the workspace W and the *C*-space X are subsets of $\mathbb{R}^2$. If we allow the robot to rotate as well, then we need to add another parameter to describe the orientation of the robot. In this case we have that $W \subset \mathbb{R}^2$ while $X \subset \mathbb{R}^2 \times [0, 2\pi\ )$ (see Figure 1). The dimension of the *C*-space (namely the number of DOFs) may be arbitrarily high. Consider, for example, a robot, which is comprised of n spatial links, where a rotational joint connects each two consecutive links. Here, we need a three-dimensional point to describe the position of one of the robot's endpoints and $n - 1$ two-dimensional parameters to describe the angle formed between each two consecutive links.

The configuration-space formalism was introduced in the late 1970s. Soon after, the first general algorithm for solving the motion-planning problem was proposed, with running time that is doubly exponential in the number of DOFs. Singly exponential-time algorithms have followed, but are generally considered too complicated to be implemented in practice Unfortunately, there is little hope to asymptotically reduce the exponential running times of these algorithms as the general problem was already shown by Reif to be PSPACE-Hard in the late 1970s. Additional hardness results have followed for different versions of the problem (for a complete description of exact methods and hardness results, see Halperin et al.[13] and references within).

Facing the aforementioned hardness results, the community has mostly considered approaching the general problem with heuristic and approximate schemes.[25,29] Arguably, the most widely used approach to address the motion-planning problem is via *sampling-based methods.*

Sampling-based methods. One reason why the motion-planning problem is computationally hard is due to the complexity of computing obstacles in C- space. However, testing if one specific configuration is collision-free or not, an operation referred to as "collision detection," is a relatively straightforward task that can be answered efficiently in the workspace. Using collision detection allows separating the motion-planning problem from the particular geometric and kinematic models of the robot. Furthermore, while the general motion-planning problem is computationally hard, computing a local path connecting two close-by configurations and then validating if this path is collision-free is, relatively speaking, a straightforward task. Following these key insights, sampling-based motion-planning algorithms abstract the robot as a point in the C-space X and plan a path in this space. The structure of X is then studied by constructing a graph $\mathcal{G}$, called a roadmap that approximates the connectivity of X_{free}. Roughly speaking, nodes of the graph are collision-free configurations and two (nearby) nodes are connected by an edge if the local path connecting their respective configurations is collision free as well.

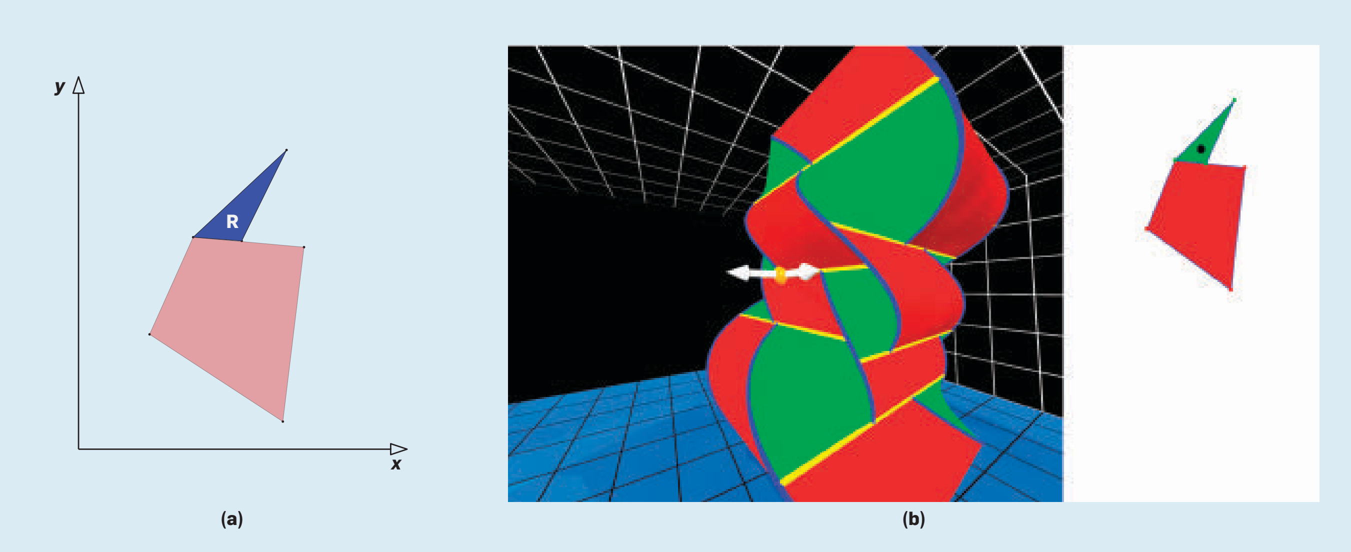
These algorithms consist of a *roadmap-construction* phase and a *roadmap-traversal* phase. These two phases can be separate, where $\mathcal{G}$ is first constructed (explicitly or implicitly) and only then searched for a (collision-free) path connecting the start and target configurations. Alternatively, the two phases can be interleaved where $\mathcal{G}$ is incrementally constructed and the search algorithm is dynamically updated as vertices and edges are added to $\mathcal{G}$.

Generally speaking, there are two different flavors of the aforementioned approach—*sampling-based methods* and *search-based methods*. In the former, which is the focus of this article, vertices are added to $\mathcal{G}$ by randomly sampling X while in the latter $\mathcal{G}$ is defined implicitly by describing a predefined set of actions, or local paths called "motion primitives" that a robot can perform at any given configuration. Both flavors can be used to solve *single-query* problems, where start and target configurations are provided to the planner or *multi-query* problems where the setting is preprocessed to efficiently answer multiple queries.

Such motion-planning algorithms are implemented using two primitive operations: Collision detection (CD), which is used to assess if a configuration is collision-free or not, and nearest neighbor (NN) search, which is used to efficiently return the neighbor (or neighbors) of a given configuration. CD is also used to test if, given two configurations q, q', a path $\pi(q, q')$ connecting the two configurations lies in X_{free}—a procedure referred to as *local planning*.

Arguably, the best-known and conceptually simplest sampling-based

Figure 1. (a) Translating and rotating polygonal robot *R* (blue) and one polygonal obstacle (light red). (b) Corresponding configuration (yellow point) and obstacle in the *C*-space $\mathbb{R}^2 \times [0, 2\pi)$.



algorithms are the Probabilistic Roadmap Method (PRM) and the Rapidly Exploring Random Tree (RRT), which are examples of multi-query and single-query motion-planning algorithms, respectively.[25] For a visualization illustrating the way the PRM algorithm constructs a roadmap in the preprocessing phase and computes a collision-free roadmap in the query stage, as illustrated in Figure 2.

General Computational Challenges in Motion Planning

The high-level description of sampling-based planners provided earlier leaves many questions that must be addressed. Here, several of these questions are listed together with the computational challenges they introduce. While the focus here is on sampling-based planners, many of these questions and challenges arise in search-based planning algorithms as well, but a detailed analysis of such algorithms is out of the scope of this article.

How to efficiently and effectively sample X? A key question in sampling-based algorithms that must be addressed is how to choose the set of configurations that will serve as vertices in $\mathcal{G}$. Intuitively, we want to *cover* the entire C-space (exploration) while using previous information to bias where new samples are placed (exploitation). The simplest approach that works well in practice is to sample points uniformly from the C-space. Many heuristics were suggested in order to speed up the probability of finding a solution. Examples include sampling near the obstacles, on the medial axis, and more.[25] Recent work has also used learning to accelerate the planning in order to devise non-uniform sampling strategies that favor sampling in those regions where an optimal solution might lie (for example, Ichter et al.[18]).

Once an initial solution is found only configurations that can provide a better solution should be considered. Considering only this set of configurations, termed the *informed set* X_{inf} , can dramatically improve the performance of asymptotically optimal motion planners such as RRT*.[10] When the C-space is Euclidean, X_{inf} is a pro-late hyper-ellipsoid[10] and it is possible to directly sample X_{inf} Several heuristic approaches attempt to generalize this result to non-Euclidean C-spaces (for example, see Kunz et al.[24]), however efficiently sampling X_{inf} for general C-spaces remains an open challenge.

From a computational point of view, sampling uniformly from the C-space can be done in O (1) time. This constant is inversely proportional to the ratio of the volume of X_{free} and X and may be extremely large in certain settings. Sampling in the informed set X_{inf} for Euclidean spaces, can still be done in O (1) time, however, it is not clear that this is the case for more complex C-spaces.

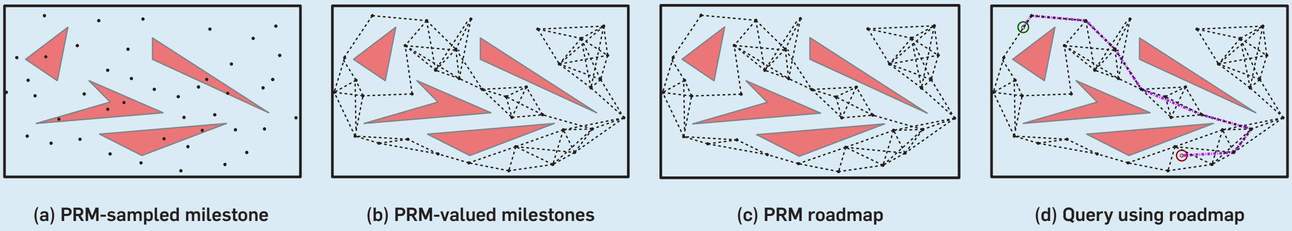
Deterministic vs. probabilistic planning. In certain domains, applying probabilistic algorithms such as the PRM, RRT and their many variants may be unacceptable. This may be due, for example, to certification and reproducibility requirements for safety-critical applications. An additional, computational reason to avoid probabilistic algorithms is the desire to use offline computation to improve online performance.

One approach is to replace probabilistic sequences with deterministic ones in sampling-based algorithms.[25,27] This raises the question regarding the theoretical guarantees and practical performance that could be obtained by using such sequences. In a recent work, Janson et al.[19] show that using (low-dispersion) deterministic sampling strategies can provide substantial benefits when compared to probabilistic ones. These benefits are both with respect to reduced computational complexity (for a given number of samples) and superior empirical performance on a wide range of problems. Their work opens the door to using deterministic sampling in optimal kino-dynamic motion planning, anytime algorithms such as RRT, designing new algorithms that explicitly leverage the structure of low-dispersion sequences and more. A key question here is how to employ deterministic sampling sequences when we are only interested in samples that lie in the informed set X_{inf}.

An alternative approach that avoids probabilistic algorithms, mentioned previously, is using search-based algorithms that systematically search over an implicit graph $\mathcal{G}$.

Figure 2. A PRM example.

In the preprocessing phase, (a) a roadmap $\mathcal{G}$ is constructed by sampling n configurations, or milestones, and (b) adding them as vertices to $\mathcal{G}$ if they are collision free. (c) If the path connecting two close-by configurations (computed using a local planner) is collision free, then an edge between the two respective vertices is added to $\mathcal{G}$. (d) In the query phase, start and target configurations (green and red circles, respectively) are provided to the planner. Using the local planner, local paths connecting these configurations to $\mathcal{G}$ are computed. If these local paths are collision free, then any graph-search algorithm can be used to assess if a path exists between the start and goal configurations in the roadmap.


(a) PRM-sampled milestone

(b) PRM-valued milestones

(c) PRM roadmap

(d) Query using roadmap

What metric should be used to evaluate distances in *X*? Recall that once a set of configurations is sampled, close-by configurations are connected by edges in $\mathcal{G}$. Thus, we must define a distance metric in *X*. Amato et al. were the first to study the effect of a metric on sampling-based planners. Specifically, they study the PRM algorithm and compare the effectiveness of some variants of the Euclidean metric when the *C*-space is SE(**3**) (see LaValle[25]). Extensive research has been carried out in order to find suitable metrics for other settings of motion planning, such as robots with differential constraints.[6]

In a recent work, Ekenna et al. employ machine learning to develop metrics tailored to the specific motion-planning problem at hand.[9] A key insight from their work is that a set of metrics, each suitable for a different setting, can be combined in order to solve more diverse settings that consist of smaller, specific, (sub)settings. Choosing the right metric(s) for a specific problem that both captures the notion of "similar" configurations and is efficient to compute is still far from being solved.

Part of the reason that choosing the right metric is such a challenge is due to the so-called "curse of dimensionality" or the "surprising behavior of distance metrics in high dimensional space."[2] Roughly speaking, in high-dimensional spaces data becomes sparse, and nearest-neighbors (NN) algorithms fail both from an efficiency and from effectiveness perspective. Indeed, it has been shown that in high-dimensional spaces the concept of proximity, distance or nearest neighbor may not even be qualitatively meaningful. Complementing this analysis, Plaku and Kavraki[32] empirically demonstrate that after a critical dimension (between 15 and 30), exact NN algorithms examine almost all the samples. This implies that exact NN algorithms become impractical for sampling-based motion planners when a considerably large number of samples need to be generated. The impracticality of exact NN algorithms motivates the use of approximate algorithms, which trade off accuracy for efficiency and motivates our next computational challenge.

How to efficiently compute nearest-neighbors? Here, efficiently computing such distances is discussed using NN data structures.

Analyzing sampling-based, motion-planning algorithms is typically done by considering their asymptotic behavior.

The first question to ask is what is the computational role of NN search in motion planning? Asymptotically, the computational complexity of NN search dominates the running time of many sampling-based motion-planning algorithms. However, in practical settings the main computational bottleneck is often considered to be the local-planning phase.[25] Having said that, this may not always be the case. In many settings, called *NN-sensitive*, the (computational) role of NN search after a finite time in many sampling-based algorithms is far from negligible and merits the use of advanced data structures.[23] NN-sensitive settings may be due to: planners that algorithmically shift the computational weight of the algorithm to NN search; scenarios in which certain planners perform mostly NN search; and fine-tuned parameters for which certain planners resort to performing many NN queries. Thus, specifically tailored NN techniques for motion planning were suggested that exploit the specific queries of the motion-planning algorithm or the specific topology of *X* (for example, see Ichnowski and Alterovitz[16] and Figure 3).

Using the aforementioned methods allows to significantly improve the running time of many search algorithms in many settings when compared to using standard NN approaches. However, they are typically not suitable for the general type of *C*-spaces that are encountered in motion planning.

How to connect close-by configurations? In the simplistic description provided earlier, we assumed that connecting two close-by configurations is a relatively straightforward task. This corresponds to solving the so-called two-point boundary value problem (BVP).[25] For systems with dynamics it is difficult, and sometimes impractical, to generate an exact BVP solver.

One approach would be to use recent advances in optimization and optimal control to exactly connect close-by nodes,[39] however, this step, performed online, may be computationally complex and introduces an additional nontrivial overhead. Another approach is to forego the requirement to exactly connect close-by configurations and build a tree-like structure by forward simulating the system's dynamics.[28]

How to efficiently and effectively search a given roadmap? In many settings, the computational cost of evaluating whether an edge of $\mathcal{G}$ is collision-free or not dominates the running time of motion-planning algorithms. Thus, when the environment can be preprocessed a common approach is to (implicitly) construct the roadmap $\mathcal{G}$ *without* evaluating if edges are collision free or not—an approach referred to as *lazy construction*.[d] Subsequently, standard path-planning algorithm such as A* can be used to traverse $\mathcal{G}$ while evaluating edges as they are being considered. However, this may lead to large planning times because these search algorithms are typically designed to minimize node expansion and not edge evaluation.

While it may seem logical to employ algorithms that minimize the number of edge evaluations,[12] what we actually want is a method to minimize the total planning time by balancing edge evaluations and graph operations. This can be done using advanced graph search[30] or by employing Bayesian active learning.[7]

How to analyze sampling-based methods? Analyzing sampling-based, motion-planning algorithms is typically done by considering their *asymptotic* behavior. Namely, does a property hold with high probability as the number of samples tends to infinity.

Recall that one fundamental insight that inspired roadmap-based algorithms was using local paths to connect nearby configurations. The longer the distance is between two configurations q and q', the higher the probability that the local path $\pi(q,q')$ connecting q and q' will intersect an obstacle. Furthermore, in many implementations, the time to verify that a path is collision free is proportional to its length, which further motivates only connecting close-by configurations. Thus, a key challenge here is understanding what is the minimal threshold r that is required to connect any two vertices in $\mathcal{G}$ in order to satisfy a certain property?

The most fundamental property required from a sampling-based motion-planning algorithm is *probabilistic completeness*. Namely, the probability that the algorithm finds a solution, if one exists, tends to 1 as the number of samples n tends to infinity.

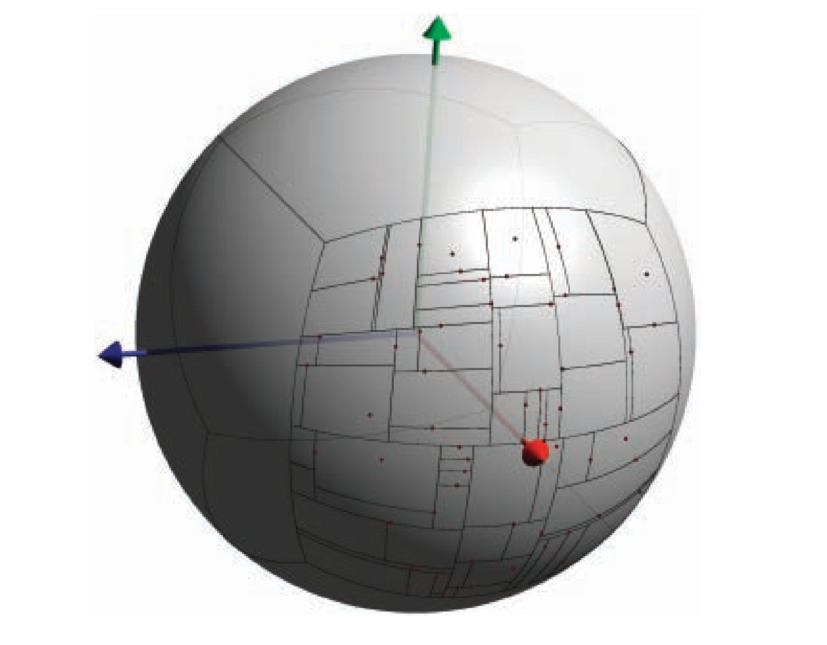
The PRM algorithm was first shown to be probabilistic complete for the case that there is an edge between every two vertices in $\mathcal{G}$.[25] This leads to a graph of size $O(n^2)$, which is computationally inefficient both with respect to the running time of the algorithm and with respect to the storage required. Later analysis (for example, Karaman and Frazzoli[21] and Solovey et al.[36]) showed that connecting two vertices whose distance is at most $r = O((\log n/n)^{1/d})$ suffices to ensure probabilistic completeness (here, d is the dimension of X) leading to a graph of size $O(n \log n)$.

This discussion is concerned with finding a path. In practice, we wish to find a high-quality path (given some optimization criteria). In their seminal work, Karaman and Frazzoli[21] give a rigorous analysis of the performance of the RRT and PRM algorithms. They show that with probability one, common implementations of these algorithms will not produce the optimal path. By modifying the connection scheme of a new sample to the existing data structure, they propose the PRM* and the RRT* algorithms (variants of the PRM and RRT algorithms, respectively) all of which are shown to be asymptotically optimal. Namely, as the number of samples tends to infinity, the solution obtained by these algorithms converges to the optimal solution with probability one. To ensure asymptotic optimality, the number of nodes each new sample is connected to is proportional to $\log(n)$ and the overall running time of these algorithms is $O(n \log n)$, where the big-O notation hides an exponential dependency on the dimension d. Recent work showed that in certain cases the connection radius can be further reduced and asymptotic (near-)optimality can be achieved when each sample is connected to only $\Theta(1)$ neighbors.[35]

Interestingly, roadmaps constructed by many sampling-based algorithms such as PRM and RRT coincide, in the absence of obstacles, with standard models of random geometric graphs (RGGs). Indeed, Karaman and Frazzoli[21] conjectured that a sampling-based planner has a certain property if

Figure 3. Computing NN in C-spaces that arise in motion-planning problems.

Here, Ichnowski and Alterovitz[16] adapt *kd*-trees for the case that $X = SE(3)$, namely, the six-dimensional space that represents rotations and translations in three dimensions.



[d] In practice, the roadmap is typically preprocessed to include only edges that *may* be collision free and edges that correspond to self-collision, unstable configurations, and so on, are removed.

the underlying RGG has this property as well. This conjecture was settled by Solovey et al.[36] who suggested a framework that allows to easily transfer arguments regarding properties of RGGs in the free unit-hypercube to spaces punctured by obstacles, which are geometrically and topologically much more complex. However, their framework holds only for Euclidean *C*-spaces and the standard Euclidean distance.

Extending this analysis to non-Euclidean spaces remains an ongoing challenge. In addition, asymptotic properties are, in fact, quite weak and are of little use in practical settings. A more interesting challenge, that is only partially answered, is arguing about finite-time properties of these algorithms—namely, what guarantees can we provide given a fixed, predefined budget of samples and / or connections (for example, see Janson et al.[19]).

How to characterize paths and *C*-spaces? Intuitively speaking, the success of roadmap-based methods is due to the fact the structure of *C*-space is typically not "pathological" everywhere. Indeed, Hsu et al.[15] attempt to attribute the success of PRM-like planners to favorable "visibility" properties of the *C*-space. Other approaches for analyzing sampling-based, motion-planning algorithms typically reasons about the clearance of a path[21,25] and some global properties of *X* such as ε-goodness or expansiveness.[25]

Figure 4. Narrow passages in three-dimensional *C*-spaces.

Both passages (yellow) have a volume of ε^2, thus when using uniform point samples, the probability of sampling in either passage is identical. However, the left passage can be characterized as "two-dimensional" while the right passage as "one-dimensional." Figure taken from Salman et al.,[33] which also includes additional information.

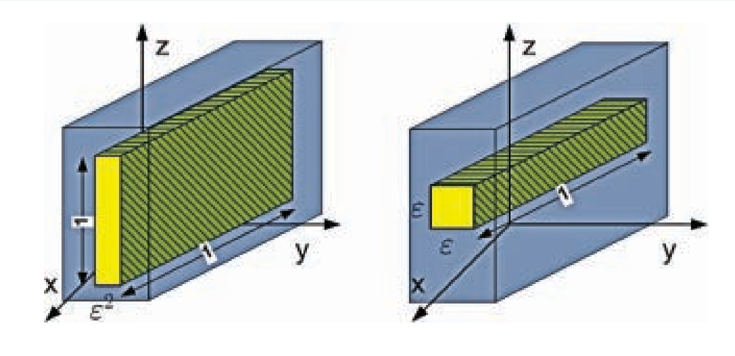
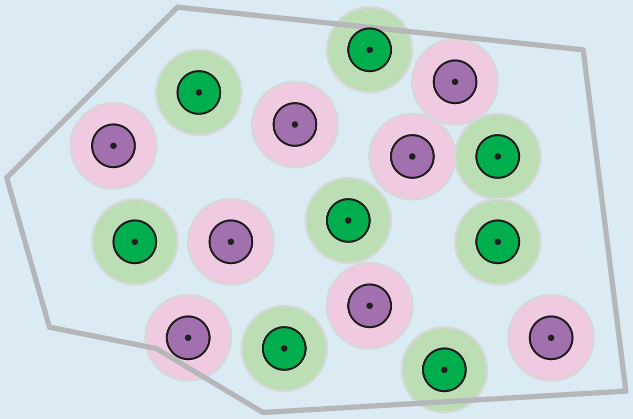


Figure 5. Example of a *well-separated* scenario for a set of identical unit discs.

The start and target positions are depicted in dark green and purple (respectively). The light green and purple circles represent discs of radius 2 placed at start and targets. Note that no two such discs intersect, which implies that every two start or target positions are at distance at least 4.



Recently, it has been noticed that more refined models can help in better capturing attributes of the *C*-space. Salzman et al.[33] observed that their planner performs well even in the presence of narrow passages. Further investigation revealed this occurs because only a small subset of the robot's degrees of freedom were simultaneously constrained (see Figure 4.) This led to a set of definitions that attempt to simultaneously grasp the "narrowness" of a passage as well as its "dimension." Although intuition may suggest that narrow passages are tunnel-shaped, a one-dimensional tunnel in a high-dimensional *C*-space would correspond to a simultaneous coupling of all parameters, which is a rarity.

Yet another example is recent work on multi-robot motion planning.[1] Here, efficient algorithms were devised by requiring a limited amount of spacing, called *separability*, among robots in their initial and final placements (see Figure 5). This notion of separability is dramatically different than that of clearance—in the latter the assumption is that a solution path has sufficient clearance in the *C*-space throughout the entire path. In the former, only separability of the robots in their initial and final placements is assumed: No assumptions are made regarding the clearance of the robots from the obstacles, or about clearance or separability along the path.

Following this discussion, there is a need for realistic models that capture non-pathological motion-planning *C*-spaces. Such models should allow for analyzing existing algorithms and to suggest efficient new ones. For example, such a model could encompass the minimal distance between obstacles (notice this is not identical to the notion of clearance, which is typically defined with respect to a path). Indeed, researchers have attempted to take structure in the *C*-space into account (for example, see Vernaza and Lee[38]). However, there is still much work needed on data structures and algorithms that efficiently exploit the structure of *C*-spaces. Note that a similar problem of realistic models has been widely addressed in the computational geometry literature and used for devising efficient motion-planning algorithms (for example, van der Stappem et al.[37]). However, it is not obvious they are relevant to high dimensional realistic *C*-spaces when one uses roadmap-based motion-planning algorithms.

How does the cost function affect the computational complexity? Up until now, we ignored the specific cost function for which we want to compute an (asymptotically) optimal path. Most analyses of sampling-based planners[20,21] assume the cost function is Euclidean and computed in the *C*-space. Janson et al.[20] showed that the guarantees regarding asymptotic optimality of many planners can be extended to other cost functions such as metric costs, line-integral costs with optimal-path connections and line-integral costs with straight-line connections. This requires computing the volume of the unit cost-ball, which may be highly non-trivial for complex *C*-spaces.[23]

In practical applications that involve human-robot interaction (HRI), we are interested in optimizing paths over complex cost functions that take the interaction with humans into account. One such example is *legibility* and *predictability* where we would like to generate motions that are intent expressive. Currently such problems have only been solved using locally optimal optimization based methods[8] and it is not clear if (asymptotically) optimal paths can be computed for such settings. Another example that arises from HRI is following tasks generated by an expert such as a surgeon in robot-assisted surgery. To accurately follow such paths, the Fréchet metric was employed[14] to compare paths in *Task space*—the space of positions and orientations of the robot's end effector. While the authors present a proof that their sampling-based algorithm is asymptotically optimal, this is done using a restricting set of assumptions and a more general proof is needed.

Putting it all together—How to efficiently compute high-quality paths? Previously, we detailed different computational challenges in roadmap-based planning algorithms and provided some insights on why general computer-science tools such as NN and graph search must be revisited in order to be better suited for this domain. Combining insights gained from analyzing the theoretical properties while addressing the specific challenges mentioned into an algorithmic framework is highly non-trivial. This is especially true when computing high-quality paths.

In practical applications that involve human-robot interaction, we are interested in optimizing paths over complex cost functions that take the interaction with humans into account.

After Karaman and Frazzoli[21] introduced the first asymptotically optimal sampling-based motion algorithms many variants followed. These algorithms used techniques such as relaxation methods,[4] lazy computation,20 heuristics,[11] and relaxing optimality to near optimally[28,e] to improve the practical convergence rate while maintaining the desired theoretic guarantees on convergence to (near) optimal solutions.

One important, yet often overlooked, problem is parameter tuning. There are a myriad of motion-planning algorithms, each with their own suit of parameters. Better characterizing *C*-spaces, would allow us to map a *C*-space to the optimal choice of parameters for a given algorithm. While this seems extremely challenging, recent developments in machine learning may serve as a valuable tool in approximating such a mapping.

Taking it all apart—Are we solving the right problem? Sampling-based methods revolutionized the field of motion planning. In contrast to exact algorithms, which are too complex to implement in practice, these methods are typically easy to implement and could be deployed on physical robots. They served as a paradigm shift by only approximating the structure of *X* instead of computing an explicit representation of it. The challenges discussed in the previous sections suggest that perhaps the community is in need for a new paradigm shift.

One question that comes to mind is do we care about computing a collision-free path? When humans compute a plan to a destination they don't verify that it can be executed, but typically plan paths that can be executed with high probability. While interleaving planning and execution has been studied, doing so while pro-

[e] An algorithm ALG is said to be asymptotically near-optimal if, given an approximation factor $\varepsilon \geq 0$, the solution obtained by ALG converges to within a factor of $(1 + \varepsilon)$ of the optimal solution with probability one, as the number of samples tends to infinity.

viding formal guarantees is far from being answered. Another question relates to the underlying assumption of roadmap-based methods where the high-level motion-planning problem can be solved effectively by reducing it to solving multiple local-planning problems by connecting close-by configurations. Effective metrics are difficult to define and distances are hard to compute in high-dimensional spaces. Can we define and analyze practical motion-planning algorithms that forego this requirement? Furthermore, *C*-spaces tend to contain large, open regions where planners could make use of long edges (which are currently approximated using a sequence of small edges). Is the search for the minimal connection radius the right approach?

Sampling-based methods revolutionized the field of motion planning. In contrast to exact algorithms, which are too complex to implement in practice, these methods are typically easy to implement and could be deployed on physical robots.

A third question is can we shift the burden of planning from the algorithm to either the robot design or the environment design? If we understand what makes a motion-planning problem easy, can we design robots or environments that do not require advanced motion-planning capabilities? Can we effectively manipulate the environment or the robot (offline or online) to simplify the motion-planning problem? These types of questions have raised some attention in the previous years (for example, see Schulz et al.[34]) but merit a deeper investigation.

Finally, a key drawback of sampling-based algorithms is that they do not know how to terminate when there is no free path. While generally solving this problem seems intractable, a possible approach would be to use the concept of resolution exact (or ε-exact) planners.[40] Such planners accept as an input a resolution parameter ε and an accuracy constant K such that if there is a path of clearance $\mathrm{K}\varepsilon$, it will output a path and if there is no path of clearance ε/K, it will correctly output that no path exists. While such planners have been proposed (see Zhou et al.[40] and references within), they are applicable for only a small set of robot systems and it would be interesting to see if more general planners with similar guarantees can be suggested.

Discussion and Future Directions

This article described several of the algorithmic challenges that arise when planning for robot systems, with a focus on the foundational algorithmic challenges prevalent even for relatively simple domains. In a nutshell, all these challenges arise because the underlying planning problem resides in a high-dimensional, topologically complex, continuous *C*-space cluttered (non-uniformly) with obstacles. Motion-planning algorithms use operations such as collision detection and graph search that reside in low-dimensional, possibly discrete, spaces to gain insight on the true structure of the *C*-space. These operations are computationally expensive and often only approximate certain properties of the underlying *C*-space (recall the role of metrics in roadmap-based algorithms).

In practical settings, in addition to the challenges mentioned here, there is a high degree of uncertainty that must modeled and addressed. This uncertainty can be due to imperfect information regarding the robot model, the environment (noisy sensor data), the robot's dynamics, and more. Accounting for uncertainty may require planning in infinite-dimensional belief spaces instead of finite-dimensional *C*-spaces, tight integration with perception, interleaving planning with execution, and more.

Similarly, almost all the research mentioned in this article assumes one perfect given model of the environment. Even if sensing and perception can obtain such a model, is it all needed? A key computational challenge is understanding what is the minimal model of the environment that is required to complete a motion planning task? Should all the objects be modeled using the same resolution? Does the planning algorithm need to reason on physical properties of the environment (speed of moving objects, the articulation or deformability of objects in the environment, and so on).

Furthermore, real-world applications often require fast and accurate physics simulation for planning with dynamics or for manipulation. Accounting for the computational cost of such simulators and for their inaccuracies is required for these planners to be effective in practice.

An orthogonal approach that could dramatically improve motion-planning algorithms is looking at the problem from the hardware perspective. Indeed, recent work suggested minimizing collision detection time by aggressively preprocessing a given scenario for a given robot. This required designing robot-specific circuitry[31] and it is interesting to see if this approach can be generalized to non-static environments. Another avenue where hardware can be exploited is parallelization—motion-planning implementations tend to be highly sequential and any advances in effective parallelization or amenability to highly parallel paradigms would also help the field (also see Ichnowski and Alterovitz[17] and references within). Finally, recent advances in cloud-based computation could be highly beneficial and has raised some initial attention from the motion-planning community.[22]

From the application point of view, robots are leaving the cages of the industrial manufacturing production lines and the safety of research labs, and moving into the unstructured environments of everyday life. From human-in-the-way to human-in-the-loop, modern robotic problems typically involve robot interactions with and around humans. Solving such problems requires research in complementary areas: algorithmic robotics, such as motion planning and human-robot interaction, such as cognitive modeling, intention recognition, and activity prediction. Accounting for humans in the planning domain adds a multitude of algorithmic constraints—from modeling human behavior to computing consistent, predictable, and safe paths. However, they also allow for additional flexibility.

Finally, as robots are being deployed, the robotics community is collectively gathering experience and data. Leveraging this experience and data to improve the efficiency of planning algorithms is an ongoing challenge—from incorporating precomputed paths in roadmap-based algorithms to applying advances in machine learning to understand when and how to apply existing tools, or to develop new tools altogether.

One should not see learning as an alternative to algorithmic, roadmap-based planning, but as a complementary tool—organized search can act as scaffolding for machine learning algorithms. While machine learning exploits correlations between similar problem instances, search exploits the structure within a problem. Thus, the two are quite complementary. Furthermore, machine learning algorithms are typically data hungry and in robotics there is often limited access to huge amounts of real-world data. For an overview of additional challenges and opportunities for robot planning, see Alterovitz et al.[3]

To truly impact our world, robot-planning capabilities must be enhanced. To do so, robotic researchers need to harness tools from other communities and revisit existing, traditional algorithmic tools in order to make them suitable for the unique, subtle challenges that arise in this domain.

References

1. Adler, A., Berg, M., Halperin, D. and Solovey, K. Efficient multi-robot motion planning for unlabeled discs in simple polygons. *IEEE Trans. Automation Science and Engineering 12*, 4 (2015), 1309–1317.
2. Aggarwal, C.C., Hinneburg, A. and Keim, D.A. On the surprising behavior of distance metrics in high dimensional space. In *Proceedings of the Intern. Conference on Database Theory*. Springer, 2001, 420–434.
3. Alterovitz, R., Koenig, S. and Likhachev, M. Robot planning in the real world: Research challenges and opportunities. *AI Magazine 37*, 2 (2016), 76–84.
4. Arslan, O. and Tsiotras, P. Use of relaxation methods in sampling-based algorithms for optimal motion planning. *ICRA*, 2013, 2421–2428.
5. Atariah, D. and Rote, G. Configuration space visualization (video). *SOCG*, 2012 http://computational-geometry.org/SoCG- videos/socg12video/
6. Boeuf, A., Cortés, J., Alami, R. and Siméon, T. Enhancing sampling-based kinodynamic motion planning for quadrotors. *IROS*, 2015, 2447–2452.
7. Choudhury, S., Srinivasa, S. and Scherer, S. Bayesian active edge evaluation on expensive graphs. *IJCAI*, 2018, 4890–4897.
8. Dragan, A.D., Lee, K.C.T. and Srinivasa, S.S. Legibility and predictability of robot motion. *HRI*, 2013, 301–308.
9. Ekenna, C., Jacobs, S.A., Thomas, S. and Amato, N.M. Adaptive neighbor connection for PRMs: A natural fit for heterogeneous environments and parallelism. *IROS*, 2013, 1249–1256.
10. Gammell, J.D., Barfoot, T.D., and Srinivasa, S.S. Informed sampling for asymptotically optimal path planning. *IEEE Trans. Robotics 34*, 4 (2018), 966–984.
11. Gammell, J.D., Srinivasa, S.S. and Barfoot, T.D. Batch Informed Trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. *ICRA*, 2015, 3067–3074.
12. Haghtalab, N., Mackenzie, S., Procaccia, A.D., Salzman, O. and Srinivasa, S.S. The provable virtue of laziness in motion planning. *ICAPS*, 2018, 106–113.
13. Halperin, D., Salzman, O. and Sharir, M. Algorithmic motion planning. *Handbook of Discrete and Computational Geometry* (3rd ed.), J.E. Goodman, C.D. Toth, J. O'Rourke (Eds.). CRC Press, Inc., 2017, 1307–1338.
14. Holladay, R.M., Salzman, O. and Srinivasa, S.S. Minimizing task space Frechet error via efficient incremental graph search. *IEEE Robotics and Automation Letters* (2019). To appear.
15. Hsu, D., Latombe, J., and Kurniawati, H. On the probabilistic foundations of probabilistic roadmap planning. I. *J. Robotics Res. 25*, 7 (2006), 627–643.
16. Ichnowski, J. and Alterovitz, R. Fast nearest neighbor search in SE(3) for sampling-based motion planning. *WAFR*, 2014, 197–214.
17. Ichnowski, J. and Alterovitz, R. Scalable multicore motion planning using lock-free concurrency. *IEEE Trans. Robotics 30*, 5 (2014), 1123–1136.
18. Ichter, B., Harrison, J. and Pavone, M. Learning sampling distributions for robot motion planning. *ICRA*, 2018, 7087–7094.
19. Janson, L., Ichter, B., and Pavone, M. Deterministic sampling-based motion planning: Optimality, complexity, and performance. *I. J. Robotics Res. 37*, 1 (2018), 46–61.
20. Janson, L., Schmerling, E., Clark, A.A. and Pavone M. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *I. J. Robotics Res. 34*, 7 (2015), 883–921.
21. Karaman, S. and Frazzoli, E. Sampling-based algorithms for optimal motion planning. *I. J. Robotics Res. 30*, 7 (2011), 846–894.
22. Kehoe, B., Patil, S., Abbeel, P., and Goldberg, K. A survey of research on cloud robotics and automation. *IEEE Trans. Automation Science and Engineering 12*, 2 (2015), 398–409.
23. Kleinbort, M., Salzman, O. and Halperin, D. Collision detection or nearest-neighbor search? On the computational bottleneck in sampling-based motion planning. *WAFR*, 2016.
24. Kunz, T., Thomaz, A. and Christensen, H. Hierarchical rejection sampling for informed kinodynamic planning in high-dimensional spaces. *ICRA*, 2016, 89–96.
25. LaValle, S.M. *Planning Algorithms*. Cambridge University Press, 2006.
26. LaValle, S.M. Motion planning: The essentials. *IEEE Robotics & Automation Mag. 18*, 1 (2011), 79–89.
27. LaValle, S.M., Branicky, M.S. and Lindemann, S.R. On the relationship between classical grid search and probabilistic roadmaps. *I. J. Robotics Res. 23*, 7-8 (2004), 673–692.
28. Li, Y., Littlefield, Z. and Bekris, K.E. Asymptotically optimal sampling-based kinodynamic planning. *I. J. Robotics Res. 35*, 5 (2016), 528–564.
29. Lynch, K.M. and Park, F.C. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017; http://hades.mech.northwestern.edu/ index.php/ Modern_Robotics.
30. Mandalika, A., Salzman, O. and Srinivasa, S.S. Lazy receding horizon A* for efficient path planning in graphs with expensive-to-evaluate edges. *ICAPS*, 2018, 476–484.
31. Murray, S., Floyd-Jones, W., Qi, Y., Sorin, D.J. Konidaris, G. Robot motion planning on a chip. *RSS*, 2016.
32. Plaku, E. and Kavraki, L.E. Quantitative analysis of nearest-neighbors search in high-dimensional sampling-based motion planning. *WAFR*, 2006, 3–18.
33. Salzman, O., Hemmer, M. and Halperin, D. On the power of manifold samples in exploring configuration spaces and the dimensionality of narrow passages. *IEEE Trans. Automation Science and Engineering 12*, 2 (2015), 529–538.
34. Schulz, A., Wang, H., Grinspun, E., Solomon, J. and Matusik, W. Interactive exploration of design trade-offs. *ACM Trans. Graph. 37*, 4 (2018), 131:1–131:14.
35. Solovey, K. and Kleinbort, M. The critical radius in sampling-based motion planning. *RSS*, 2018.
36. Solovey, K., Salzman, O. and Halperin, D. New perspective on sampling-based motion planning via random geometric graphs. *I. J. Robotics Res. 37*, 10 (2018), 1117–1133.
37. Stappen, A., Halperin, D. and Overmars, M.H. The complexity of the free space for a robot moving amidst fat obstacles. *Computational Geometry 3*, 6 (1993), 353–373.
38. Vernaza, P. and Lee, D.D. Learning and exploiting low-dimensional structure for efficient holonomic motion planning in high-dimensional spaces. *I. J. Robotics Res. 31*, 14 (2012), 1739–1760.
39. Xie, C., van den Berg, J.P., Patil, S. and Abbeel, P. Toward asymptotically optimal motion planning for kinodynamic systems using a two-point boundary value problem solver. *ICRA*, 2015, 4187–4194.
40. Zhou, B., Chiang, Y. and Yap, C. Soft subdivision motion planning for complex planar robots. *ESA*, 2018, 73:1–73:14.

Oren Salzman (osalzman@cs.technion.ac.il) is an assistant professor in the computer science department at the Technion–Israel Institute of Technology.

DOI:10.1145/3339824

Digital fabrication technologies open new doors—and challenges—for real-world support.

BY JENNIFER MANKOFF, MEGAN HOFMANN, XIANG 'ANTHONY' CHEN, SCOTT E. HUDSON, AMY HURST, AND JEEEUN KIM

Consumer-Grade Fabrication and Its Potential to Revolutionize Accessibility

PICTURE YOURSELF AT the recital of a 10-year-old boy, Wilbur (Figure 1 and featured on next page). Wilbur plays the cello beautifully. Like many of his peers, as he grows he needs to move to a larger instrument. However, unlike his peers, he also needs a new device with which to hold his cello bow. Wilbur is limb different, with a bow arm that ends just past his elbow. His family has worked hard to provide access to the best resources available: he has physical and occupational therapists and summer camp staff who are skilled at creating custom adaptations for him. However, creating an adaptation that works for Wilbur and his bow is difficult to do with existing prosthetics, which were designed for general tasks. His first cello-holding arm was patched together with rubber bands from a prosthetic. It was a start, but one he quickly outgrew. However, the best alternative specific to a stringed instrument was hinged in all the wrong places because it was designed for a violin.

Consider now a community of volunteers with 3D printers that can print complex, three-dimensional physical forms, with 3D modeling experience, and with an enormous capacity to donate their time and effort. This real-world grassroots community—e-NABLE (http://enablingthefuture.org)—consists of a diverse swath of people, from Boy and Girl Scout troops to university researchers, scattered across the world. e-NABLE innovators have 3D printed thousands of prosthetic hands for children. Two e-NABLErs, Drew Murray and Stephen Davies, created the first e-NABLE arm for children without a wrist. They collaborated with the authors to create a solution for Wilbur.

The power and potential of computational fabrication technologies to change the world is evident in this example and the many other solutions e-NABLErs have created for children and adults of all abilities. In fact, we

» key insights

- **Digital fabrication and craft enable people with disabilities to create assistive devices that meet their unique needs. This is valuable as a tool for co-design between researchers and people with disabilities and as a means toward a more accessible world for all.**
- **The creation of assistive technology is a multidisciplinary and collaborative effort. Beyond people with disabilities, we must support professional and personal caregivers who create and co-create assistive technology.**
- **Assistive technologies involve intimate devices often attached to the body or embedded in a personal environment. To match that value, digital fabrication must support a wider variety of materials, such as soft fabrics and strong metals.**

PHOTO BY ROB AMEND

are on the cusp of a radical change in the economy that is being driven by the advent of consumer-grade fabrication technologies. Just as content creation has progressed from languages such as HTML to advanced and easily used graphical user interfaces for website creation, so fabrication technologies will progress from today's complicated hobbyist technologies to user-friendly and ubiquitous techniques that alter daily life.

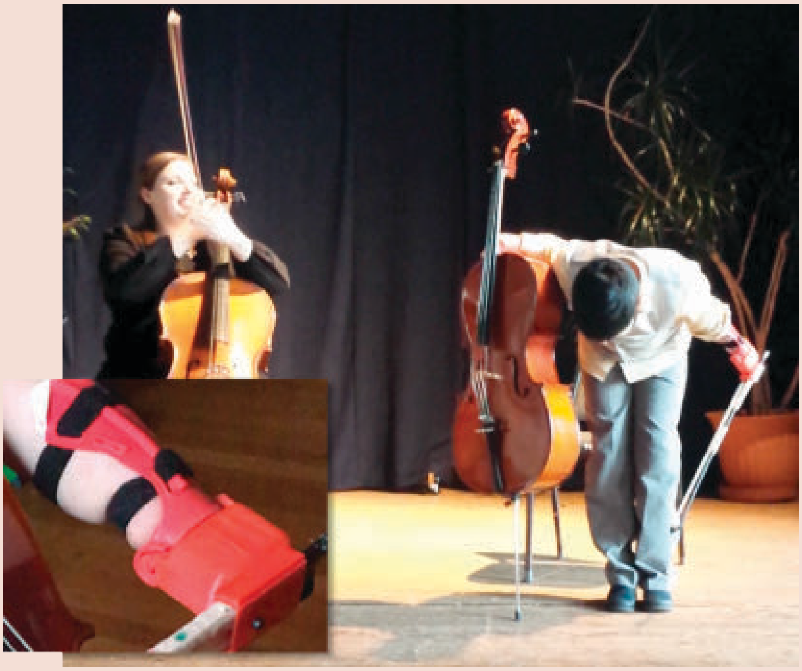
The progress toward consumer-grade manufacturing offers likely and still unforeseeable applications. Here, we are interested in its utility for changing who can access and produce assistive technology worldwide. We use the term assistive technology to reference devices that can increase the functional capacity of people with disabilities. While the etiology of disability varies greatly, its occurrence is constant, and the likelihood of experiencing a disability increases with age.

While disability need not be a barrier to employment, it significantly affects employability. As of 2016, only 17.5% of people with disabilities were in the U.S. workforce.[10] While many barriers faced by the disabled are sociological, others are structural or individual and have been addressed through design and computation. Studies have shown that website accessibility continues to be a significant challenge.[4]

Just as software automation can help to address some of online challenges, consumer-grade fabrication technologies can dramatically extend the power of non-experts to address structural issues in the physical world. For example, they can let fabricators create: tactile interfaces to digital[22] and physical objects,[12] maps of physical spaces,[40] and children's books.[38] They also help inexperienced designers build and customize their assistive technology,[5,15] increasing adoption and reducing costs.[19]

In this article, we first discuss applications of fabrication in the domain of assistive technology (AT) to highlight its potential value. We also review some

Figure 1. Wilbur at his recital.



challenges to the vision of consumer production of AT, such as the lack of a clinical perspective. While these are important problems, their resolution would be insufficient for the creation of fabricated AT without advances in fabrication research, as well. Our studies show that even the advent of low-cost, consumer-grade fabrication machines will not simplify the process of producing useful and usable AT artifacts.

While the vision of consumer-grade fabrication is intriguing, many challenges remain before it can be fully realized. Good design still requires engineering knowledge; the hardware used for fabrication is limited and difficult to operate; and the materials available are limited. Currently, rapid prototyping and personal scale fabrication are the domains of craftspeople and makers, but we expect this technology to democratize,[39] expanding the domain of fabrication from experts and enthusiasts to consumers. From maker spaces where consumers can gain expertise to 3D-printing firms that will manage the hardware for you, solutions are beginning to appear. However, our studies show that empowering consumers will require better tools, as well.

We discuss these challenges and approaches to overcome them. We conclude by defining barriers to 3D modeling that must be addressed for end users to produce practical, efficient objects. Framing consumer-grade fabrication technologies as tools for enabling accessibility presents unique and difficult technical challenges in terms of developing new materials, manufacturing processes, and design tools.

Fabricating Accessible Solutions

Assistive technology research has traditionally focused on two problem areas for people with disabilities: improving computer access, and improving access to the world through ubiquitous and now Internet of Things (IoT) technologies. However, as a field, it has only recently begun to assess the potential of fabrication technologies. Consumer-grade fabrication technologies can create a paradigm shift that will significantly improve both of these traditional domains. Grassroots efforts to use fabrication for these purposes have already appeared on the most popular 3D model sharing sites (Figure 2).[5] *This is not surprising given the importance of self-made AT historically in the disability community*, as described in Chen et al.[9] Most of these devices, however, were designed to interact with everyday objects with minimal, or no, mechanism or computation. The cello bow holder Wilbur uses is a fabricatable example that epitomizes many of these solutions in its high impact and relative simplicity.

While grassroots efforts have been effective, one of the most compelling aspects of fabrication technologies is the opportunity to further enhance AT production and use by leveraging computational power. Computation can enhance the set of things that can be created and broaden participation to include a wider set of producers.

Fabrication for computer access. One important opportunity for fabrication technology lies in making computers more accessible to blind users. While GUIs offered a paradigm shift for sighted users that enormously improved their interactive experience, they have made interaction more *difficult* for blind users. Even relatively simple tasks, such as Web browsing, which generally do not require mastering an entire windowing system, can take more than twice as long as they do for sighted interactions.[4] An alternative is to embody information in tangible form. This has proven valuable for spatial information[22] and contextual information.[37,38]

For example, we created a tangible scrollbar to convey information about content as blind users move its thumbs; its software updates the scrollbar as context switches.[2] We also embodied context in physical icons associated with a physical task switcher (see Figure 3). Together, these two techniques reduced task completion times in a simplified e-commerce task by over 50%.[2] Other tangible techniques, such as access overlays, also significantly cut task completion times.[22]

A growing research space explores how 3D printing can move both physical and audible information into a 3D-tactile space, offering new access conduits to blind users. Stangl et al.[38] used 3D printing as an artistic conduit for creating tactile picture books for blind children. Taylor et al.[40] applies 3D printing to the design and generation of tactile maps for blind navigation. Shi et al.[37] focuses more generally on how blind users interact with fabrication technology with respect to labeling models and creating them, respectively.

Figure 2. Example objects that address accessibility (http://Thingiverse.com).[5]

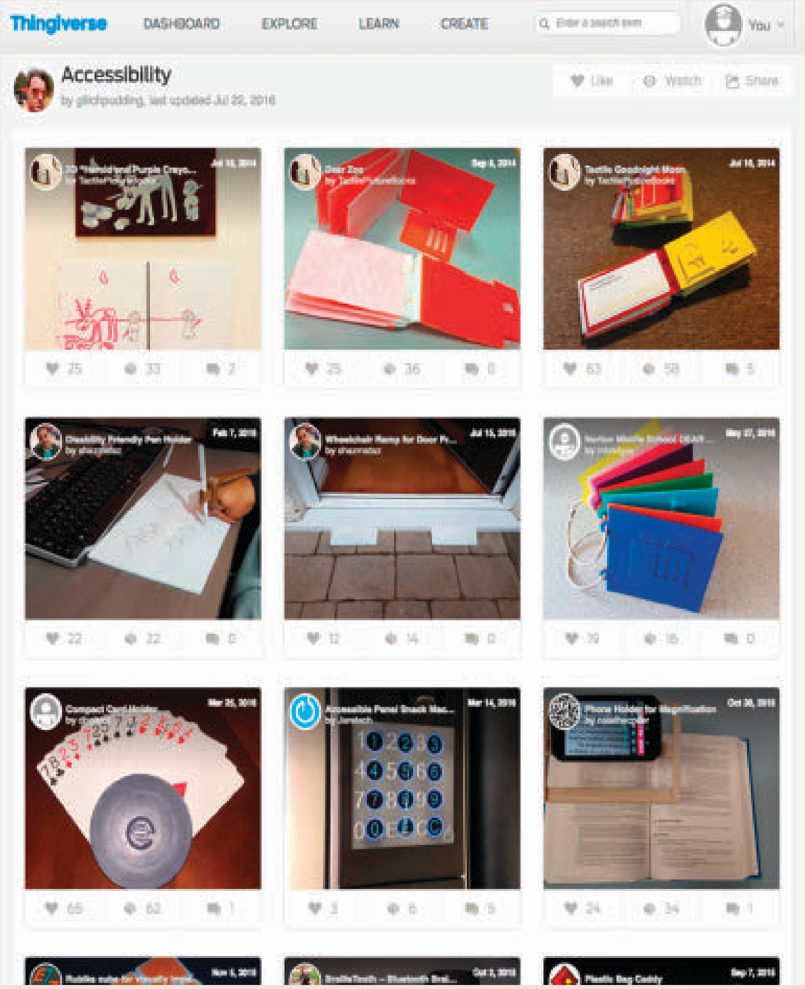
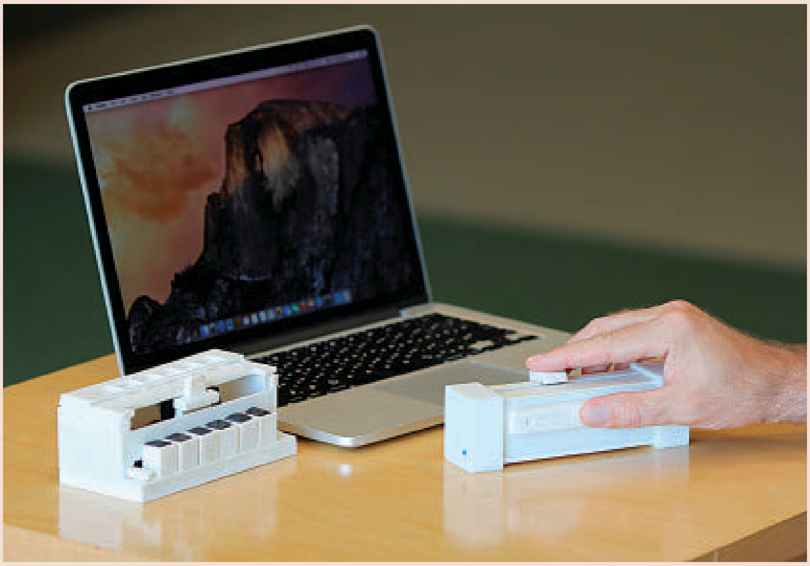


Figure 3. A tangible scrollbar and task switcher.[2]



We challenge the research community to further explore interaction alternatives and use them to develop tools that will improve access to desktop, Web, and mobile computing not just for the blind, but for those with other disabilities, as well. To do so, advances in underlying technologies for parsing, error correcting, and representing applications and their accessibility information are sorely needed.

Using 3D printing to create tactile representations of digital information benefits consumers of online content, but little existing research tackles the production of accessible physical content. We envision two avenues for research in this space. The first focuses on automatically converting existing content into tangible, accessible content in the vein of TactileMaps.net, used to generate portable physical representations of geographic data.[40] The second avenue for research focuses on fabrication of authoring tools that let producers design accessible physical content specifically for these modalities, as with physical augmentations that trigger audio playback.[37]

Fabricating access to the world. Much of the accessibility work now shared online focuses on improving access to the physical world rather than improving computer access.[5] Indeed, a long history of grassroots and craft-based creation of AT is summarized in Robitaille.[34]

When computation joins fabrication, powerful forms of customization become possible. For example, the advent of inexpensive touchscreen technology has led to at interface panels on appliances, reducing accessibility for the blind. While braille stickers are an option, not all blind people read braille, and such stickers can obscure labels for sighted people who share appliance use. The Facade application[12] uses a crowdsourcing pipeline to produce custom, semi-transparent tactile overlays for appliances. A Facade user first places a fiducial marker (a dollar bill) on the appliance near the control buttons and photographs it (using software designed to support photography by the blind). Next, crowd workers are asked to label the appliance buttons. Multiplexing this task among multiple crowd workers speeds completion times. Finally, Facade generates a custom 3D model with either braille or symbolic/text labels based in part on user-specified preferences. A home printer or commercial service can produce the final overlay, which can then be attached to the appliance (Figure 4). It is notable that the crowd workers in this process are not professionals or makers; they and the end user are not expected to have manufacturing skills, a true example of consumer-grade fabrication.

Broadening Participation in Production

The preceding examples demonstrate the value of fabrication in solving AT problems. Less visible, however, is the degree of *expertise* necessary to produce working solutions. Expertise may reside in a variety of stakeholders. For example, when trying to design a bow holder, Wilbur's family, teachers, and clinicians worked together at different times to try to solve his problem.[15] Although they all contributed valuable perspectives, they lacked the technical expertise needed to turn them into a solution.

Related to expertise is the *difficulty of designing new* devices. The traditional solution has been to create a single, high-cost generalist design that meets most needs and thus can be reused. This approach fails in cases such as Wilbur's specialized need (holding a cello bow). In addition, Wilbur has no interest in generalist devices since he usually chooses not to wear a prosthetic.

The notion that consumer-digital fabrication technologies will democratize the means of production has been explored and criticized from many perspectives. Tannenbaum et al. examine the overlap between "hedonistic technologies" and practical technologies in the context of 3D printing, suggesting that consumer-fabrication can benefit the fabricator from both emotional and economic perspectives.[39] Ames et al. criticize this framing, reflecting on how, in Western culture, corporate interests related to consumer-grade fabrication privilege certain stakeholders over others.[1] Lindtner et al. note that it is the design of CAD tools, primarily informed by HCI research, that can encourage a wider range of stakeholders to participate in consumer-grade fabrication.[25]

To bring these efforts to the AT space, specific tools and communities must be supported. Buehler et al.'s explorations of AT in the context of disability lays much of the groundwork for democratizing assistive technology design in disability-related contexts. Buehler et al. explored the 3D printed AT practices of nondomain experts on Thingiverse,[5] which highlights the gap between nonprofessional AT fabrication and traditional AT design spaces (such as educational and clinical practice). Buehler et al. developed recommendations for special-education maker spaces and their potential uses.[6,7] McDonald et al. used a similar approach to develop recommendations for physical therapists interested in adopting 3D printing into their clinical practice.[27] However, general-purpose CAD tools have not yet adapted to effectively support AT design.

The e-NABLE community could in principle address such issues. In prac-

Figure 4. Appliance façades.[12]

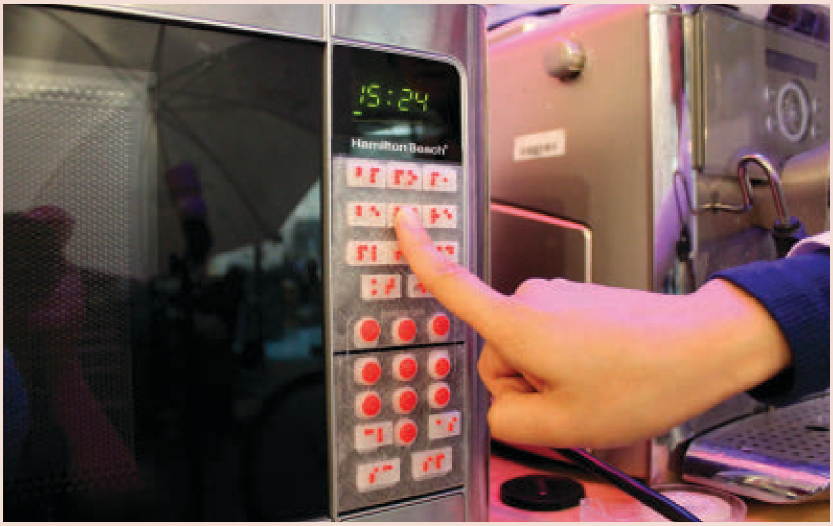


Figure 5. Clinicians, academics, and e-NABLErs working together to understand varied perspectives.[13]

tice, the difficulty of designing new devices creates a bottleneck for potential recipients. Only a few community members can design, as opposed to make or deliver, new devices.[30] Access to the few who can design ultimately helped to provide key parts of what Wilbur needed.

Design is difficult at many levels. For example, challenging design variables for those who attempted to create solutions for Wilbur's cello bow included: the optimal length of the holder (the total distance from shoulder to holder), the correct angle of the bow in its holder, the direction of the bow in its holder, the fit to Wilbur's arm, the degree of give in various directions, the ease of removing and replacing the bow, the ability to easily store the bow in his cello case (with something attached to it), and the materiality and durability of the bow holder. These are but a subset of all problems encountered when trying to understand or invent the best solution for Wilbur.

How do we ensure appropriate forms of stakeholder expertise are solicited? A workshop with clinicians and e-NABLE community members surfaced serious tensions between the clinical culture of do no harm and the e-NABLE culture of help where you can (Figure 5).[13] These tensions point to opportunities for collaboration, design process improvements among amateurs (including better follow-up and data collection), and new deployment models that include both clinical and community effort. Many clinicians push back on the inclusion of amateurs in the creation of AT, specifically prosthetic-like devices, because they fear that amateurs are unable to identify potential harms, let alone counteract them. Conversely, volunteer AT creators point out the harm of limiting access to devices when a clinician's time is expensive and scarce. Wilbur would not have access to a cello bow without help from the many stakeholders involved in creating his bow holder, but this is a notably minimal risk task, and the design process that worked in this scenario may not be generalizable.

Equally challenging is how to determine the level of expertise needed to express a solution using today's tools. Basic design capabilities taken for granted in computer science, such as reuse and modularity, are not supported. In addition, tools that streamline the engineering process, such as version control, are lacking. Finally, tools that empower non-experts are extremely limited, in part because of the lack of supporting capabilities such as those just described. In practice, creating models is so difficult that many end users are limited to 3D-printing models created by others.

These difficulties represent underlying challenges that are ripe for research and product advances. Here, we detail a few such opportunities, focusing in particular on the variety of materials and design tools available to fabricators.

To truly broaden the range of materials, we must consider new ways of printing.

Fabrication Materials and Machines

A wide variety of materials are available to end users for fabrication if we define 'materials' broadly to include crafting. For example, fiber arts—including knitting, crochet, felting, weaving, and sewing—are hugely popular, as evidenced by sites such as http://ravelry.com. Most use a range of natural and synthetic fibers. However, hobbyist crafting extends far beyond fiber arts; it includes a wide array of materials, from wood to metals to glass to ceramics, used to create beautiful, practical, and desirable objects of different sizes and types and increasingly leverages digital tools for some aspects of the process.

In contrast, consumer-grade printing is typically limited to about 200x200x-200mm (or less) and is primarily associated with two plastics: acrylonitrile butadiene styrene (ABS) and polylactic acid (PLA). Although the range of available materials is rapidly increasing (for example, flexible polymers and conductive materials), the basic method of construction used by most consumer-grade 3D printers requires having something that will melt, has the right viscosity when melted, will cool quickly, and will hold its form.

Thus, to truly broaden the range of materials, we must consider new ways of printing. Commercial alternatives, such as resin- or paste-based printing, are available to consumers. However, compared to the range of materials that most people associate with quality products and choose to touch and interact with daily (such as wood, silk, cotton, and stone), available 3D-printing materials are limited. Having a variety of materials is especially critical in the design of AT. These materials must be: *durable*, to withstand daily use over years; *comfortable and wearable* when touching the user's skin; and *sufficiently strong* to withstand the weight or strength of the user. 3D-printable filaments fall short on all of these criteria, limiting the scope of existing 3D-printed AT.

The next steps in consumer-grade material production do not necessarily require the wholesale redesign of 3D printers, though that may be part of the solution. For example, if we consider attempts to use fiber-based materials (such as cloth or yarn) in 3D printing, we can point to a range of solutions as exemplars, described here and illustrated in Figure 6.

New types of manufacturing technologies can be used (Figure 6a). For example, knitting machines are commercially available but not easily programmable. By making their capabilities more accessible, we would enable a new form of manufacturing to reach consumers. Doing so would require an underlying language and compiler that describes knittable objects in terms of shapes (sheets and tubes) instead of low-level knitting machine instructions.[26] Having such capabilities for AT could enable the construction of customized fabrics embeddable in clothing for people with disabilities.

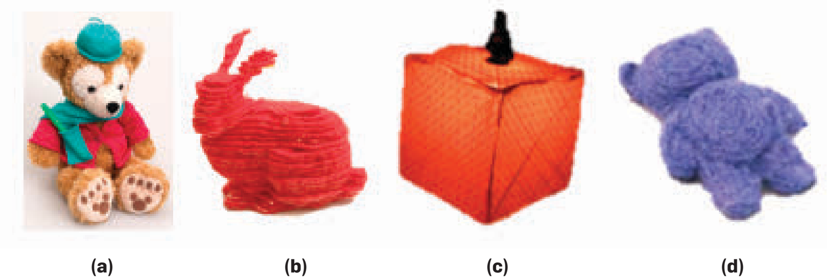
The printer itself could be redesigned. For example, it is possible to print in laser-cut layers of cloth (Figure 6b).[31] In this design, a roll of cloth is placed just below a surface to which it is held by suction. The partially printed form is laser cut out of the cloth. The print bed (with the partially printed object on it) is then raised and the suction released. When the print bed is lowered, a hot iron adheres the new layer to the one below it; the cloth must be prepared with appropriate glue on its under-side. Alternatively, the print head of a standard 3D printer could be modified to take a radical new approach (Figure 6d). For example, a consumer-grade 3D printer could feed wool yarn, instead of plastic, through a special print head that would adhere it to the print using a felting needle.[16] This would permit the creation of entirely fiber-based, printed soft objects, but it could also accommodate other materials.[16] By combining soft and hard, for example, we could potentially create an orthotic with soft materials where it touches the body, but hard materials for interacting with physical world objects.

Cloth could also be incorporated into a standard desktop 3D printer (Figure 6c).[33] For example, the printer could be paused to add a layer of cloth, or cloth could be adhered to the print bed and 3D printed upon. This would make larger scale 3D-printed objects possible, allow the creation of custom sensor shapes, and enable rapid prototyping, among other benefits.[33] In the AT space, printing cloth has numerous applications, from creating soft- and large-scale mechanisms to advancing the mixed material properties of tactile aids, such as picture books.[38]

This series of examples illustrates a range of approaches for expanding the set of consumer-fabricatable materials.

Figure 6. Four examples of cloth combined with fabrication.

(a) A knitting machine compiler is used to make clothing for a teddy bear.[26] **(b)** A rabbit is printed in layers of cloth.[31] **(c)** A desktop 3D printer is used to print on cloth to create new types of objects, such as this lampshade.[33] **(d)** A desktop printer is modified to print using felt.[16]



However, improving the accessibility and viability of consumer-grade printing for additional materials remains an open problem. Metals, wood, glass and ceramics are all materials that consumers might like to use for printing. Each poses a challenge to automation.

Additionally, better consumer-grade production pipelines are needed. For example, to fully leverage knitting machines, a pipeline might encompass pattern design or selection, modification based on scanned or measured properties of a real-world object (such as the shape of a body or a limb difference), verification of printability, and production. As we describe next, such pipelines will require changes not only in the manufacturing technologies available to consumers, but in the design software available to them as well.

Fabrication Design Tools

The design tools available to today's home hobbyists are fairly basic. They consist most often of a pen, pencil, and deep knowledge of a craft, sometimes supplemented by easily available designs that can be reused or modified. In contrast, extremely powerful design software is available for creating the input files used by 3D printers. However, we maintain this software is not well suited to the needs or abilities of the types of people who might use consumer-grade fabrication technologies. Further, this software is not well suited for many of the stakeholders who create AT: clinicians, teachers, peers, and family.

This section focuses on problems that home hobbyists (including do-it-yourself AT creators) encounter, none of which is streamlined by existing software. These problems generally stem from the fact that functional objects must engage in some way with the real world. Further, the design process, which extends from conception to assembly, requires study to identify and understand where and how end users encounter difficulties. Some examples of unexpected end user challenges in the design of functional objects include measurement (and its potential for error), attachment or interface with the world, modification or adaptation (particularly important for building AT). Finally, to be effective on a large scale, we believe that designs must encode information specific to reuse to be easily modified for new contexts. We discuss each challenge.

Measurement. When a model must conform to a specific real-world goal after it is 3D printed, it is important that the goal be precisely specified. However, *measurement errors* pose a significant yet often overlooked challenge for end users, as determined by a systematic study of the sources and types of such errors.[23]

Kim et al. found that user error (such as misaligning instruments and misreading units), measurement instrument precision, and even task definition made measurement error common.[23] Figure 7 depicts some examples of faulty measurements from the study. Compounding these errors is the fact that 3D printing itself is not perfectly precise. For example, some materials shrink slightly as they cool. Thus, measurements are at best approximations that contain some degree of *uncertainty*. A model robust to this uncertainty will be less likely to fail.

Measurement error can be addressed at the prototyping stage using mixed design approaches that incorporate simple materials such as foam or Lego Bricks.®[28] This same approach has been successful at facilitating experimentation and iteration in AT design.[15] An example is our iterative design of the cello bow using Lego Bricks to estimate length (Figure 8).

Design strategies can accommodate measurement error, as illustrated in Figure 9. For example, by inserting a flexible buffer around an uncertain real-world object, small differences would no longer require reprinting. A related (and synergistic) strategy could support the replacement of small areas of a 3D-printed object likely to have errors. This would reduce cost and waste because that region could be reprinted and then connected with a snap joint, adhesive, or other method. Innovation is needed to further expand this set of methods and develop robust automatic tools for applying them in a wide range of contexts.

Figure 7. Some examples of inaccurate measurement practices.[23]

(a) The tick mark on the paper is not aligned with the end of the phone for measuring phone length. **(b)** A ruler is not an ideal way to measure angles accurately. **(c)** The width of the light bulb's base is difficult to estimate, and the task is ill defined (that is, should the outside or the inside of the threads be measured?).

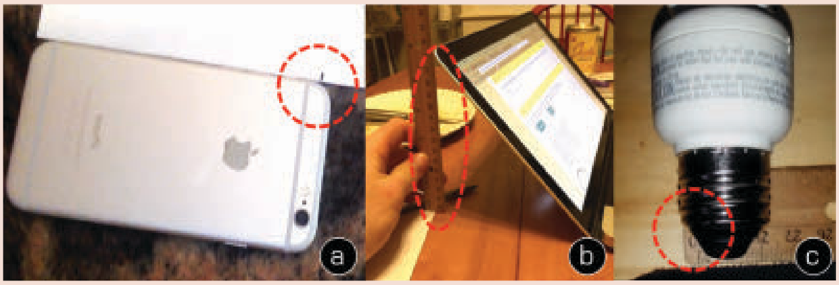


Figure 8. Prototyping the length of a cello bow holder (inset shows final result). This length was challenging to determine due to the lack of a physical object to measure and physiological subtleties in finding the right length for the dynamic activity of cello playing.[15]

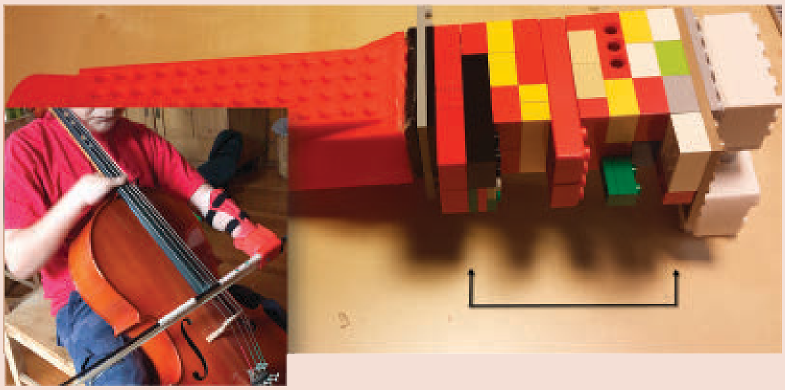
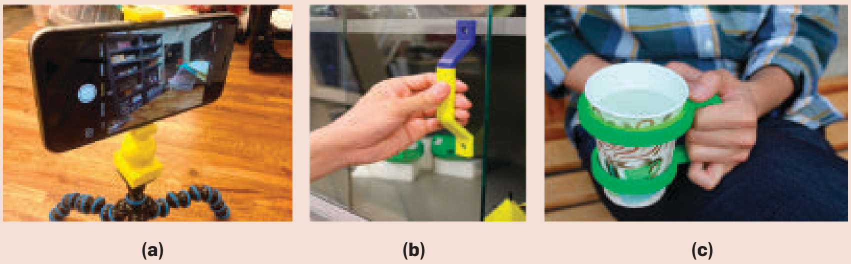


Figure 9. Some examples of objects designed for measurement uncertainty.

(a) This tripod's angle can be adjusted. **(b)** Part of this handle can be replaced to resize it. **(c)** This cup holder has buffers in it. For most of these objects, flexibility in the face of error also affords new flexibility in use (for example, the cup holder can fit many cups).[23]



Attachment. For functional objects to be useful, they must typically interact in some way with real-world objects (people or items to be extended, manipulated, or repaired). Interaction, in turn, typically requires attachment, the temporary or permanent connection of two or more objects. Thus, the problem of attaching 3D-printed object to a real world one must be addressed.

Attachment has been explored extensively outside the domain of 3D printing. Material properties, strength, usability, and aesthetics must all be considered when attaching objects.

The issue is sufficiently complex to support websites such as ThisToThat (Glue Advice),[a] which help answer questions about how to connect two objects with glue.

In the domain of 3D printing, incorporating existing objects is also important. Incorporating Lego Bricks can

[a] http://www.thistothat.com/

speed up a print by reducing the amount of printed material.[28] To improve 3D printing interactivity, a way to design for embedded electronics is needed.[36] Jones et al. approached this by combining sculpting and modular interaction toolkits to prototype interactive sculptures.[21] Alternatively, 3D printers could produce a new facade supporting alternative interactions for existing physical interfaces (for example, Ramakers et al.[32]).

These examples do not specifically address or provide control over how the 3D-printed object should be attached to the real-world object it is modifying. A set of attachment methods could provide a basis for exploring and modifying alternatives. Several challenges arise when attaching objects:

Collision. If an object is on the print bed (to be printed on or through), the design of the attachment must ensure no collisions occur between the print head and object. A design tool can detect and visualize potential collisions to help the user determine a viable position for the attached component.

Insertion. Specifically when printing through an object, there must be a viable insertion path for the object. Such a path can be estimated using a reverse gravity model (that is, if the object can easily fall out when inverted, it can easily be placed when in normal orientation).

Durability. Strength or durability of the attachment can be influenced by the size of the connection (a very small footprint connecting two objects is less secure), the object's flatness, and the direction and area of force applied to the attached object.

Semantics. At a higher level, the intended use can influence the effectiveness of an attachment. For example, balance, direction of hold (for a handle) and cost might be concerns that influence an effective attachment technique.

Automated tools such as Autoconnect[24] help address these challenges by creating customized connectors, which take into account the position and weight of the objects being connected. Interactive tools such as Encore system[8] can support exploration of potential attachment techniques and visualize the effectiveness of attachment over a possible set of metrics (Figure 10). Further research is needed to determine the best metrics, and the best way to express those metrics computationally.

An open area for future investigation is how to develop tools that function in real-world settings where an object to be modified may not portable or is too large to be brought into a scanner or 3D printer. This requires the high-quality, low-cost capture of real-world object models and ideally the ability to convert them to high-quality models. One approach to this problem is to make 3D printing more portable.[35] Another challenge is rethinking CAD in an object-centric fashion, meaning that models would be designed with respect to an existing object.

While attachment is a basic capability needed for many 3D printed objects to be functional, it is only the first step. For an end user, the design of the *function* is at least as difficult as the design of the attachment. Sample tools that address this problem include Grafter,[35] RetroFab,[32] and Reprise[9] (Figure 12).

Adaptation and reuse. Bridging the gap between geometry and function presents a substantial challenge, even for experienced users. A powerful way to bridge this gap permits the work of experienced designers to be easily adapted and reused by others. Many resources for 3D-printable designs have been extensively studied; they show that adaptation of existing designs is often trivial but rarely improves on the original designs.[29] However, CAD tools and the models they produce, while general and powerful, are not necessarily designed with reuse in mind. Functional information implicit in an object's geometric form is never expressed explicitly; hence, it is inaccessible to anyone who is not also sufficiently skilled to recognize the underlying mechanical rationale.

Modelers would benefit from the equivalent of an end user programming tool and a set of abstractions for encoding design information in an interactively accessible way. This is what the Parameterized Abstractions of Reusable Things (PARTs) framework provides; it puts advanced methods for capturing 3D modeling design intent into the hands of non-expert modelers.[14] Doing so supports reuse, experimentation, and sharing.

Figure 10. Encore visualizes attachment goodness using a heat map.[8]

Three different metrics are shown: (left) Viability for printing; (center) Likely durability based on curvature; (right) Estimated usability based on the assumption that balance will be better in areas near to the center of mass (This assumes the forces applied have the same direction as the surface normals).

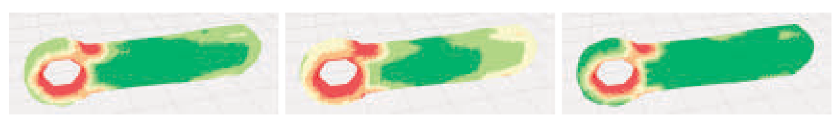
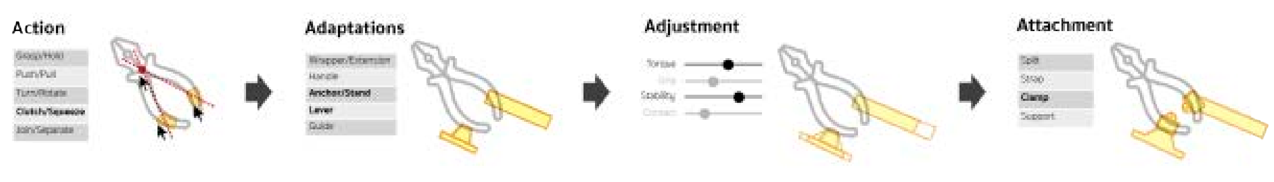


Figure 11. The Reprise workflow assumes the existence of a model of the object to be adapted.

It starts with a specification of the type of action to be supported. Each action has a set of associated adaptations, from which the user picks. Reprise generates an appropriate model adaptation. Parameters of the adaptation can then be adjusted. Finally, an attachment method is selected.[9]



PARTs' basic abstraction is *functional geometry*. Functional geometry incorporates the modern programming concept of classes, which encapsulate data and functionality, making it easier to validate and mutate data, manage complexity, and support modularity. Specifically, it includes *assertions* that test whether a model is used correctly and *integrators* that mutate the larger design context. These abstractions are available in an interactive graphical form and increase model usability and reusability.

The PARTs framework can flexibly address a wide variety of 3D modeling challenges for non-experts. It supports many tasks in-situ that are normally handled in separate dialogues or tools that non-experts may not find or understand how to use. While PARTs intentionally uses simple concepts, many model-specific design goals can be encapsulated using its assertions and integrators.

One of the benefits of PARTs is its generality. Reprise and Encore represent tools that provide specific, carefully constructed solutions to important problems. However, they were each created outside the traditional 3D-modeling context (CAD tools). In contrast, PARTs is integrated into the professional Autodesk Fusion360 CAD tool.

Figure 12. Some examples of objects generated using Reprise and the original designs from our survey that inspired them (shown in insets). Clockwise from top left: a wrapper for a fork; a lever for controlling a spray bottle; an anchor for using a tool with one hand; a handle for a key.

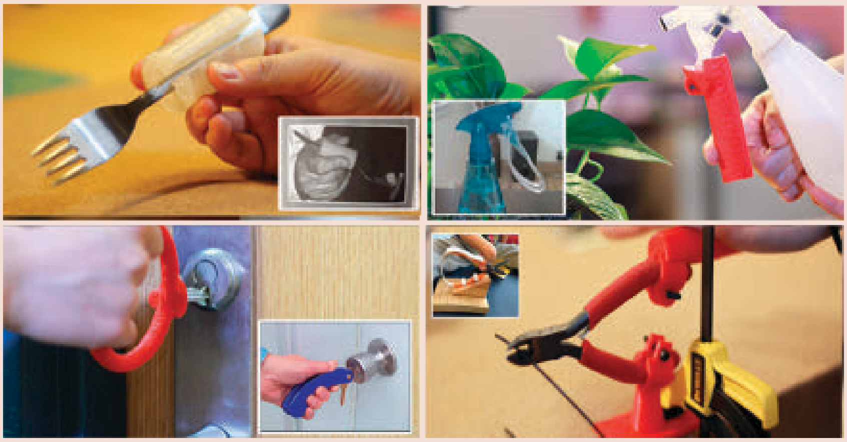
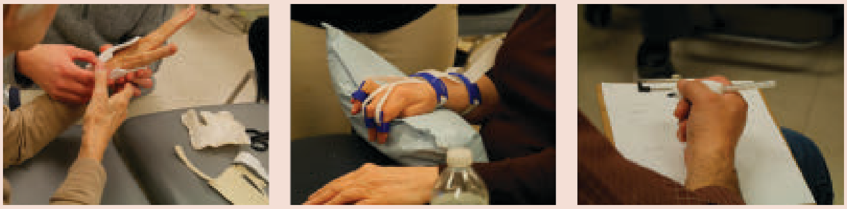


Figure 13. 3D printed assistive technologies designed by Physical Therapy graduate students using clay that was later 3D scanned and printed.[27] These include a wrist brace (left) hand spreader (center) and pencil grip (right).



Discussion and Future Work

The body of work we described in this article outlines the beginning of a path for empowering end users to design pleasing and functional assistive technologies to share with others. Despite the many opportunities for consumer-grade, desktop 3D printers to solve accessibility challenges, we have observed few examples of end users adopting these solutions. This slow adoption is troubling, and we must delve deeper to understand the impact of fabrication on assistive technology. Here, we discuss the stakeholders currently engaged in AT production and the barriers they face in adopting fabrication technology. We then discuss the importance of understanding use and abandonment of AT that has been 3D printed. Finally, we examine potential models for the sustainable production and personalization of digitally fabricating assistive technology.

Stakeholders. To fully understand the potential for new fabrication technologies to transform AT use, we must understand the stakeholders involved in the design, production, and use of do-it-yourself (DIY) AT. Most existing research in this area focuses on the design of the AT, DIY-AT end users, and volunteers who help with fabrication. Studies have explored AT's efficacy[3,19] and the potential for people with disabilities to participate more directly in AT fabrication.[15,18] The volunteer communities that supports DIY-AT have also been thoroughly studied, both by reviewing the artifacts they produce[5,9] and by interviewing members of communities such as e-NABLE.[30] These communities tend to be dominated by people with strong STEM backgrounds and education. This lack of diversity reveals opportunities to expand who can be a maker, particularly in the AT context.

In contrast, many of the stakeholders involved in the more traditional AT ecosystem do not have a STEM background. These may include educators, clinicians, family, and students (for example, Buchler et al.[6,7]) and physical therapists (for example, Hofmann et al.[13] and McDonald et al.[27]). Further study is needed to explore how best to support these stakeholders.

Our own plans include teaching physical therapists to use fabrication tools, expanding on the methodologies developed by McDonald et al.[27] and Buehler et al.,[6] who had therapists design assistive technology using clay that is later 3D scanned. Figure 13 depicts custom AT recently co-designed by older adults and physical therapy graduate students.

In contrast to DIY-AT communities, much less is known about AT making in medical settings. Clinicians are using fabrication for more than just assistive technology. For example, MakerNurse[b] is an organization that supports nurses who fabricate technology to improve patient care. Similarly, the U.S. Veteran's Administration (VA) has been fostering multiple internal efforts to use 3D-printing technologies.[c]

It is unclear whether the current state of consumer-grade fabrication is fit to meet the needs of these new clinical stakeholders. Perhaps more importantly, we do not know exactly what the needs of these stakeholders will be. More research into a culture of medical making is needed, beginning with studies of existing clinical practice and the perspectives of those clinicians. Research should focus on understanding existing clinical practice around "making," how this is currently taught, and the perspectives of clinicians toward digital fabrication tools.

Abandonment and adoption. Abandonment rates for assistive technology are very high.[19] Consumer-grade fabrication technologies may reduce AT abandonment. Hurst and Tobias[19] note that AT users find it empowering to create their own AT, which makes them more likely to continue using it. However, communities such as e-NABLE that are deploying 3D-printed assistive technology lack sufficient information. It is unclear whether or not the devices produced by e-NABLE meet U.S. medical device standards.[3]

Perhaps one AT issue with long-term success is that its creation is not a complete solution to any problem. Things break and needs change, yet follow up is not baked into the system when we leave the clinic. Worse, volunteers may move on, or otherwise be unavailable when follow up is requested. To better understand these challenges, we are interviewing e-NABLE device recipients about topics such as knowledge transfer across volunteers. We predict that careful documentation and knowledge transfer will continue to be a challenge in volunteer communities and may also be a challenge for clinics and clinicians who experience high turnover or limited availability.

One advantage of 3D-printed AT is the potential to support a more data-driven process than traditional assistive technology. For example, it is possible to embed sensing capabilities during printing.[20] Inexpensive, reliable sensing represents an unparalleled opportunity to collect usage data at scale, and to study the varied circumstances under which AT can be successful. Further, real-time data collection to support volunteers and clinicians by providing alerts when abandonment is predicted, or help provide information that can be used to support discussion of what is working and what is not.

One advantage of 3D-printed AT is the potential to support a more data-driven process than traditional assistive technology.

Production and personalization. As a result of improved understanding of AT abandonment or acceptance and the many stakeholders in assistive technology creation, we can improve the production and personalization of AT with consumer-grade fabrication techniques.

One key consideration for using consumer-grade fabrication to produce AT is who actually runs the printer and where is the printer situated. Volunteer AT models situate the printer in the home of a volunteer or a person with a disability; the printing is done by the volunteer or rarely the person with a disability. However, this model excludes many potential AT users who may not have access to maker technologies or a skilled volunteer.

Instead, with improved understanding of the stakeholders in the traditional healthcare system, people with disabilities could access customized AT through healthcare providers, assuming an effective infrastructure is in place. This raises new questions: Which types of clinicians should use 3D printing? What educational resources need to be available to clinicians to start fabricating? Should clinicians run the printers? Perhaps fabrication technicians, situated in a pharmacy setting, should fill AT "prescriptions" instead. Alternatively, should medical maker spaces be built into clinics and hospitals?

As the demand for 3D-printed AT grows, so does the potential to create new technical jobs in digital fabrication design and fabrication. Given the wide scope of design tools, 3D-printer technologies, and materials there are opportunities for high-tech careers and many entry-level technical jobs. We think a sustainable and cost-effective solution for the de-

[b] http://makernurse.com/

[c] https://www.innovation.va.gov/

ployment of 3D-printed AT may be to outsource the fabrication. In our research with physical therapy graduate students (Figure 13), the 3D scanning and printing was performed by local high school students who were working in a nearby 3D print shop.[11,17] In addition to efficiently fabricating the assistive technologies, this was a valuable and meaningful experience for these young adults who were working in their first technical job.

Conclusion

The promise of 3D printing and other digital fabrication technology lies in its ability to create custom, relevant solutions to real-world problems. The ability to produce customized objects offers transformative potential for new assistive technology that must be customized to meet an individual's current abilities. In order to reach this potential we must create powerful, flexible, and inclusive design tools. When these tools meet the needs of the variety of stakeholders impacted by the production of assistive technology, we will have the potential to empower and increase participation for all.

References

1. Ames, M.G., Bardzell, J., Bardzell, S., Lindtner, S., Mellis, D.A., and Rosner, D.K. Making cultures: Empowerment, participation, and democracy–Or not? In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*. pages 1087–1092. ACM, 2014.
2. Baldwin, M.S., Hayes, G.R., Haimson, O.L., Mankoff, J., and Hudson, S.E. The tangible desktop: A multimodal approach to nonvisual computing. *ACM Trans. Accessible Computing 10*, 3 (2017), 9.
3. Bennett, C.L., Cen, K., Steele, K.M., and Rosner, D.K. An intimate laboratory? Prostheses as a tool for experimenting with identity and normalcy. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2016, 1745–1756.
4. Bigham, J.P., Cavender, A.C., Brudvik, J.T., Wobbrock, J.O., and Ladner, R.E. WebinSitu: A comparative analysis of blind and sighted browsing behavior. In *Proceedings of the 9th Intern. ACM SIGACCESS Conference on Computers and Accessibility*. ACM, 2007, 51–58.
5. Buehler, E., Branham, S., Ali, A., Chang, J.J., Hofmann, M.K., Hurst, A., and Kane, S.K. Sharing is caring: Assistive technology designs on Thingiverse. In *Proceedings of the 33rd Annual ACM Conf. on Human Factors in Computing Systems*. ACM, 2015, 525–534.
6. Buehler, E., Comrie, N., Hofmann, M., McDonald, S., and Hurst, A. Investigating the implications of 3D printing in special education. *ACM Trans. Access. Comput. 8*, 3 (Mar. 2016), 11:1–11:28.
7. Buehler, E., Kane, S.K., and Hurst, A. ABC and 3D: Opportunities and obstacles to 3D printing in special education environments. In *Proceedings of the 16th Intern. ACM SIGACCESS Conf. Computers & Accessibility*. ACM, 2014, 107–114,
8. Chen, X., Coros, S., Mankoff, J., and Hudson, S.E. Encore: 3D printed augmentation of everyday objects with printed-over, affixed and interlocked attachments. In *Proceedings of the 28th Annual ACM Symp. User Interface Software & Technology*. ACM, 2015, 73–82.
9. Chen, X., Kim, J., Mankoff, J., Grossman, T., Coros, S., and Hudson, S.E. Reprise: A design tool for specifying, generating, and customizing 3D printable adaptations on everyday objects. In *Proceedings of the 29th Annual Symp. User Interface Software and Technology*. ACM, 2016, 29–39.
10. Persons with a Disability: Labor Force Characteristics—2016. News Release USDL-17-0857. U.S. Department of Labor: Bureau of Labor Statistics. June 2017.
11. Easley, W., Hamidi, F., Lutters, W.G., and Hurst, A. Shifting expectations: Understanding youth employees' handoffs in a 3D print shop. In *Proceedings of ACM Hum.-Comput. Interact.* (CSCW). Nov. 2018, 47:1–47:23.
12. Guo, A., Kim, J., Chen, X., Yeh, T., Hudson, S.E., Mankoff, J., and Bigham, J.P. Facade: Auto-generating tactile interfaces to appliances. In *Proceedings of the 2017 Conf. Human Factors in Computing Systems*. ACM, 2017, 5826–5838.
13. Hofmann, M., Burke, J., Pearlman, J., Fiedler, G., Hess, A., Schull, J., Hudson, S.E., and Mankoff, J. Clinical and maker perspectives on the design of assistive technology with rapid prototyping technologies. In *Proceedings of the 18th Intern. ACM SIGACCESS Conf. Computers and Accessibility*. ACM, 2016, 251–256.
14. Hofmann, M., Hann, G., Hudson, S.E., and Mankoff, J. Greater than the sum of its PARTs: Expressing and reusing design intent in 3D models. In *Proceedings of the 2018 Conf. Human Factors in Computing Systems*. ACM, 2018, 301:1–301:12.
15. Hofmann, M., Harris, J., Hudson, S.E., and Mankoff, J. Helping hands: Requirements for a prototyping methodology for upper-limb prosthetics users. In *Proceedings of the 2016 Conf. Human Factors in Computing Systems*. ACM, 2016, 1769–1780.
16. Hudson, S.E. Printing teddy bears: A technique for 3D printing of soft interactive objects. In *Proceedings of the Conf. Human Factors in Computing Systems*. ACM, 2014, 459–468.
17. Hurst, A., Grimes, S., McCoy, D., Carter, N., Easley, W., Hamidi, F., and Salib, G. Lessons learned creating youth jobs in an afterschool maker space. In *Proceedings of the 2017 ASEE Annual Conf. & Exposition*. (Columbus, OH, June 2017).
18. Hurst, A. and Kane, S. Making 'making' accessible. In Proceedings of the 12th Intern. Conf. Interaction Design and Children. ACM, 2013, 635–638.
19. Hurst, A. and Tobias, j. Empowering individuals with do-it-yourself assistive technology. In *Proceedings of the 13th Intern. ACM SIGACCESS Conf. Computers and Accessibility*. ACM, 2011, 11–18.
20. Iyer, V., Chan, J., Culhane, I., Mankoff, J., and Gollakota, S. Wireless analytics for 3D printed objects. In *Proceedings of the 31st Annual ACM Symp. User Interface Software and Technology*. ACM, 2018, 141–152.
21. Jones, M.D., Seppi, K., and Olsen, D.R. What you sculpt is what you get: Modeling physical interactive devices with clay and 3D printed widgets. In *Proceedings of the 2016 Conf. Human Factors in Computing Systems*. ACM, 2016, 876–886.
22. Kane, S.K., Morris, M.R., Perkins, A.Z., Wigdor, D., Ladner, R.E., and Wobbrock, J.O. Access overlays: improving non-visual access to large touch screens for blind users. In *Proceedings of the 24th Annual ACM Symp. User Interface Software and Technology*. ACM, 2011, 273–282.
23. Kim, J., Guo, A., Yeh, T., Hudson, S.E., and Mankoff, J. Understanding uncertainty in measurement and accommodating its impact in 3D modeling and printing. In *Proceedings of the 2017 Conf. Designing Interactive Systems*. ACM, 2017, 1067–1078.
24. Koyama, Y., Sueda, S., Steinhardt, E., Igarashi, T., Shamir, A., and Matusik, W. Autoconnect: Computational design of 3D-printable connectors. *ACM Trans. Graphics 34*, 6 (2015), 231.
25. Lindtner, S., Bardzell, S., and Bardzell, J. Reconstituting the utopian vision of making: HCI after technosolutionism. In *Proceedings of the 2016 Conf. Human Factors in Computing Systems*. ACM 2016, 1390–1402.
26. McCann, J., Albaugh, L., Narayanan, V., Grow, A., Matusik, W., Mankoff, J., and Hodgins, J. A compiler for 3D machine knitting. *ACM Trans. Graphics 35*, 4 (2016), 49.
27. McDonald, S., Comrie, N., Buehler, E., Carter, N., Dubin, B., Gordes, K., McCombe-Waller, S., and Hurst, A. Uncovering challenges and opportunities for 3D printing assistive technology with physical therapists. In *Proceedings of the 18th Intern. ACM SIGACCESS Conf. Computers and Accessibility*. ACM, 2016, 131–139.
28. Mueller, S., Mohr, T., Guenther, K., Frohnhofen, J., and Baudisch, P. Fabrickation: Fast 3D printing of functional objects by integrating construction kit building blocks. In *Proceedings of the 32nd Annual ACM Conf. Human Factors in Computing Systems*. ACM, 2014, 3827–3834.
29. Oehlberg, L., Willett, W., and Mackay, W.E. Patterns of physical design remixing in online maker communities. In *Proceedings of the 33rd Annual ACM Conf. Human Factors in Computing Systems*. ACM, 2015, 639–648.
30. Parry-Hill, J., Shih, P.C., Mankoff, J., and Ashbrook, D. Understanding volunteer AT fabricators: Opportunities and challenges in DIY-AT for others in e-NABLE. In *Proceedings of the 2017 Conf. Human Factors in Computing Systems*. ACM, 2017, 6184–6194.
31. Peng, H., Mankoff, J., Hudson, S.E., and McCann, J. A layered fabric 3D printer for soft interactive objects. In *Proceedings of the 33rd Annual ACM Conf. Human Factors in Computing Systems*. ACM, 2015, 1789–1798.
32. Ramakers, R., Anderson, F., Grossman, T., and Fitzmaurice, G. Retrofab: A design tool for retrofitting physical interfaces using actuators, sensors and 3D printing. In *Proceedings of the 2016 Conf. Human Factors in Computing Systems*. ACM, 2016, 409–419.
33. Rivera, M.L., Moukperian, M., Ashbrook, D., Mankoff, J., and Hudson, S.E. Stretching the bounds of 3D printing with embedded textiles. In *Proceedings of the 2017 Conf. Human Factors in Computing System*. ACM, 2017, 497–508.
34. Robitaille, S. *The Illustrated Guide to Assistive Technology and Devices: Tools and Gadgets for Living Independently: Easyread Super Large 18-pt Edition*. ReadHowYouWant.com, 2010.
35. Roumen, T.J., MÃjller, W., and Baudisch, P. Grafter: Remixing 3D-printed machines. In *Proceedings of the 2018 Conf. Human Factors in Computing Systems*. ACM, 2018, 63:1–63:12.
36. Savage, V., Follmer, S., Li, J., and Hartmann, B. Makers' marks: Physical markup for designing and fabricating functional objects. In *Proceedings of the 28th Annual ACM Symp. User Interface Software & Technology*. ACM, 2015, 103–108.
37. Shi, L., Zhao, Y., and Azenkot, S. Markit and Talkit: A low-barrier toolkit to augment 3D printed models with audio annotations. In *Proceedings of the 30th Annual ACM Symp. User Interface Software and Technology*. ACM, 2017, 493–506.
38. Stangl, A., Hsu, C.L., and Yeh, T. Transcribing across the senses: Community efforts to create 3D printable accessible tactile pictures for young children with visual impairments. In *Proceedings of the 17th Intern. ACM SIGACCESS Conf. Computers & Accessibility*. ACM, 2015, 127–137.
39. Tanenbaum, J.G., Williams, A.M., Desjardins, A., and Tanenbaum, K. Democratizing technology: Pleasure, utility and expressiveness in DIY and maker practice. In *Proceedings of the 2013 Conf. Human Factors in Computing Systems*. ACM, 2013, 2603–2612.
40. Taylor, B., Dey, A., Siewiorek, D., and Smailagic, A. Customizable 3D printed tactile maps as interactive overlays. In *Proceedings of the 18th Intern. ACM SIGACCESS Conf. Computers and Accessibility*. ACM, 2016, 71–79.

Jennifer Mankoff is Richard E. Ladner Professor in the Paul G. Allen School of Computer Science and Engineering at the University of Washington, Seattle, USA.

Megan Hofmann is a Ph.D. student at Carnegie Mellon University's HCI Institute, Pittsburgh, PA, USA.

Xiang 'Anthony' Chen is an assistant professor at the University of California, Los Angeles, CA, USA.

Scott E. Hudson is a professor at Carnegie Mellon University's HCI Institute, Pittsburgh, PA, USA.

Amy Hurst is an associate professor of Human-Centered Computing at New York University, NY, USA.

Jeeeun Kim is is a Ph.D. student at the University of Colorado, Boulder, CO, USA.

DOI:10.1145/3338124

Protein design algorithms can leverage provable guarantees of accuracy to provide new insights and unique optimized molecules.

BY MARK A. HALLEN AND BRUCE R. DONALD

Protein Design by Provable Algorithms

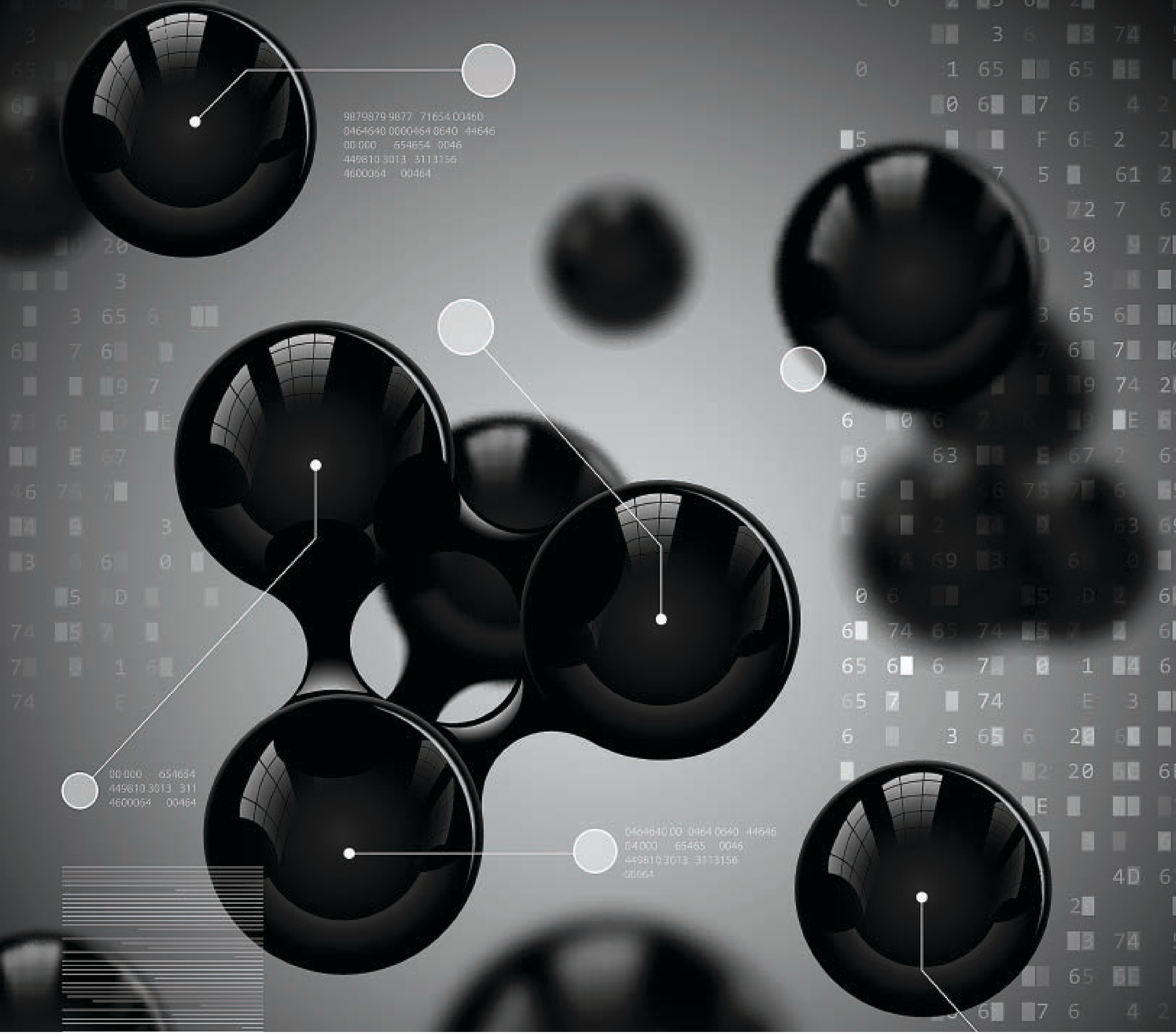
PROTEINS ARE A class of large molecules that are involved in the vast majority of biological functions, from cell replication to photosynthesis to cognition. The chemical structure of proteins is very systematic[5]—they consist of a chain of atoms known as the *backbone*, which consists of three-atom (nitrogen-carbon-carbon) repeats known as *residues*, each of which features a *sidechain* of atoms emanating from the first carbon. In general, there are 20 different options for sidechains, and a residue with a particular type of sidechain is known as an *amino acid* (so there are also 20 different amino acid types). For billions of years, the process of evolution has optimized the sequence of amino acids that make up naturally occurring proteins to suit the needs of the organisms that make them. So we ask: Can we use computation to design non-naturally occurring proteins that suit our biomedical and industrial needs?

This question is a combinatorial optimization problem, because the output of a protein design computation is a sequence of amino acids. Due to the vast diversity of naturally occurring proteins, it is possible—and very useful—to begin a protein design computation with a naturally occurring protein and then to modify it to achieve the desired function. In this article, we focus on protein design algorithms that perform this optimization using detailed modeling of the 3D structure of the protein.[5,8] Thus, they will begin with a *starting structure*, a 3D structure of a (typically naturally occurring) protein we wish to modify.

To illustrate this concept, imagine we wish to perform a simple example modification to a protein to make it more stable, so it can still function at higher temperatures. In this case, we must minimize the protein's energy with respect to its sequence of amino acids. In structure-based design, energy is typically estimated using *energy functions*, which map the 3D geometry of a molecule to its energy, so the optimization becomes slightly more complex: we minimize the energy with respect to both the *sequence* (of amino acids) and the *conformation* (the 3D geometry of the protein, that is, the locations of all its atoms in space). While the sequence is a discrete variable, the conformation is a continuous one because coordinates in $\mathbb{R}^3$ are continuous variables. There are some physical (for example,

» key insights

- **Protein design algorithms optimize proteins for desired properties, such as improved stability and stronger and more specific binding to their ligands. These algorithms have many applications in the design of new therapeutics.**
- **Highly efficient, provably accurate algorithms are available for protein design using a simplified "pairwise discrete" model of protein chemistry.**
- **Recent algorithms have maintained these provable guarantees and computational tractability while introducing more physically realistic models of protein chemistry. These models account for proteins' and ligands' continuous flexibility, model multiple binding or structural states of proteins, and blend more complicated energetic effects.**



holonomic) constraints on how atoms can move relative to each other, and thus the conformational space can be represented most effectively using internal coordinates, resulting in the joint angle *configuration space* familiar in robotics and motion planning in computer science. Nevertheless, the full conformational space of a protein is too vast to search exhaustively, especially with a simultaneous search over sequence space.

Computational structure-based protein design arose as a response to this difficulty. Its initial goal was to overcome certain combinatorial obstructions to designing with a discretized version of the conformational space. Hence, in order to study protein design, it is first necessary to understand the structure of this simpler (but still non-trivial) discrete optimization problem. To this end, we first give a flavor for the issues that arise in discrete optimization. We examine a very special case—the case of discrete *rotamers* and a simple Markov random field (MRF)-like energy function. Next, we carefully define a mixed discrete-continuous optimization problem that gives sidechains and then backbones continuous flexibility within a conformational voxel. Then, we present algorithms that provably approximate partition functions over many states, analogously to well-known statistical inference and machine learning computations, and that exploit improved, more realistic energy functions.

It is also often useful in protein design to optimize objectives other than simply the energy of a protein. However, many useful design objectives can still often be posed in terms of the energies of multiple *biophysical states* of a protein—for example, states where it is bound to particular other molecules. Thus, the problem of *multistate design*, which we will formalize, is appropriate for tasks like optimizing the binding of one protein to another molecule, or even specific binding to a second molecule while excluding binding to a third molecule. Together with some novel types of objective functions, multistate design is a more general tool to optimize the desired function of a protein with respect to sequence.

IMAGE BY LUZVYKOVA LAROSLAVA

We will highlight provable computational techniques employed for each of these problems. These include techniques from combinatorial optimization, constraint satisfaction, machine learning, and other areas. For the relatively simple protein design problems addressed in this article, we find that algorithms with a beautiful mathematical structure suffice. This permits us to illustrate by specific examples the situation confronting practical protein designers in academic or biopharmaceutical laboratories. Throughout the article, we review algorithms of intrinsic mathematical interest and with the potential for high impact on the engineering of new molecular therapies for human disease.

In addition to this review of core algorithmic work, we will briefly discuss methods to accelerate protein design computations using GPU hardware, as well as some cases in which proteins designed using provable algorithms have performed well in experimental tests. Protein design with provable algorithms has already had success in the design of novel enzymes and proteins with therapeutic applications. As the field matures and significant errors are eliminated from more steps of the protein design process, we expect to see even more successes from this promising technique.

The Pairwise Discrete Model

Problem definition. We will now formalize this problem of stabilizing a protein using some simplifying assumptions, which will yield the most commonly used mathematical formulation of the protein design problem. We will present several algorithms to attack this problem as well as enhancements to the formulation with more sophisticated objectives and/or modeling assumptions.

Changing the sequence of a protein—that is, *mutating* it—does not alter the chemical structure of its backbone,[a] and the largest conformational changes are typically found in sidechains near the site of the mutations (we will designate these residues as *flexible*, that is, we will consider it necessary to search their conformational space). Thus, we will assume the backbone conformation (and possibly some of the sidechain conformations for residues farther from the site of mutations) is the same as in the starting structure. Moreover, analyses of sidechain conformational space have found sidechain conformations for each amino-acid type to occur in clusters known as *rotamers*. We will refer to the modal sidechain conformation in each cluster as an *ideal rotamer*. Then, for the sidechains with respect to whose amino-acid type and conformation we wish to optimize, we will assume the sidechain conformations will be ideal rotamers, meaning we need only optimize over a discrete set of (sequence, conformation) pairs in which each residue must be assigned an amino-acid type and one of the ideal rotamers for that amino-acid type.

Let $\mathbf{r}$ be a list of rotamers (which may be of any amino-acid type) for the residues that we are treating as flexible and/or mutable. If we use only ideal rotamers, $\mathbf{r}$ fully defines a sequence and conformation for the protein, so our energy function gives us a well-defined energy $E(\mathbf{r})$, and our optimization problem becomes simply finding arg min $E(\mathbf{r})$. However, one more simplifying assumption is often applied: that we are using a *pairwise energy function*, which is a sum of terms that each depend on the amino-acid types and conformations of at most two residues. In this case, we can expand

$$E(\mathbf{r}) = \sum_{i} E(i_{\mathbf{r}}) + \sum_{j<i} E(i_{\mathbf{r}}, j_{\mathbf{r}}) \qquad (1)$$

where i and j are residues, and $i_{\mathbf{r}}$ is the rotamer that $\mathbf{r}$ assigns to residue i (we place the residue position in the subscript, following the convention of the field). The pairwise energy function gives us a well-defined 1-body energy $E(i_r)$ and 2-body energy $E(i_r, j_s)$ for any rotamers i_r and j_s, and indeed these energies can be precomputed (generating an *energy matrix*) before the process of optimization begins, allowing the optimization to simply operate on the energy matrix rather than calling the energy function directly. Thus, we can formalize the protein design problem in this simple pairwise discrete model as

$$\arg\min_{\mathbf{r}} \left(\sum_{i} E(i_{\mathbf{r}}) + \sum_{j<i} E(i_{\mathbf{r}}, j_{\mathbf{r}}) \right). \qquad (2)$$

We will refer to the solution of Eq. (2) as the *global minimum-energy conformation*, or GMEC. This problem is equivalent to finding the maximum-likelihood solution for a Markov random field with only pairwise couplings.[5,7]

Finding the GMEC is unfortunately NP-hard,[27] even to approximate.[1] But much algorithmic and development work has attacked it, and most biophysically relevant cases of the problem can be solved efficiently in practice with provable guarantees of accuracy. We now review some of this work.

Work on this problem using heuristic protocols such as simulated annealing, Monte Carlo simulation, and genetic algorithms is surveyed comprehensively in Donald[5] and Gainza et al.[8] Moreover, Monte Carlo simulation in this context is often not ergodic, rendering it less reliable than mathematical methods like Monte Carlo integration that can obtain accurate error bars based on the variance of an ergodic simulation. As a result, estimates of the GMEC even from a highly optimized Monte Carlo/simulated annealing protocol exhibit empirically significant deviations from the true optimum.[31] Similar empirical deviations have been found in several other areas of structural biology requiring global minimizers, as reviewed in Gainza et al.[8] For these reasons, in this article we concentrate on provable algorithms that may be of greater interest to computer scientists.

Approaches to the problem: *The classic DEE/A* framework.* The first breakthrough toward solving Eq. (2) was the DEE algorithm[4] (with refinements due to Goldstein), which eliminates rotamers that cannot be part of the GMEC. It works by comparing two rotamers i_r and i_t for the same residue. i_r can be pruned if every conformation $\mathbf{r}$ containing i_r is higher in energy than the corresponding conformation in which i_r has been replaced by i_t, that is, if

$$\min_{\mathbf{r}} \left(E(i_r) - E(i_t) + \sum_{j \neq i} E(i_r, j_{\mathbf{r}}) - E(i_t, j_{\mathbf{r}}) \right) > 0. \qquad (3)$$

[a] Actually, there is one amino acid—proline—whose sidechain bonds to the backbone in two places, but it does not alter the repeating nitrogen-carbon-carbon pattern of backbone atoms.

Evaluating Eq. (3) is as difficult as finding the GMEC directly. But the sum of minima is always a lower bound for the minimum of a sum, so we obtain the following sufficient condition for Eq. (3), which can be evaluated in time linear in the number of residues:

$$E(i_r) - E(i_t) + \sum_{j \neq i} \min_{s} \Big(E(i_r, j_s) - E(i_t, j_s) \Big) > 0. \qquad (4)$$

We call Eq. (4) the DEE criterion. By evaluating it for each residue i and each pair of rotamers i_r and i_t that are available at i, we can greatly prune the space of rotamers that may be part of the GMEC. This pruning step is polynomial-time.[5] Thus, the combinatorial bottleneck must occur later, in the enumeration step.

DEE is an efficient algorithm, but it still may leave multiple possible rotamers for some or all of the residues. This problem has been solved by deploying the A* algorithm from artificial intelligence to find the GMEC using only the rotamers remaining, that is, using DEE/A*.[22] Briefly, the A* algorithm in this context builds a priority queue of nodes that represent a partially defined conformation **q**, which consists of rotamer assignments for only a subset $S(\mathrm{q})$ of the residues. The score of a node is a lower bound on the energy of any conformation containing all the rotamers in **q** (that is, $\min_{i_r = i_q \, \forall \, i \in S(\mathrm{q})} E(\mathbf{r})$). We repeatedly extract the lowest-scoring node from the queue and expand it by creating nodes for which one more residue has a defined rotamer. Eventually the lowest-scoring node will be a fully defined conformation. Since all conformations in other nodes must have higher energies (based on the nodes' lower bounds), this fully defined conformation must be the GMEC.

This shows that it is possible to find the GMEC with guaranteed accuracy, and indeed to do so significantly faster (in practice) than exhaustive enumeration of conformations. We will now discuss even more sophisticated and efficient algorithms for this problem.

Algorithms from weighted constraint-satisfaction problems. One source of such improved algorithms is from the field of weighted constraint-satisfaction problems (WCSPs), of which the pairwise discrete protein design problem (Eq. 2) can be seen as a special case. To use these techniques, the energy matrix is encoded as a cost function network (CFN), which includes the same type of 1- and 2-body terms as an energy matrix from protein design.[33] The most efficient provably accurate algorithms for WCSPs perform a tree search like A*, but with much more refined heuristics to guide the search (including both upper and lower bounds). They also usually employ a depth-first branch-and-bound approach rather than a best-first search like A*. As a result, far less memory is required in practice. A large set of empirical benchmarks in Traoré et al.[32] showed the Toulbar package for WCSPs significantly improved the state-of-the-art efficiency for protein design in the discrete pairwise model. Moreover, this increase in efficiency allowed direct comparison of the true GMEC (computed by WCSP algorithms) to estimated GMECs from the popular but non-provable simulated annealing algorithm, as implemented in the Rosetta software, for very large protein design problems. Significant discrepancies were found,[31] and indeed the error in simulated annealing's estimates increased with protein size. This highlights the need for algorithms with provable guarantees for protein design.

A related and also provable approach is to reduce Eq. (2) to an integer linear programming problem.[21]

Algorithms making sparsity assumptions. Although protein design as expressed in Eq. (2) is NP-hard even to approximate,[1] it is possible to add additional assumptions that make it solvable in polynomial time. Suppose we assume that some pairs of residues have uniformly zero interaction energies, such that the graph whose nodes are residues and whose edges denote residue pairs with nonzero 2-body energies is sparse, making it a *sparse residue interaction graph* (SPRIG, see Figure 1). The TreePack algorithm[36] can find the GMEC in polynomial time when the SPRIG has constant tree width. Moreover, the BWM* algorithm can find the GMEC in polynomial time and also efficiently enumerate the k best conformations in gap-free order when the SPRIG has constant branch width (where k is requested by the user).

Improved Models

The pairwise discrete model (Eq. 2) captures the most essential aspects of computational protein design, but it falls short for many practical applications. Despite the prevalence of rotameric conformations of protein sidechains, real proteins do have significant continuous flexibility in the neighborhood of each ideal rotamer. Backbone motions due to mutations are often non-negligible as well. Moreover, the energy model in Eq. (2) falls short in two ways: the most accurate energy functions are not explicitly pairwise, and the behavior of a protein is actually determined by its free energy—a quantity based on the distribution of

Figure 1. Pairwise energy functions.

(a) Pairwise energy functions compute energies between pairs of mutable residues (colored) in a protein design problem, but in practice many pairs have very small interaction energies (marked with Xs).

(b) A sparse residue interaction graph (SPRIG) has mutable residues as nodes; edges with small interaction energies can be deleted, enabling highly efficient protein design computations. Figure adapted with permission from Jou et al.[20]

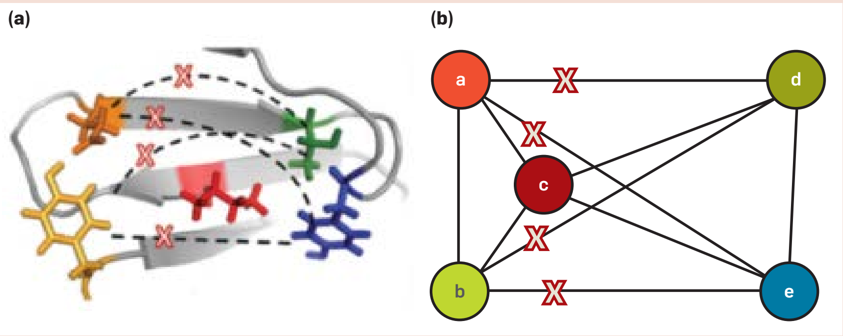
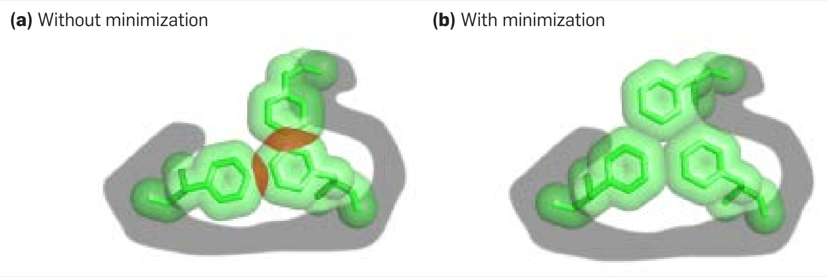


Figure 2. Favorable conformation.

(a) A conformation modeled using ideal rotamers may have steric clashes—atom pairs that are unphysically close together—even when (b) continuous minimization of the conformation's energy, without changing the rotamers of any residues, results in a very favorable energy. This underscores the need to account for continuous flexibility throughout sequence and conformational search for protein design.

Figure adapted with permission from Gainza et al.[9]



its conformations' energies—rather than on the single minimum-energy conformation. Finally, as mentioned earlier, it is often useful to have a more sophisticated objective function than simply minimizing the energy of a single biophysical state of a protein. Here, we review algorithms to address these five shortcomings (vide supra) of the discrete pairwise model of protein design.

Continuous flexibility: *Defining the problem.* The problem of continuously flexible protein design differs from Eq. (2) in that each rotamer is no longer a single conformation of its residue. Rather, each rotamer is a set of conformations, which we can model as a *voxel* in the form of bounds on each of several continuous internal coordinates. Sidechain flexibility in proteins occurs mainly in the form of changes in dihedral angles, and thus the conformation space of a protein can be modeled accurately as a union of voxels in dihedral angle space. For example, in Georgiev et al.,[12] each voxel is centered at an ideal rotamer, and allows up to ±9° of flexibility in each dihedral angle in either direction from the ideal rotamer'*s* dihedral angle. The problem is then to find the list of rotamer assignments **r** whose voxel contains the lowest-energy conformation—the *minGMEC*.

This problem has both discrete and continuous components, much like AI planning, where there are discrete steps like STRIPS or TWEAK and continuous steps like motion planning. Like robust optimization, its aim is to prevent error due to insufficiently fine sampling of conformational space—we wish to avoid eliminating a rotamer merely because its ideal rotameric conformation appears unfavorable, since a small continuous adjustment may turn out to make it optimal. Indeed, it is relatively common for ideal rotamers to be physically infeasible due to a clash (a pair of atoms too close to each other), but for a small continuous adjustment to suffice to find a favorable conformation[9,10,12] (as illustrated in Figure 2). Moreover, the optimal sequence is often significantly different, and more biophysically realistic, when continuous flexibility is taken into account than when it is neglected.[9,10]

Notably, no benefit in design is obtained by simply performing a discrete optimization and then continuously minimizing the energy of the discrete GMEC post hoc: such minimization does not change the optimal sequence that is selected. Rather, to obtain the full benefits of continuous flexibility, one must perform *minimization-aware* design that finds the minGMEC with guarantees of accuracy by taking continuous flexibility into account from the beginning. There are two general approaches to minimization awareness.

Adapting discrete algorithms to bound the continuous problem. Algorithms for discrete protein design can be adapted to be minimization-aware by having them prune using bounds on conformational energies rather than using conformational energies directly. If a list of voxels **r** represents a region in conformational space rather than a single conformation, then its energy (Eq. 1) may not be well defined per se, but a lower bound on its energy can be expressed in the form of Eq. (1), simply by minimizing each of the 1- and 2-body energy terms over the voxel. Discrete protein design algorithms can then be used to enumerate conformations in order of lower bound. Once these conformations have been continuously minimized, additional conformations can be pruned based on their lower bounds as well, allowing provable computation of the minGMEC. This approach has been developed effectively by Gainza et al.[9] and Georgiev,[12] who adapt the entire DEE/A* framework to be minimization-aware.

Other discrete algorithms also fit well into the framework of minimization-awareness based on bounds. For example, both belief propagation (BP) and the self-consistent mean field method (SCMF) are usually employed to estimate a GMEC, with no proofs of closeness to the optimal solution. However, SCMF can generate a provably correct lower bound on the GMEC energy, while tree-weighted belief propagation can generate a provably correct upper bound. Thus, by operating on bounds, both algorithms become provable. This contrasts with the exact rigid energies used with methods from weighted constraint satisfaction and integer linear programming.

Reducing the continuous problem to a discrete one. A more recent approach to minimization-aware protein design is based on machine learning and reducing the continuous protein design problem to a discrete one, without significantly compromising accuracy. Although the energy of a voxel **r** is not explicitly in the form required for discrete protein design algorithms (Eq. 1), there is a well-defined energy E(r) (generally the continuously minimized energy) that we want to optimize, and we can fit it to the form of Eq. (1) using machine learning. This approach is very efficient as implemented in the LUTE algorithm,[17] and also accommodates other improvements in biophysical modeling[b] because the user can choose the

[b] Such as non-pairwise energy functions, including those modeling solvation effects, quantum chemistry, and continuous entropy.

function E(r) that is taken as input. The implementation of LUTE described in Hallen et al.[17] also incorporates some elements of the bound-based approach to continuous flexibility, because it uses iMinDEE,[9] a minimization-aware version of DEE, as a preprocessing step, resulting in a critical improvement in its training and test error.

Backbone flexibility. Continuous sidechain flexibility handles discrepancies between ideal rotamers and the actual sidechain conformation. But an additional type of continuous flexibility—backbone flexibility—is necessary to handle discrepancies between the starting structure's backbone conformation (experimentally observed for the original sequence) and the backbone conformation that is optimal for each mutant sequence. Like continuous sidechain flexibility, backbone flexibility can be handled using voxels, which can bound the backbone's continuous internal coordinates in a neighborhood around the starting structure's backbone. The main difference is that the choice of internal coordinates is less straightforward—one must find coordinates that adequately represent the biophysically important backbone flexibility in the vicinity of the mutations without obtaining an intractably large conformational space to search. These are properties that are satisfied by sidechain dihedrals, whose locality makes them the obvious choice of internal coordinates for sidechains. But they are not satisfied by the standard backbone dihedrals ϕ and ψ, because local changes in the backbone dihedrals will propagate throughout the protein, disrupting its large-scale structure unless the changes are very small.

The DEEPer algorithm[18] addresses this problem by using only backbone motions based on experimental observations, such as the backrub motion observed in crystallographic alternates. The CATS algorithm[16] allows a larger degree of continuous motion by constructing a new type of backbone internal coordinates that can model the local motion of a contiguous segment of the protein backbone in all biophysically feasible directions (Figure 3). Both algorithms can be used in conjunction with continuous sidechain flexibility modeling and design.

Multistate design. *Defining the multistate problem.* Protein design software is already quite effective at stabilizing proteins, but we must pursue other objectives if it is truly to meet the full range of biomedical and bioengineering needs for modified proteins. Most of the important objectives involve binding—for example, binding to a protein in the human body that is involved with disease, and also not binding to other, possibly similar, proteins that are essential to normal functioning of the body. These objectives can be modeled in terms of multiple *biophysical states*—states in which the protein being designed is unbound, bound to a particular desired target, or bound to a particular undesired target, and so on. Each state a has an energy $E_a(s)$, which we can approximate as the energy of the lowest energy conformation for the state (as a function of sequence s). We want favored states to be low in energy and unfavored states to be high in energy, since this will cause the protein to adopt the favored states in preference to the unfavored ones.

Thus, following Hallen and Donald,[15] we can pose the problem of multistate design as a kind of linear programming on protein state energies. We will define linear multistate energies (LMEs), which are functions of sequence s, in the form

$$c_0 + \sum_a c_a E_a(s), \qquad (5)$$

where the coefficients c are chosen by the user. For example, to make an LME representing the binding energy between the protein we are designing and another molecule, we would set $c_b = 1$ and $c_u = -1$ where b is the bound state and u is the unbound state. We then wish to minimize not a single state's energy, but an LME, with respect to sequence. We may also wish to constrain other LMEs to have values above or below a user-specified threshold—for example, we may wish to keep the binding energy to an undesired target higher than the observed binding energy of the unmutated protein to that undesired target.

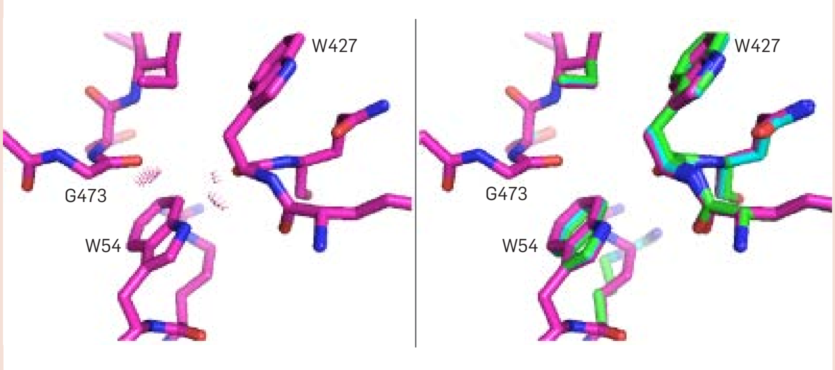
Algorithms for multistate design. The formulation in the previous section comes from Hallen and Donald,[15] who also present the first provable algorithm to solve this problem without exhaustive enumeration of sequences. This algorithm, COMETS, builds an A* tree with nodes representing partial sequences. Conformational search is handled with a combination of bounding techniques and construction of a "tree within a tree" for each promising sequence. The main tree is thus responsible for sequence search, while the inner trees each correspond to a single node of the main tree and perform conformational search for the sequence corresponding to that node.

DEE itself has also been adapted for multistate design. Specifically, within each sequence and biophysical state,

Figure 3. Backbone flexibility.

(left) Mutating residue 54 of the anti-HIV antibody VRC07 to the amino-acid tryptophan (W) improves its function in experimental tests,[30] but rigid-backbone modeling of this mutation shows unavoidable steric clashes (purple conformation).

(right) CATS finds a non-clashing conformation (green), resolving this conundrum, while DEEPer (blue) alleviates the clashes partially. Figure adapted with permission from Hallen and Donald.[16]



multistate design (as defined previously) is simply computing a GMEC, and as a result it is provably accurate to perform DEE pruning within each biophysical state as long as only competitor and candidate rotamers of the same amino-acid type are considered.[37] This technique is known as *type-dependent DEE*. The multistate design problem has also been addressed using belief propagation.[7]

As in the case of continuous flexibility, machine learning has yielded a novel and very promising technique for multistate design. The *cluster expansion* technique[14] calculates energies for a training set of sequences (for each state) and then learns an energy function that is a sum of terms dependent only on 1 or a few residues' amino acid types. In this formulation, multistate design becomes mathematically equivalent to discrete single-state design, although combinatorially easier because there are fewer amino acid types than possible rotamers. This technique has yielded designer peptides with high selectivity for their desired target in experimental tests.[13]

Finally, other formulations of multistate design besides that discussed earlier have been used quite fruitfully. The paradigm of meta-multistate design,[3] which accounts for protein dynamics, has yielded designed proteins known as DANCERS (Dynamic And Native Conformational ExchangeRs), which not only exchange between specified conformational states, but do so on the timescale of milliseconds.

Improved energy modeling. We have so far taken the energy function as an *input* to the algorithm, and assumed that given a sequence and a biophysical state, a protein will necessarily be found in the lowest-energy conformation. However, to correctly model reality, we must dig deeper.

Free energy. Physically, we must define the *energy* of a conformation c as a quantity proportional to $-T \ln P(c)$, where $P(c)$ is the probability of finding the molecule in conformation c and T is the temperature. Without loss of generality, we will choose a proportionality constant R (this defines units for the energy); R is the universal gas constant. Since different biophysical states are ultimately just different regions of conformational space, this notion of energy suffices to perform any single- or multistate design: we simply wish to maximize the probability of the molecule being in the state we desire. The probability of a biophysical state s is the sum (or integral) of the probabilities of each of its conformations $c \in C(s)$, and is thus proportional to the partition function q_s, where

$$q_s = \sum_{c \in C(s)} \exp\left(-\frac{E(c)}{RT}\right). \qquad (6)$$

It is often useful to work not with the partition function directly, but with the *free energy* $G_s = -RT \ln q_s$ of the state. Then, we simply design to reduce the free energy of desired states and increase the free energy of undesired states. Importantly, as the temperature goes to 0, G_s becomes simply the energy of the state's lowest-energy conformation, and thus we arrive at the more approximate formulation of multistate design presented previously. But this approximation introduces error at nonzero temperature, and algorithms have been developed to actually use G_s at physiological temperatures and thus account for the distribution of energies across conformational space.

Computing the partition function is unfortunately *#P*-hard, analogously to similar calculations in statistics. However, the partition function can be efficiently approximated in practice for a particular sequence and biophysical state, while modeling continuous flexibility, using the *K** algorithm.[5,23,29] The *K** algorithm builds on DEE/A* to model a thermodynamic ensemble of low-energy conformations for the bound and unbound biophysical states of a protein the user wishes to design for binding. Moreover, design based only on GMECs has been shown not to recapitulate sequences designed with *K** that performed well empirically.[29]

More efficient provable algorithms have also been developed for this problem. A partition function approximator similar to *K** but accelerated by WCSP techniques has achieved high efficiency,[34] albeit neglecting continuous flexibility, which has been shown to compromise accuracy in the *K** context.[10] The *BBK** algorithm[25] uses an A* tree with nodes from many sequences to compute the same top sequences as *K**, and thus provide the same guarantees of accuracy as *K**, in time sublinear in the number of sequences. Thus *BBK** achieves high efficiency while approximating free energy with continuous flexibility.

Improved energy functions. We have not yet addressed one very important question: How do we accurately estimate $E(c)$ for a conformation c? The most commonly used energy functions in protein design,[5] like AMBER, EEF1, and the Rosetta energy function, make many approximations due to their prioritization of speed over accuracy. More accurate energy functions based on induced electric multipoles, quantum chemistry, and Poisson-Boltzmann solvation theory are available, but they are expensive, and they violate a key assumption of the discrete pairwise model of protein design: they are not explicitly a sum of terms depending on at most 2, or indeed on any small number of residues' conformations.

One approach to these problems is to use discrete rotamers and precompute pairwise energies by choosing a "reference" conformation, perturbing it by 1 or a few rotamers at each position, and using the differences in energy between the perturbed and reference conformations as 1-, 2-, and sometimes 3-body energies. This approach yields relatively accurate energies for many systems, using either the Poisson-Boltzmann solvation model[35] or the AMOEBA forcefield (featuring induced multipoles)[24] as the energy function.

A second approach is to *learn* a representation of the energy suitable for protein design, from a training set that can be generated with any energy function. This approach has the advantages of accommodating continuous flexibility and not requiring all the 1- through 2- or 3-body perturbed conformations from the reference conformation to be physically realizable (this can be an issue in the case of backbone flexibility). Two algorithms in the OSPREY[19] protein design software exploit this approach: the EPIC algorithm learns a polynomial approximation of the continuous energy surface within a voxel, and the LUTE algorithm[17] directly learns a pairwise energy matrix (possibly augmented by triples) from sampled single-voxel minimized energies. Both EPIC and LUTE have been shown to achieve small residuals, while calling the energy function just enough to obtain an accurate characterization of the energy costs of design decisions. Thus, they greatly accelerate design using quantum chemistry- and Poisson-Boltzmann-derived energies.[17]

"Exotic" Objective Functions

Not all protein design algorithms optimize energy with respect to sequence; we now review two other approaches.

No matter how tightly a designed protein therapeutic binds its desired target, a strong reaction by the human immune system against this new protein may prevent it from remaining in the body for long, rendering it ineffective in the clinic. The EpiSweep algorithm[26] addresses this problem by finding sequences on the Pareto frontier between an OSPREY-based[2,10,19] stability design, and an objective function based on avoiding an immune reaction.

It is also sometimes useful, even when optimizing binding, to search the space of known protein backbone conformations to find one that will place sidechains in a desired pose, as in the Rosetta-Match,[38] SEEDER,[11] and MASTER[39] algorithms.

Protein Design on Graphics Processing Units

In the past decade, graphics processing unit (GPU) computation has transformed nearly every area of computational science, from molecular dynamics to computer vision to quantum chemistry. For suitably structured computations, GPUs can perform approximately 1,000 times more FLOPS per dollar spent on hardware.

In the past few years, the computational tasks that are bottlenecks in protein design computation have been implemented for GPUs. For the pairwise discrete model, the bottleneck is combinatorial optimization, which the gOSPREY software[40] accelerates on GPUs. For continuously flexible protein design, continuous energy minimization within a voxel is the bottleneck. Thus, the OSPREY software, which pioneered minimization-aware protein design, allows continuous energy minimization on GPUs as of its version 3.0, achieving >10x speedups.[19] This compares favorably with the previous flagship application of GPUs in computational structural biology, which is molecular dynamics (MD) simulations of proteins (temporal simulation of proteins using the classical mechanical potential defined by an energy function).

GPUs can exploit two types of parallelism in order to accelerate the biomolecular energy computations central to MD and protein design: (a) processing different conformations of a protein in parallel, and (b) processing different parts of the molecule in parallel. MD is better positioned to exploit (b) than protein design is, because MD evaluates energies for the entire molecule rather than merely the region around the mutations. On the other hand, continuously flexible protein design can minimize energies for a huge number of conformations in parallel, while MD must proceed through different conformations (such as, timesteps) in sequence. This type (a) parallelism in protein design applies both to conformations enumerated in order of lower bound, as in iMinDEE,[9] and to conformations sampled for the purpose of learning a discrete model of the continuously minimized energy, as in LUTE.[17]

Thus, the success of GPUs in accelerating MD computations and the favorable parallelizability of protein design compared to MD bode well for the prospect of very efficient continuously flexible protein design on GPUs, which is already quite impressive in OSPREY 3.0.[19]

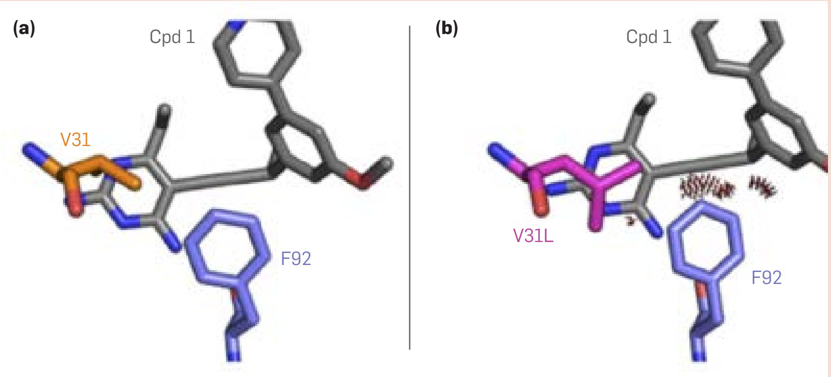
Successful Applications of Computational Protein Design with Provable Algorithms

Provable computational protein design algorithms have already produced many designs that perform well in experimental tests.[5,8,10] They have engineered a shift in substrate specificity from one "molecular operand" (input molecule) to another,[2] and they can predict bacterial mutations in enzyme-coding genes that make the bacterial enzymes resistant to particular antibiotics (Figure 4)— predictions that have been confirmed both in vitro[6] and in vivo.[28]

Finally, and perhaps most importantly, proteins designed using provable algorithms have shown promise in the design of therapeutics. Using the techniques reviewed in this paper (in particular, the K^* algorithm[30] in OSPREY[19]), we collaborated with the NIH Vaccine Research Center to design a broadly neutralizing antibody against HIV with unprecedented breadth and potency (that is, stronger activity against a broader range of HIV strains) that is now in clinical trials (Clinical Trial Identifier: NCT030151817 and six others). The OSPREY/K^* algorithm has also produced peptides that inhibit a protein involved in cystic fibrosis.[29] In addition to such direct design of therapeutics, computational prediction of resistance mutations to drug candidates[6,28] will help combat resis-

Figure 4. Computational prediction of antibiotic resistance.

(a) the bacterial (Staphyoloccus aureus) enzyme dihydrofolate reductase binds a drug candidate ("Cpd 1") tightly, inhibiting the enzyme's function, but (b) mutating position 31 of the enzyme from amino-acid type valine to leucine causes steric clashes that impeded binding, allowing the bacteria to resist the antibiotic. This predicted resistance mutation was observed experimentally after being predicted by the K^* algorithm as implemented in the OSPREY software.[19] Figure adapted with permission from Reeve et al.[28]



tance against new drugs (especially antibiotics) entering the clinic.

Conclusion

Provable computational protein design algorithms have advanced significantly in the last decade. Algorithms for the pairwise discrete approximations have matured, and significant progress is being made with improved biophysical models and for the design of clinically relevant proteins and peptides. Proteins, especially antibodies, are attracting increasing attention from the pharmaceutical industry as drug candidates. These algorithms also have the potential to be transformative in the design of non-protein drugs, because unlike most drug design algorithms, they can search a large space of drug candidates in time sublinear in the size of the space and still guarantee to find the best candidates as if searching one by one.

To achieve the full potential of protein design, it is necessary to further improve the accuracy of the biophysical model. More accurate energy functions, improved modeling of protein-water interactions, and modeling of broader conformational spaces (both for search and for entropy computations) are likely to be important here. Provable guarantees are essential in this endeavor, as they ensure modeling error is the only error in protein design calculations, both allowing new models to be evaluated accurately and preventing design calculations based on accurate models from nonetheless failing due to algorithmic error. As work continues on these important problems, the future of computational protein design with provable algorithms looks bright.

Acknowledgments

Thanks to Lydia Kavraki, Tomás Lozano-Pérez, Nate Guerin, Jeff Martin, Pablo Gainza, Cynthia Rudin, and members of the Donald Lab. We also thank Toyota Technological Institute of Chicago (M.A.H) and the NIH (grants RO1 GM-78031 and RO1 GM-118543 to B.R.D) for funding.

References

1. Chazelle, B., Kingsford, C. and Singh, M. A semidefinite programming approach to side chain positioning with new rounding strategies. *INFORMS J. Computing, Computational Biology Special Issue 16*, 4 (2004), 380–392.
2. Chen, C., Georgiev, I., Anderson, A. and Donald, B. Computational structure-based redesign of enzyme activity. *Proc. Nat. Acad. Sci. U. S. A. 106*, 10 (2009), 3764–3769.
3. Davey, J., Damry, A., Goto, N. and Chica, R. Rational design of proteins that exchange on functional timescales. *Nature Chemical Biology 13*, 12 (2017), 1280.
4. Desmet, J., Maeyer, M., Hazes, B. and Lasters, I. The dead-end elimination theorem and its use in protein sidechain positioning. *Nature 356* (1992), 539–542.
5. Donald, B. *Algorithms in Structural Molecular Biology.* MIT Press, Cambridge, MA, 2011.
6. Frey, K., Georgiev, I., Donald, B. and Anderson, A. Predicting resistance mutations using protein design algorithms. *Proc. Nat. Acad. Sci. U. S. A. 107*, 31 (2010), 13707–13712.
7. Fromer, M., Yanover, C., and Linial, M. Design of multispecific protein sequences using probabilistic graphical modeling. *Proteins: Structure, Function, and Bioinformatics 78*, 3 (2010), 530–547.
8. Gainza, P., Nisonoff, H. and Donald, B. Algorithms for protein design. *Current Opinion in Structural Biology 39* (2016), 16–26.
9. Gainza, P., Roberts, K. and Donald, B. Protein design using continuous rotamers. *PLoS Computational Biology 8*, 1 (2012), e1002335.
10. Gainza, P. et al. OSPREY: Protein design with ensembles, flexibility, and provable algorithms. *Methods in Enzymology 523* (2013), 87–107.
11. Gainza, P., Vollers, S. and Correia, B. Mining protein surfaces for binding seeds. (Aug. 2017). RosettaCon.
12. Georgiev, I., Lilien, R. and Donald, B. The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. *J. Computational Chemistry 29*, 10 (2008), 1527–1542.
13. Grigoryan, G., Reinke, A. and Keating, A. Design of protein-interaction specificity affords selective bZIP-binding peptides. *Nature 458*, 7240 (2009), 859–864.
14. Grigoryan, G., Zhou, F., Lustig, S., Ceder, G., Morgan, D., and Keating, A. Ultra-fast evaluation of protein energies directly from sequence. *PLoS Computational Biology 2*, 6 (2006), e63.
15. Hallen, M. and Donald, B. COMETS (Constrained Optimization of Multistate Energies by Tree Search): A provable and efficient protein design algorithm to optimize binding affinity and specificity with respect to sequence. *J. Computational Biology 23*, 5 (2016), 311–321.
16. Hallen, M. and Donald, B. CATS (Coordinates of Atoms by Taylor Series): Protein design with backbone flexibility in all locally feasible directions. *Bioinformatics 33*, 14 (2017), i5–i12.
17. Hallen, M., Jou, J. and Donald, B. LUTE (Local Unpruned Tuple Expansion): Accurate continuously flexible protein design with general energy functions and rigid-rotamer-like efficiency. In *Proceedings of the Intern. Conf. on Research in Computational Molecular Biology*. Springer, 2016, 122–136.
18. Hallen, M., Keedy, D. and Donald, B. Dead-end elimination with perturbations (DEEPer): A provable protein design algorithm with continuous sidechain and backbone flexibility. *Proteins: Structure, Function and Bioinformatics 81*, 1 (2013), 18–39.
19. Hallen, M. et al. OSPREY 3.0: Open-source protein redesign for you, with powerful new features. *J. Computational Chemistry 39*, 30 (2018), 2494–2507.
20. Jou, J., Jain, S., Georgiev, I., and Donald, B. BWM*: A Novel, Provable, Ensemble-Based Dynamic Programming Algorithm for Sparse Approximations of Computational Protein Design. *Journal of Computational Biology 23*, 6 (2016), 413–424.
21. Kingsford, C., Chazelle, B., and Singh, M. Solving and analyzing sidechain positioning problems using linear and integer programming. *Bioinformatics 21*, 7 (2005), 1028–1039.
22. Leach, A. and Lemon, A. Exploring the conformational space of protein side chains using dead-end elimination and the A* algorithm. *Proteins: Structure, Function, and Bioinformatics 33*, 2 (1998), 227–239.
23. Lilien, R., Stevens, B., Anderson, A. and Donald, B. A novel ensemble-based scoring and search algorithm for protein redesign and its application to modify the substrate specificity of the gramicidin synthetase A phenylalanine adenylation enzyme. *J. Computational Biology 12*, 6 (2005), 740–761.
24. LuCore, S., Litman, J., Powers, K., Gao, S., Lynn, A., Tollefson, W., Fenn, T., Washington, T. and Schnieders, M. Dead-end elimination with a polarizable force field repacks PCNA structures. *Biophysical J. 109*, 4 (2015), 816–826.
25. Ojewole, A., Jou, J., Fowler, V. and Donald, B. BBK*(Branch and Bound Over K*): A provable and efficient ensemble-based protein design algorithm to optimize stability and binding affinity over large sequence spaces. *J. Computational Biology* (2018). Epub ahead of print.
26. Parker, A., Choi, Y., Griswold, K. and Bailey-Kellogg, C. Structure-guided deimmunization of therapeutic proteins. *J. Computational Biology 20*, 2 (2013), 152–165.
27. Pierce, N. and Winfree, E. Protein design is *NP*-hard. *Protein Engineering 15*, 10 (2002), 779–782.
28. Reeve, S., Gainza, P., Frey, K., Georgiev, I., Donald, B. and Anderson, A. Protein design algorithms predict viable resistance to an experimental antifolate. In *Proceedings of the Nat. Acad. Sci. U. S. A. 112*, 3 (2015), 749–754.
29. Roberts, K., Cushing, P., Boisguerin, P., Madden, D. and Donald, B. Computational design of a PDZ domain peptide inhibitor that rescues CFTR activity. *PLoS Computational Biology 8*, 4 (2012), e1002477.
30. Rudicell, R. et al. Enhanced potency of a broadly neutralizing HIV-1 antibody in vitro improves protection against lentiviral infection in vivo. *J. Virology 88*, 21 (2014), 12669–12682.
31. Simoncini, D., Allouche, D., Givry, S., Delmas, C., Barbe, S. and Schiex, T. Guaranteed discrete energy optimization on large protein design problems. *J. Chemical Theory and Computation 11*, 12 (2015), 5980–5989.
32. Traoré, S., Allouche, D., André, I., Givry, S., Katsirelos, G., Schiex, T. and Barbe, S. A new framework for computational protein design through cost function network optimization. *Bioinformatics 29*, 17 (2013), 2129–2136.
33. Traoré, S., Roberts, K., Allouche, D., Donald, B., André, I., Schiex, T., and Barbe, S. Fast search algorithms for computational protein design. *J. Computational Chemistry 37*, 12 (2016), 1048–1058.
34. Viricel, C., Simoncini, D., Allouche, D., Givry, S., Barbe, S. and Schiex, T. Approximate counting with deterministic guarantees for affinity computation. *Modelling, Computation and Optimization in Information Systems and Management Sciences.* Springer, 2015, 165–176.
35. Vizcarra, C., Zhang, N., Marshall, S., Wingreen, N., Zeng, C. and Mayo, S. An improved pairwise decomposable finite-difference Poisson-Boltzmann method for computational protein design. *J. Computational Chemistry 29*, 7 (2008), 1153–1162.
36. Xu, J. and Berger, B. Fast and accurate algorithms for protein side-chain packing. *J. ACM 53*, 4 (2006), 533–557.
37. Yanover, C., Fromer, M. and Shifman, J. Dead-end elimination for multistate protein design. *J. Computational Chemistry 28*, 13 (2007), 2122–2129.
38. Zanghellini, A., Jiang, L., Wollacott, A., Cheng, G., Meiler, J., Althoff, E., Röthlisberger, D. and Baker, D. New algorithms and an in silico benchmark for computational enzyme design. *Protein Science 15*, 12 (2006), 2785–2794.
39. Zhou, J. and Grigoryan, G. Rapid search for tertiary fragments reveals protein sequence–structure relationships. *Protein Science 24*, 4 (2015),508–524.
40. Zhou, Y., Xu, W., Donald, B. and Zeng, J. An efficient parallel algorithm for accelerating computational protein design. *Bioinformatics 30*, 12 (2014), i255–i263.

Mark A. Hallen (mhallen@ttic.edu) is a research assistant professor at the Toyota Technological Institute at Chicago, IL, USA.

Bruce R. Donald (brd+cacm19@cs.duke.edu) is the James B. Duke Professor of Computer Science at Duke University, as well as a professor of chemistry and biochemistry in the Duke University Medical Center, Durham, NC, USA.

The authors are founders of Gavilán Biodesign, Inc.

Watch the authors discuss this work in the exclusive *Communications* video. https://cacm.acm.org/videos/protein-design-by-provable-algorithms

Digital Threats:
Research and Practice
Association for Computing Machinery
Advancing Computing as a Science & Profession

Attention: Undergraduate *and* Graduate Computing Students

There's an ACM Student Research Competition (SRC) at a SIG Conference of interest to you!

STUDENT RESEARCH COMPETITION

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."

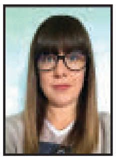
David Mueller
North Carolina State University | SIGDOC 2018

"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

"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



"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."

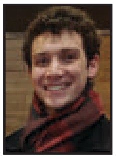
Gemma Catolino
University of Salerno | MobileSoft 2018

"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

"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



"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

Check the SRC Submission Dates: ***https://src.acm.org/submissions***

- Participants receive: $500 (USD) travel expenses
- All Winners receive a medal and monetary award. First place winners advance to the SRC Grand Finals
- Grand Finals Winners receive a handsome certificate and monetary award at the ACM Awards Banquet

Questions? Contact Nanette Hernandez, ACM's SRC Coordinator: ***hernandez@hq.acm.org***

research highlights

DOI:10.1145/3356906

To view the accompanying paper, visit doi.acm.org/10.1145/3356903 rh

Technical Perspective
The Scalability of CertiKOS

By Andrew W. Appel

FOR MODERATE-SIZE SEQUENTIAL programs, formal verification works—we can build a formal machine-checkable proof that a program is correct, with respect to a formal specification in logic. Machine-checked formal verifications of functional correctness have already been demonstrated for operating-system microkernels, optimizing compilers, cryptographic primitives and protocols, and so on.

But suppose we want to verify a high-performance hypervisor kernel programmed in C, that runs on a real (x86) machine, that is capable of booting up Linux in each of its (hypervisor) guest partitions? Real machines these days are multicore—the hypervisor should provide multicore partitions that can host multicore guests, all protected from each other, but interacting via shared memory synchronized by locks. Furthermore, the operating system itself should be multicore, with fine-grain synchronization—we do not want one global lock guarding all the system calls by all the cores and threads.

The authors of the following paper illustrate that formal verification can scale up to a moderate-size program (6,500 lines of C) that has substantial shared-memory concurrency. They succeed by a ruthless and disciplined use of modularity and *contextual refinement*: each module of the C program behaves "the same" as its functional specification *in any context*; and each module compiles to assembly language that behaves "the same" as the C program *in any context*. They certify this with machine-checkable proofs. Therefore, they call the approach *Certified Abstraction Layers*.

The authors illustrate that formal verification can scale up to a moderate-size program that has substantial shared-memory concurrency.

Here's an example of contextual refinement: Suppose module *A* interfaces to module *B* using the principle of Abstract Data Types (also known as "representation hiding"): Module *B* has private variables with public interface methods. Module *A* never reads and writes *B*'s variables directly but calls upon *B*'s methods to do it. We can say that module *A* is the *context* for running module *B*. What can *A* observe about *B*? Only the data values passed to, and returned from, *B*'s interface methods. We can substitute a different implementation for *B* that "behaves the same" from *B*'s point of view. In particular, we could write a *functional specification* for *B* written in a functional programming language or written in mathematical logic; then we could write a C program implementing module *B*. Module *A* cannot tell the difference between the functional specification and the C-language implementation. Then, use a proved-correct C compiler, and module *A* cannot tell the difference between the C program and the assembly-language program.

That explanation works well for single-threaded programs where the modules are connected at function-call interfaces. The CertiKOS team has previously demonstrated their Certified Abstraction Layer methodology to prove the correctness of a single-core version of CertiKOS.

In this paper, the authors describe what it takes to extend the methodology to concurrency.

With multithreading, module *A* (the context) can interface to module *B* not only through function calls, but by acquiring and releasing locks and then reading and writing the shared memory locations controlled by those locks. This gives a more complicated notion of "behaves the same"—it's not just the arguments and return-values of procedure calls. With multicore, contextual refinement describes, in each thread individually, the relation between a *synchronization trace* of the functional specification and of the C program.

What we all know about software is that it never stays still. It would do no good to verify correctness of an operating system, if next week when we commit a change to the source-code repository, we would have to throw away the proof and start over. Proofs about programs must be highly modular, just like the programs themselves. The CertiKOS team shows how to design contextual refinements that are module-by-module, even in the presence of concurrency. That means, when you commit a change to the implementation of one module, you just need to adjust the proof of that module alone.

To make their proofs so modular, the CertiKOS team has had to pay careful attention to abstraction interfaces. As a result, the C program is extremely well structured, layered, and modular. This has benefits even for those who read the C program without ever looking at the proofs. C

Andrew W. Appel is the Eugene Higgins Professor of Computer Science at Princeton University, Princeton, NJ, USA.

DOI:10.1145/3356903

Building Certified Concurrent OS Kernels

By Ronghui Gu, Zhong Shao, Hao Chen, Jieung Kim, Jérémie Koenig, Xiongnan (Newman) Wu, Vilhelm Sjöberg, and David Costanzo

Abstract

Operating system (OS) kernels form the backbone of system software. They can have a significant impact on the resilience and security of today's computers. Recent efforts have demonstrated the feasibility of formally verifying simple general-purpose kernels, but they have ignored the important issues of concurrency, which include not just user and I/O concurrency on a single core, but also multicore parallelism with fine-grained locking. In this work, we present CertiKOS, a novel compositional framework for building verified concurrent OS kernels. Concurrency allows interleaved execution of programs belonging to different abstraction layers and running on different CPUs/threads. Each such layer can have a different set of observable events. In CertiKOS, these layers and their observable events can be formally specified, and each module can then be verified at the abstraction level it belongs to. To link all the verified pieces together, CertiKOS enforces a so-called contextual refinement property for every such piece, which states that the implementation will behave like its specification under any concurrent context with any valid interleaving. Using CertiKOS, we have successfully developed a practical concurrent OS kernel, called mC2, and built the formal proofs of its correctness in Coq. The mC2 kernel is written in 6500 lines of C and x86 assembly and runs on stock x86 multicore machines. To our knowledge, this is the first correctness proof of a general-purpose concurrent OS kernel with fine-grained locking.

1. INTRODUCTION

Operating system (OS) kernels and hypervisors form the backbone of safety-critical software systems. Hence, it is highly desirable to verify the correctness of these programs formally. Recent efforts[5, 6, 10, 13, 17] have shown that it is feasible to formally prove the functional correctness of simple general-purpose kernels, file systems, and device drivers. However, none of these systems have addressed the important issues of concurrency,[2] such as not only user and I/O concurrency on a single CPU but also multicore parallelism with fine-grained locking. This severely limits the applicability of today's formally verified system software.

What makes the verification of concurrent OS kernels so challenging? First, concurrent kernels allow interleaved execution of kernel/user modules belonging to different abstraction layers; they contain many interdependent components that are difficult to untangle. Several researchers[22, 23] believe that the combination of fine-grained concurrency and the kernels' functional complexity makes formal verification intractable, and even if it is possible, its cost would far exceed that of verifying a sequential kernel.

Second, concurrent kernels need to make all three types of concurrency (i.e., user, I/O, and multicore) coherently work together. User and I/O concurrency are difficult to reason about because they rely on thread yield/sleep/wakeup primitives or interrupts to switch control and support synchronization but still provide the illusion that each user process is executed uninterruptedly and sequentially. Multicore concurrency with fine-grained locking may utilize sophisticated spinlock implementations such as MCS locks[21] that are also hard to verify.

Third, concurrent kernels may also require that some of their system calls eventually return, but this depends on the progress of the concurrent primitives used in the kernels. Formally proving starvation-freedom[15] for concurrent objects only became possible recently.[20] Standard Mesa-style condition variables (CV)[18] do not enforce starvation-freedom; this can be fixed by storing CVs in a FIFO queue. But the solution is not trivial, and even the popular, most up-to-date OS textbook,[2] has gotten it wrong.

Fourth, given the high cost of building certified concurrent kernels, it is important that these kernels can be quickly adapted to support new hardware platforms and applications.[3] However, if we are unable to model the interference among different components in an extensible way, even a small change to the kernel could incur a huv ge reverification overhead.

In this paper, we present CertiKOS, a compositional framework that tackles all these challenges. We believe that, to control the complexity of concurrent kernels and to prove a strong support of extensibility, we must first have a *compositional* specification that can untangle *all* the kernel interdependencies and encapsulate interference among different kernel objects. Because the very purpose of an OS kernel is to build layers of abstraction over bare machines, we insist on uncovering and specifying these layers, and then verifying each kernel module at the abstraction level it belongs to.

The functional correctness of an OS kernel is often stated as a *refinement*—that is, the behavior of the C/assembly implementation of a kernel K is fully captured by its abstract functional specification S. Of course, the ultimate goal of having a certified kernel is to reason about programs

The original version of this paper is entitled "CertiKOS: An Extensible Architecture for Building Certified Concurrent OS Kernels" and was published in the *Proceedings of 12th USENIX Symposium on Operating System Design and Implementation*, 2016, 653–669

running on top of (or along with) the kernel. It is thus important to ensure that given any kernel extension or user program P, the combined code $K \oplus P$ also refines $S \oplus P$. If this fails to hold, the kernel is functionally incorrect as P can observe some behavior of K that does not satisfy S.

In the concurrent setting, such a *contextual refinement* property must hold not only for any context program P but also for any *environment* context ε. When focusing on some thread set, each ε defines a specific instance on how other threads/CPUs respond to this thread set. With shared-memory concurrency, interference between ε and the focused thread set is both necessary and common.

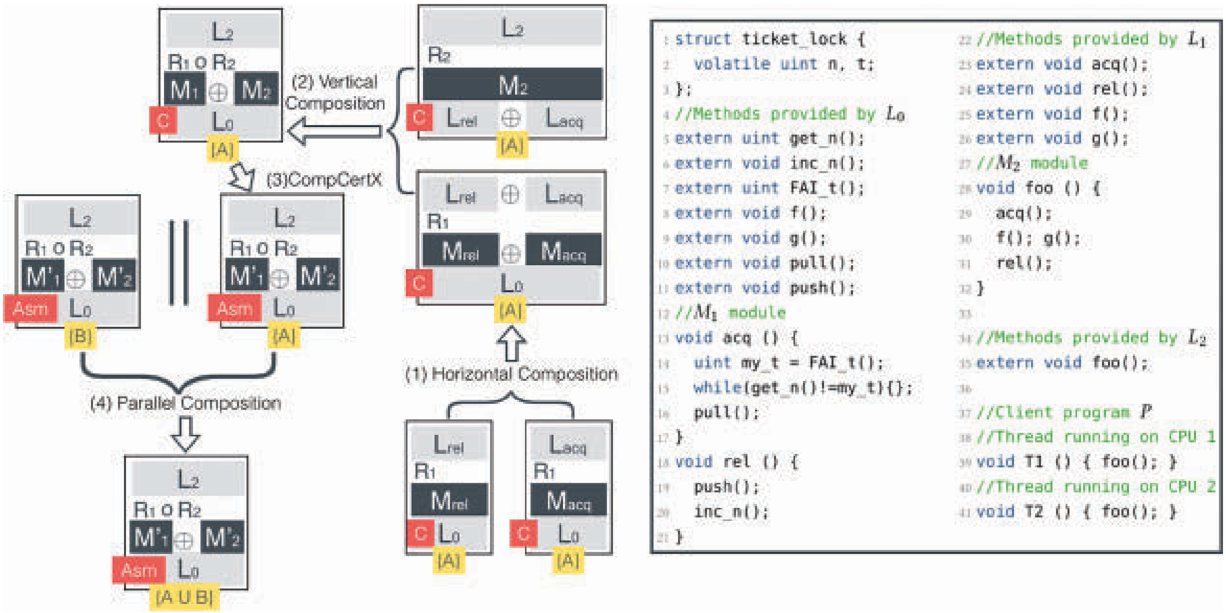
In CertiKOS, we introduce *certified concurrent abstraction layers* to state such contextual refinement properties (see Figure 1). Each abstraction layer, parameterized over some specific ε, is an *assembly-level machine* extended with a particular set of abstract objects, that is, abstract states plus atomic primitives. These layers enable modular verification and can be composed in several manners. Later in Section 3, we show how the use of ε at each layer allows us to verify concurrent programs using standard techniques for verifying sequential programs. Indeed, most of our kernel components are written in a variant of C (called ClightX[10]) and verified at the C level. These certified C layers can be compiled and linked together into certified assembly layers using CompCertX[10, 12]—a *thread-safe* version of the CompCert compiler.[19] Thus, under CertiKOS, an otherwise prohibitive verification task can be decomposed into many simple and easily automatable ones, and proven global properties can be propagated down to the assembly level.

Using CertiKOS, we have successfully developed a fully certified concurrent OS kernel mC2 in the Coq proof assistant. The mC2 kernel consists of 6500 lines of C and x86 assembly, supports both fine-grained locking and thread yield/sleep/wakeup primitives, and can run on stock x86 multicore machines. mC2 can also double as a hypervisor and boot multiple instances of Linux in guest virtual machines (VM) running on different CPUs. It guarantees not only functional correctness, that is, the mC2 kernel implementation satisfies its system-call specification, but also liveness property, that is, all system calls will eventually return. The entire proof effort for supporting concurrency took less than two person-years. To the best of our knowledge, mC2 is the first fully verified general-purpose concurrent OS kernel with fine-grained locking.

2. OVERVIEW OF OUR APPROACH

In this section, to illustrate our layered techniques, we will walk through a small example (see Figure 1) that uses a lock to protect a critical section. In this example, client program P has two threads running on two different CPUs; each thread makes one call to primitive foo provided by concurrent layer interface L_2. Interface L_2 is implemented by concurrent module M_2, which in turn is built on top of interface L_1. Method foo calls two primitives f and g in a critical section protected by a lock. The lock is implemented over interface L_0 using the ticket lock algorithm[21] in module M_1. The lock maintains two integer variables n (the "now serving" ticket number) and t (the "next" ticket number). Lock-acquire method acq fetches and increments the next ticket number (by FAI_t) and spins until the fetched number is served. Lock-release

Figure 1. The certified (concurrent) abstraction layer, $L_0 \vdash_{R_1} M_{acq} : L_{acq}$, is a predicate plus its mechanized proof object showing that the implementation of the ticket lock acquire M_{acq} running on the underlay interface L_0 indeed faithfully implements the desirable overlay interface L_{acq}. The implementation M_{acq} is written in C, whereas the interfaces L_0 and L_{acq} are written in Coq. The implementation relation is denoted as R_1. This layer can be (1) horizontally composed with another layer (e.g., the lock release operation) if they have identical state views (i.e., with the same R_1) and are based on the same underlay interface L_0. The composed layer can also be (2) vertically composed with another layer that relies on its overlay interface. Certified C layers can be compiled into certified assembly layers using our (3) CompCertX compiler. In the concurrent setting, these layers can also be (4) composed in parallel.



```
struct ticket_lock {
  volatile uint n, t;
};
//Methods provided by L0
extern uint get_n();
extern void inc_n();
extern uint FAI_t();
extern void f();
extern void g();
extern void pull();
extern void push();
//M1 module
void acq () {
  uint my_t = FAI_t();
  while(get_n()!=my_t){};
  pull();
}
void rel () {
  push();
  inc_n();
}
//Methods provided by L1
extern void acq();
extern void rel();
extern void f();
extern void g();
//M2 module
void foo () {
  acq();
  f(); g();
  rel();
}

//Methods provided by L2
extern void foo();

//Client program P
//Thread running on CPU 1
void T1 () { foo(); }
//Thread running on CPU 2
void T2 () { foo(); }
```

method rel simply increments the "now serving" ticket number by inc_n. These primitives are provided by L_0 and implemented using x86 atomic instructions. Interface L_0 also provides primitives f and g that are later passed on to L_1, as well as *ghost* primitives pull and push that logically mark the acquisition and release of locks. Such ghost primitives only help the verification process and are not needed for the program to execute.

Here, the concurrent layer interface (e.g., L_0) provides a set of primitives that can be invoked at this level and uses *events* to capture primitives' effects that are *visible* to other CPUs/threads. For example, event $\boxed{\text{1.FAI\_t}}$ represents the invocation of FAI_t by CPU 1. In this way, one execution of a concurrent program running on a layer machine can be *specified* by a sequence of events, which we call a *logical log*. For example, if two CPUs are executed in the order 1–2–2–1–1–2–1–2–1–1–1–2–2, running program P (see Figure 1) over the layer machine of L_0 generates the log:

$$\begin{gathered}\boxed{\text{1.FAI\_t}} \bullet \boxed{\text{2.FAI\_t}} \bullet \boxed{\text{2.get\_n}} \bullet \boxed{\text{1.get\_n}} \bullet \boxed{\text{1.pull}} \bullet \boxed{\text{2.get\_n}} \\ \bullet \boxed{\text{1.f}} \bullet \boxed{\text{2.get\_n}} \bullet \boxed{\text{1.g}} \bullet \boxed{\text{1.push}} \bullet \boxed{\text{1.inc\_n}} \bullet \boxed{\text{2.get\_n}} \bullet \boxed{\text{2.pull}}\end{gathered} \tag{2.1}$$

Thus, a concurrent module M over L can be specified by how M produces events (provided by L). M can then be verified by building a *certified abstraction layer*, $L \vdash_R M : L'$, stating that the events generated by M over L are fully captured by the desirable interface L'. Note that the events provided by L and L' might not be exactly the same, and the relation between events at different layers is denoted as R.

Take the lock-acquire implementation M_{acq} in Figure 1 as an example. The goal is to prove that $\mathrm{L}_0 \vdash_{\mathbf{id}} M_{\mathsf{acq}} : L_{\mathsf{acq}}$ holds with an identical relation **id** (between events at two layers), where the events generated by L_{acq} (on behalf of thread t) satisfy the pattern:

$$\ldots \bullet \boxed{\text{t.FAI\_t}} \bullet \ldots \bullet \boxed{\text{t.get\_n}} \bullet \ldots \bullet \boxed{\text{t.get\_n}} \bullet \ldots \bullet \boxed{\text{t.get\_n}} \bullet \ldots \bullet \boxed{\text{t.pull}} \bullet \ldots$$

Events generated by other threads (or CPUs) are omitted here.

To achieve modular verification, we parameterize each layer interface L with an *active* thread set A, and then carefully define its set of valid *environment contexts*, denoted as $\mathrm{EC}(L, A)$. Each environment context ε captures a specific instance—from a particular run—of the list of events that other threads or CPUs (not in A) return when responding to the events generated by threads in A. We can then define a new *thread-modular* machine $\Pi_{\mathrm{L(A)}}(P, \varepsilon)$ that will operate like the usual assembly machine when P switches control to threads in A, but will only obtain the list of events from the environment context ε when P switches control to threads outside A. Here, we use $L(A)$ to denote the layer interface with an active thread set A that consists the same set of abstract objects with L.

Note that if A is a singleton, for each ε, $\Pi_{\mathrm{L(A)}}$ behaves like a sequential machine: it first queries ε for the events generated by other threads, and then executes the next instruction of the active thread. We use $\blacktriangleright$ to denote a query to ε. The lock-acquire function, on behalf of thread t, can be specified in $L_{\mathsf{acq}}(\{t\})$ as:

$$\blacktriangleright\boxed{\text{t.FAI\_t}} \blacktriangleright\boxed{\text{t.get\_n}} \blacktriangleright\boxed{\text{t.get\_n}} \blacktriangleright\boxed{\text{t.get\_n}}\ldots \blacktriangleright\boxed{\text{t.pull}} \tag{2.2}$$

In this model, other threads' behaviors and the potential interleaving are encapsulated into those queries to ε. We can then verify module M_{acq} as it were sequential:

$$L_0(\{t\}) \vdash_{\mathbf{id}} M_{\mathsf{acq}} : L_{\mathsf{acq}}(\{t\})$$

By verifying that the lock-release function M_{rel} also meets its specification L_{rel}, we can apply the *horizontal composition* rule to obtain the composed layer (where we use L_1' to denote $L_{\mathsf{acq}} \oplus L_{\mathsf{rel}}$):

$$L_0(\{t\}) \vdash_{\mathbf{id}} M_{\mathsf{acq}} \oplus M_{\mathsf{rel}} : L_1'(\{t\}) \tag{2.3}$$

If every valid environment context $\varepsilon \in \mathrm{EC}(L_1', \{t\})$ guarantees that the loop of get_n in thread t terminates, we can lift $L_1'(\{t\})$ to a higher level layer interface $L_1(\{t\})$, which specifies the lock-acquire as $\blacktriangleright\boxed{\text{t acq}}$. We use R_1 to denote the relation between the events of $L_1'(\{t\})$ and $L_1(\{t\})$, for example, $\boxed{\text{t.acq}}$ is mapped to the event sequence in (2.2). We can prove the following certified layer:

$$L_1'(\{t\}) \vdash_{R_1} \varnothing : L_1(\{t\}) \tag{2.4}$$

where Ø states that no code is involved at this step. By applying the *vertical composition* rule to (2.3) and (2.4), we have that:

$$L_0(\{t\}) \vdash_{\mathbf{id} \circ R_1} M_{\mathsf{acq}} \oplus M_{\mathsf{rel}} : L_1(\{t\})$$

With our new compositional layer semantics, these "per-thread" certified layers can be soundly composed in parallel when their *rely conditions* (i.e., the constraints to environmental interference) are compatible with each other. For example, we can also derive the certified layer for the ticket lock on behalf of some thread $t'(\neq t)$. By showing that the events generated by t' belong to $\mathrm{EC}(L_1, \{t\})$ and vice versa, we can apply the *parallel composition* rule to derive:

$$L_0(\{t, t'\}) \vdash_{\mathbf{id} \circ R_1} M_{\mathsf{acq}} \oplus M_{\mathsf{rel}} : L_1(\{t, t'\})$$

Any observable behavior of running P with $M_{\mathsf{acq}} \oplus M_{\mathsf{rel}}$ (denoted as M_1 in Figure 1) over $L_0(\{1, 2\})$ can be captured by running P directly on top of $L_1(\{1, 2\})$. For example, the behavior in (2.1) can be captured by the following log over $L_1(\{1, 2\})$:

$$\boxed{\text{1.acq}} \bullet \boxed{\text{1.f}} \bullet \boxed{\text{1.g}} \bullet \boxed{\text{1.rel}} \bullet \boxed{\text{2.acq}}$$

Based on the layer interface $L_1(\{t\})$, we can continue verifying that the module M_2 satisfies a higher level interface $L_2(\{t\})$, where foo is specified as $\blacktriangleright\boxed{\text{t.foo}}$. The relation between these two layer interfaces maps the event $\boxed{\text{t.foo}}$ of $L_2(\{t\})$ into the event sequence $\boxed{\text{t.acq}} \bullet \boxed{\text{t.f}} \bullet \boxed{\text{t.g}} \bullet \boxed{\text{t.rel}}$ of $L_1(\{t\})$. As the primitive foo is specified by a single event, we call it an *atomic object*. The observable behaviors of running P over the layer machine $L_2(\{1, 2\})$ consist of only two logs: $\boxed{\text{1.foo}} \bullet \boxed{\text{2.foo}}$ and $\boxed{\text{2.foo}} \bullet \boxed{\text{1.foo}}$.

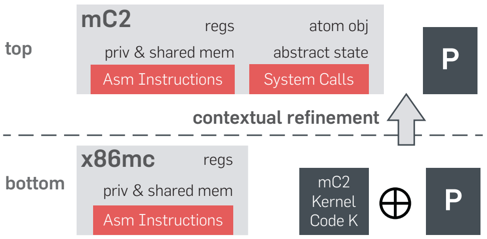
In this way, we can decompose our mC2 kernel K into many modules and verify them at the layer interfaces they belong to, as if there were only a single active, sequential thread. These per-thread layers (whose topmost layer interface is L_{mC2}) can be composed into per-CPU layers and then further combined into a single multicore machine (see Section 3 and Figure 5). We use x86mc to denote this assembly-level multicore machine, $[\![\cdot]\!]_{\text{x86mc}}$ to denote the whole-machine semantics for x86mc, and $[\![\cdot]\!]_{\text{mC2}}$ to denote the machine semantics equipped with the topmost layer interface. The composed certified layers imply the *contextual refinement* property:

$$\forall P,\ [\![K \oplus P]\!]_{\text{x86mc}} \sqsubseteq [\![P]\!]_{\text{mC2}}$$

which says that, for any context user program P, the observable behaviors of running P together with K over the multicore machine x86mc are fully captured by running P directly over L_{mC2} (see Figure 2). We call L_{mC2} a *deep specification*[10] of K over x86mc, because there is no need to ever look at K again; any property about K over x86mc can be proved using L_{mC2} alone.

Overview of the mC2 kernel. Figure 3 shows the system architecture of mC2. The mC2 system was initially developed in the context of a large DARPA-funded research project. It is a concurrent OS kernel that can also double as a hypervisor. It runs on an unmanned ground vehicle (UGV) with a multicore Intel Core i7 machine. On top of mC2, we run three Ubuntu Linux systems as guests (one each on the first three cores). Each virtual machine runs several robot architecture definition language (RADL) nodes that have fixed hardware capabilities such as access to GPS, radar, etc. The kernel also contains a few simple device drivers (e.g., interrupt controllers, serial and keyboard devices). More complex devices are either supported at the user level, or passed through (via IOMMU) to various guest Linux VMs. By running different RADL nodes in different VMs, mC2 provides strong isolation so that even if attackers take control of one VM, they still cannot break into other VMs and compromise the overall mission of the UGV.

Figure 2. The contextual refinement property that has been proved for mC2.

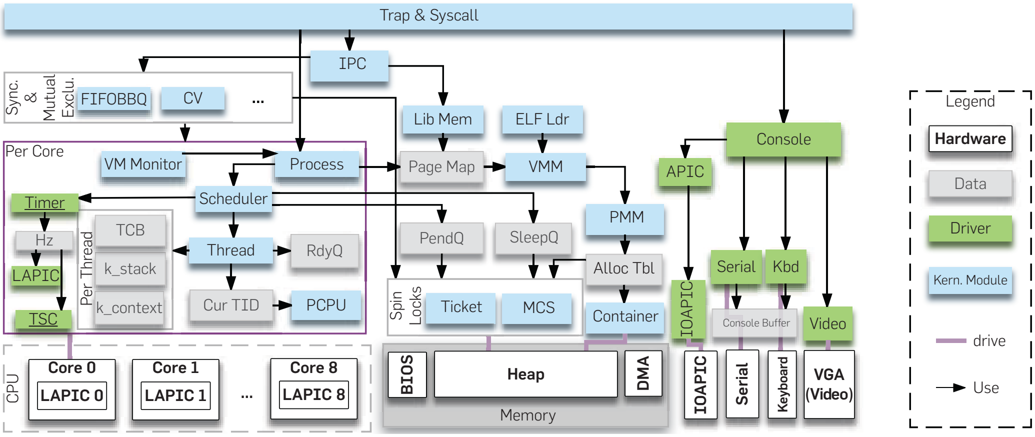


What have we proved? Using CertiKOS, we have successfully built a fully certified version of the mC2 kernel and proved its contextual refinement property with respect to a high-level deep specification for mC2. This functional correctness property implies that all system calls and traps will always strictly follow high-level specifications, run *safely*, and eventually *terminate*; there will be no data race, no code injection attacks, no buffer overflows, no null pointer access, no integer overflow, etc.

More importantly, because for any program P, we have $[\![K \oplus P]\!]_{\text{x86mc}}$ refines $[\![P]\!]_{\text{mC2}}$, we can also derive the *behavior equivalence* property for P, that is, whatever behavior a user can deduce about P based on the high-level specification for the mC2 kernel K, the actual linked system $K \oplus P$ running on the concrete x86mc machine would indeed behave exactly as expected. All global properties proven at the system-call specification level can be propagated down to the lowest assembly machine.

Assumptions and limitations. The mC2 kernel is not as comprehensive as real-world kernels such as Linux. For example, mC2 currently lacks a certified storage system. The main goal of this work is to show that it is feasible to build certified concurrent kernels with fine-grained locking. We

Figure 3. The mC2 hypervisor kernel contains various shared objects such as spinlock modules (Ticket, MCS), sleep queues (SleepQ, for implementing queuing locks and condition variables), pending queues (PendQ, for waking up a thread on another CPU), container-based physical and virtual memory management modules (Container, PMM, VMM), a Lib Mem module (for implementing shared-memory IPC), synchronization modules (FIFOBBQ, CV), and an IPC module. Within each core (the purple box), we have the per-CPU scheduler, the kernel-thread management module, the process management module, and the virtualization module (VM Monitor). Each kernel thread has its own thread-control block (TCB), context, and stack.



did not try to incorporate all the latest advances for multicore kernels into mC2.

Regarding specification, there are 450 lines of Coq code (LOC) to specify the system calls (the topmost layer interface; see Table 1) and 943 LOC to specify the x86 hardware machine model (the bottommost layer interface). These are in our trusted computing base. We keep them small to limit the room for errors and ease the review process.

Our assembly machine assumes strong sequential consistency for all atomic instructions. We believe our proof should remain valid for the x86 TSO model because (1) all our concurrent layers guarantee that nonatomic memory accesses are properly synchronized; and (2) the TSO order guarantees that all atomic synchronization operations are properly ordered. Nevertheless, more formalization work is needed to turn our proofs over sequential-consistent machines into those over the TSO machines.[23]

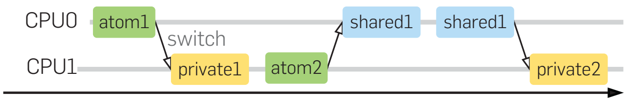
Also, our machine model only covers a small portion of the x86 hardware features and cannot be used to verify some kernel components, such as a bootloader, a PreInit module (which initializes the CPUs and the devices), an ELF loader, and some device drivers (e.g., disk driver). Their verification is left for future work.

We also trust the Coq proof checker and the CompCertX assembler for converting assembly into machine code.

3. CONCURRENT LAYER MACHINES

In this section, we explain the concurrent layer design principles, and show how to introduce per-CPU layer interfaces, based on a multicore hardware machine model.

Π_{x86mc} **multicore hardware model** allows arbitrary interleavings at the level of *assembly instructions*. At each step, the hardware *nondeterministically* picks one CPU and executes the next assembly instruction on that CPU. Each assembly instruction is classified as *atomic*, *shared*, or *private*, depending on the memory it accesses. One interleaving of an example program running on two CPUs is:



The memory locations are *logically* categorized into two kinds: the ones *private* to a single CPU/thread and the ones *shared* by multiple CPUs/threads. Private memory accesses do not need to be synchronized, whereas nonatomic shared memory accesses need to be protected by some synchronization mechanisms (e.g., locks), which are normally implemented using atomic instructions (e.g., fetch-and-add). With proper protection, each shared memory operation can be viewed as if it were atomic.

The *atomic object* is an abstraction of some segment of well-synchronized shared memory, combined with operations that can be performed over that segment. It consists of a set of primitives, an initial state, and a *logical log* containing the entire history of the operations that were performed on the object during an execution schedule. Each primitive invocation records a *single* corresponding event in the log. For example, the above interleaving produces the logical log $\boxed{0.\text{atom1}} \bullet \boxed{1.\text{atom2}}$. We require that these events contain enough information so we can derive the current state of each atomic object by *replaying* the entire log over the object's initial state.

As shown in Figure 4, a *concurrent layer interface* contains both *private objects* (e.g., O_i) and *atomic objects* (e.g., O_j), along with some invariants imposed on them. These objects are verified by building certified concurrent layers via forward simulations, which imply strong *contextual refinement* relations:

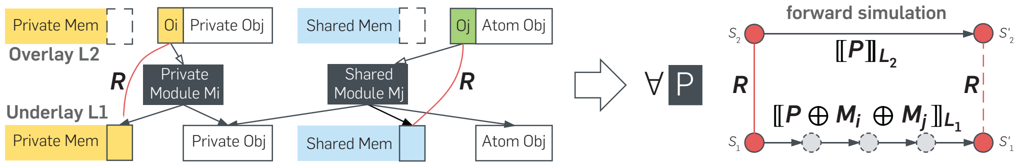
Definition 1 (Contextual Refinement). *We say that machine* Π_{L_1} *contextually refines machine* Π_{L_2} *(written as* $\forall P, [\![P]\!]_{L_1} \sqsubseteq [\![P]\!]_{L_2}$*), if, and only if, for any P that does not get stuck on* Π_{L_2}*, we also have that (1) P does not get stuck on* Π_{L_1}*; and (2) any observable behavior of P on* Π_{L_1} *is also observed on* Π_{L_2}*.*

However, proving such contextual refinements directly on a multicore, nondeterministic hardware model is difficult

Table 1. Verified system calls of the mC2 hypervisor kernel.

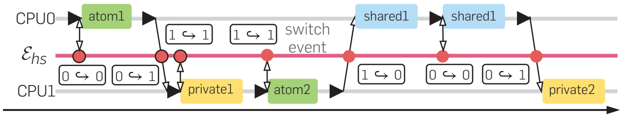
kernel_init, get_quota, send, recv, rz_spawn, spawn, sleep, yield, wakeup, kill, getc, putc, get_tsc_per_ms, get_curid, vm_exit_info, vm_mmap, vm_set_seg, vm_get_reg, vm_set_reg, vm_get_next_eip, vm_inject_event, vm_check_int_shadow, vm_run, vm_check_pending_event, vm_intercept_int_window, vm_get_tsc_offset, vm_set_tsc_offset, vm_rdmsr, vm_wrmsr

Figure 4. The overlay interface L_2 is a more abstract interface, built on top of the underlay interface L_1, and implemented by private module M_i and shared module M_j. Private objects in L_2 only access the private memory of L_1. Atomic objects are implemented by shared modules (e.g., M_{acq} in Figure 1) that may access lower-level atomic objects (e.g., FAI_t), private objects, and shared memory. Memory regions of L_1 accessed by the layer implementation are hidden and replaced by newly introduced objects of L_2. The simulation relation R is defined between these memory regions and objects, for example, R_1 in Section 2. Then, the certified concurrent layer $L_1 \vdash_R M_i \oplus M_j : L_2$ can be built by proving the forward simulation: whenever two states s_1, s_2 are related by R, and running any P over the layer machine based on L_2 takes s_2 to s_2' in one step, then there exists s_1' such that running $P \oplus M_i \oplus M_j$ over L_1 takes s_1 to s_1' in multiple steps, and s_1' and s_2' are also related by R.



because we must consider all possible interleavings. In the rest of this section, we show how to gradually refine this hardware model into a more abstract one that is suitable for reasoning about concurrent code in a CPU-local fashion.

Π_{hs}: machine model with hardware scheduler. By parameterized with a *hardware scheduler* ε_{hs} that specifies a particular interleaving for an execution, the machine model Π_{hs} becomes *deterministic*. To take a program from Π_{x86mc} and run it on top of Π_{hs}, we insert a *logical switch point*, denoted as ▶, before each assembly instruction. At each switch point, the machine first queries ε_{hs} and gets the CPU ID that will execute next. All the *switch decisions* made by ε_{hs} are stored in the logical log state as switch events, for example, $\boxed{i \hookrightarrow j}$ denotes a switch event from CPU i to j. The previous example on Π_{x86mc} can then be simulated on Π_{hs} by the following ε_{hs}:



The log recorded by this execution is as follows:

$$\boxed{0 \hookrightarrow 0} \bullet \boxed{0.\texttt{atom}_1} \bullet \boxed{0 \hookrightarrow 1} \bullet \boxed{1 \hookrightarrow 1} \bullet \boxed{1 \hookrightarrow 1} \bullet \boxed{1.\texttt{atom}_2} \bullet \boxed{1 \hookrightarrow 0} \bullet \boxed{0 \hookrightarrow 0} \bullet \boxed{0 \hookrightarrow 1}$$

The behavior of running a program P over this machine with a particular ε_{hs} is the generated log denoted as $\Pi_{\text{hs}}(P, \varepsilon_{hs})$. We write EC_{hs} to represent the set of all possible hardware schedulers. Then, the whole-machine semantics can be defined as a set of logs:

$$[\![P]\!]_{hs} = \{\Pi_{\text{hs}}(P, \varepsilon_{hs}) \mid \varepsilon_{hs} \in \text{EC}_{hs}\}$$

To ensure the correctness of this machine model, we prove that it is *contextually refined* by the hardware model Π_{x86mc}:

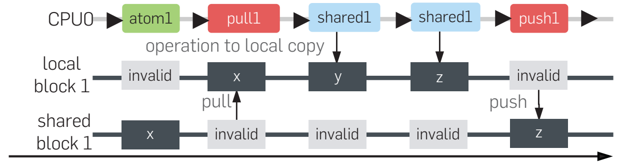
Lemma 1 (Correctness of Π_{HS}). $\forall P, [\![P]\!]_{x86mc} \sqsubseteq [\![P]\!]_{hs}$

Π_{lcm}: machine with local copies of the shared memory. To enforce that shared memory accesses are well synchronized, we introduce a new machine model (Π_{lcm}) that equips each CPU with local copies of shared memory blocks along with *valid bits*. The relation between CPU's local copies and the global shared memory is maintained through two new *ghost* primitives, pull and push.

The pull operation over a particular CompCert-style memory block[19] updates a CPU's local copy of that block to be equal to the one in the shared memory, marking the local block as valid and the shared version as invalid. Conversely, the push operation updates the shared version to be equal to the local block, marking the shared version as valid and the local block as invalid.

If a program tries to pull an invalid shared memory block or push/access an invalid local block, the program gets stuck. We make sure that every shared memory access is always performed on its valid local copy, thus systematically enforcing valid accesses to the shared memory. Note that all of these constructions are completely *logical* and do not introduce any performance overhead.

The shared memory updates of the previous example can be simulated on Π_{lcm} as follows:



Among each shared memory block and all of its local copies, only one can be valid at any moment of the machine execution. Therefore, for any program P with a potential *data race*, there exists a hardware scheduler such that P gets stuck on Π_{lcm}. By showing that a program P is safe (never gets stuck) on Π_{lcm} for *all possible* hardware schedulers, we guarantee that P is data-race free.

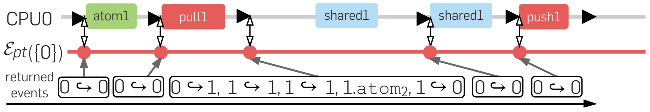
We have shown (in Coq) that Π_{lcm} is correct with respect to the previous machine model Π_{hs} with the EC_{hs}:

Lemma 2 (Correctness of Π_{LCM}). $\forall P, [\![P]\!]_{hs} \sqsubseteq [\![P]\!]_{lcm}$

Π_{pt}: partial machine with environment context. To achieve local reasoning, we introduce a partial machine model Π_{pt} that can be used to reason about the programs running on a subset of CPUs, by parametrizing the model over the behaviors of an *environment context*, that is, the rest of the CPUs.

We call a given local subset of CPUs the *active CPU set* (denoted as A). The partial machine model is configured with an active CPU set and it queries the environment context whenever it reaches a switch point that attempts to switch to a CPU outside the active set.

The set of environment contexts for A in this machine model is denoted as $\text{EC}(pt, A)$. Each environment context $\varepsilon_{pt(A)} \in \text{EC}(pt, A)$ is a *response function*, which takes the current log and returns a list of events from the context programs, that is, those outside of A. The response function simulates the observable behavior of the context CPUs and imposes some invariants over the context. The hardware scheduler is also a part of the environment context. In other words, the events returned by the response function also include switch events. The execution of CPU 0 in the previous example can be simulated with an $\varepsilon_{pt(\{0\})}$ function:



For example, at the third switch point, $\varepsilon_{pt(\{0\})}$ returns the event list $\boxed{0 \hookrightarrow 1} \bullet \boxed{1 \hookrightarrow 1} \bullet \boxed{1 \hookrightarrow 1} \bullet \boxed{1.\texttt{atom}_2} \bullet \boxed{1 \hookrightarrow 0}$.

Suppose we have verified that two programs, separately running with two *disjoint* active CPU sets A and B, produce event lists satisfying invariants INV_A and INV_B, respectively. If INV_A is consistent with the environment-context invariant of B, and INV_B is consistent with the environment-context invariant of A, then we can compose the two separate programs into a single program with active set $A \cup B$. This combined program is guaranteed to produce event lists satisfying the combined invariant $\text{INV}_A \wedge \text{INV}_B$. Using the machine semantics as a set of produced logs, this composition can then be defined as a contextual refinement:

Lemma 3 (Composition of partial machines).

$$\forall P, [\![P]\!]_{pt(A\cup B)} \sqsubseteq [\![P]\!]_{pt(A)} \cap [\![P]\!]_{pt(B)} \qquad if\ A\cap B=\varnothing$$

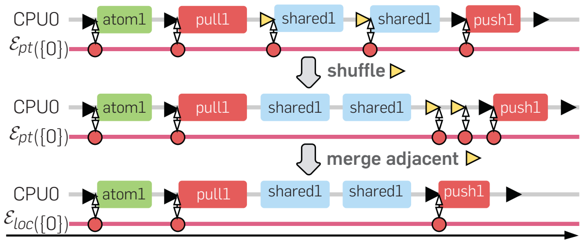
After composing the programs on all CPUs, the context CPU set becomes empty and the composed invariant holds on the whole machine. As there is no context CPU, the environment context is reduced to the hardware scheduler, which only generates the switch events. In other words, letting C be the entire CPU set, we have that EC(pt, C) = EC_{hs}. Thus, we can show that this *composed machine* with the entire CPU set C is refined by Π_{lcm}:

Lemma 4 (Correctness of Π_{PT}). $\forall P, [\![P]\!]_{lcm} \sqsubseteq [\![P]\!]_{pt(C)}$

Π_{loc}: CPU-local machine model. If we focus on a single active CPU i, the partial machine model provides a sequential-like interface configured with an environment context representing all other CPUs. However, in this model, there is a switch point before each instruction, so program verification still needs to handle many unnecessary interleavings, for example, those between private operations. Thus, we introduce a CPU-local machine model (denoted as Π_{loc}) for a CPU i, in which switch points only appear before atomic or push/pull operations. The switch points before shared or private operations are removed via two steps: *shuffling* and *merging*.

Every switch point before a shared or private operation can be shuffled to the front of the next atomic operation by introducing a *log cache*. For such switch points, query results from the environment context are stored in the log cache. The cached events are applied to the logical log just before the next atomic or push/pull operations. This is sound because a shared operation can only be performed when the current local copy of shared memory is valid, meaning that no other context program can interfere with the operation.

Once the switch points are shuffled properly, we merge all the adjacent switch points together. When we merge switch points, we also need to merge the switch events generated by the environment context. For example, the change of switch points for the previous example on CPU-local machine is as follows:



Lemma 5 (Correctness of Π_{LOC}).

$$\forall P, [\![P]\!]_{pt(\{i\})} \sqsubseteq [\![P]\!]_{loc(\{i\})}$$

Finally, we obtain the refinement relation from the multicore hardware model to the CPU-local machine model by composing all of the refinement relations together (see Figure 5).

We introduce and verify the mC2 kernel on top of the CPU-local machine model Π_{loc}. The refinement proof guarantees that the proven properties can be propagated down to the multicore hardware model Π_{x86mc}.

All our proofs (such as every step in Figure 5) are implemented, composed, and machine-checked in Coq. Each refinement step is implemented as a CompCert-style upward-forward simulation from one layer machine to another. Each machine contains the usual (CPU-local) abstract state, a logical global log (for shared state), and an environment context. The simulation relation is defined over these two machine states, and matches the informal intuitions given in this and next sections.

4. CERTIFYING THE mC2 KERNEL

Based on the CPU-local layer machine model Π_{loc}, the certified mC2 kernel can be built by introducing a series of logical abstraction layers and decomposing the otherwise complex verification tasks into a large number of small tractable ones.

In the mC2 kernel, the preinitialization module forms the bottom layer machine that connects to Π_{loc}, instantiated with a particular *active CPU c*. The trap handler forms the top layer machine that provides system call interface and serves as a specification to the whole kernel, instantiated with a particular active thread running on that active CPU c. Our main theorem states that any global properties proved at the topmost layer machine can be propagated down to the lowest hardware machine. In this section, we explain selected components in more detail.

The preinitialization layer machine defines some x86 hardware behaviors, such as page walking upon memory load (when paging is turned on), saving and restoring the trap frame in the case of interrupts and exceptions (e.g., page fault), and exchanging data between devices and memory. The hardware memory management unit (MMU) is modeled in a way that mirrors the paging hardware (see Figure 6a). When paging is enabled, memory accesses made by both

Figure 5. Contextual refinement between concurrent layer machines.

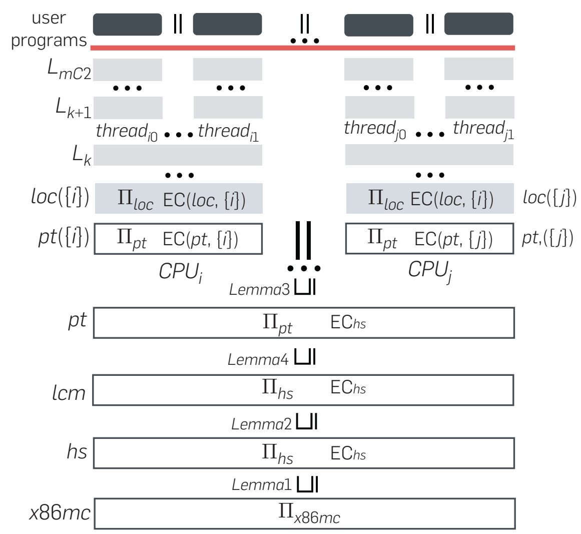
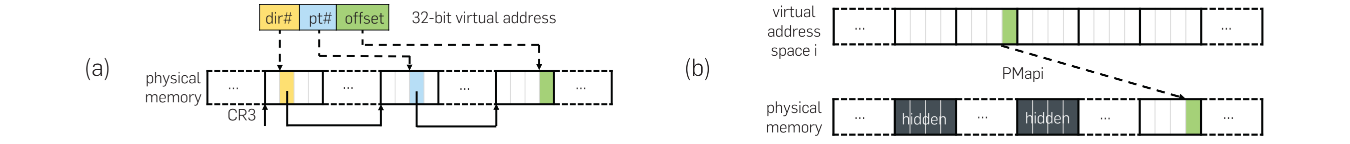


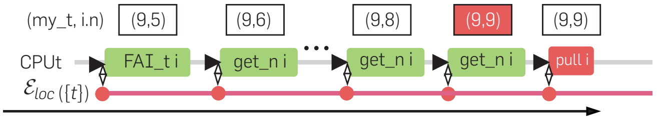
Figure 6. (a) Hardware MMU using two-level page map; (b) virtual address space *i* set up by page map .



the kernel and the user programs are translated using the page map pointed to by the register CR3. When page faults occur, the fault information is stored in CR2 and the page fault handler is triggered.

Spinlock module provides fine-grained lock objects as the base of synchronization mechanisms. Figure 1 shows one spinlock implementation using the ticket lock algorithm. It depends on an *atomic ticket object* consisting of two fields: next ticket number t and now-serving ticket number n. In mC2, we introduce an array of ticket objects; each of them (identified by a specific lock index i) can be used to protect a segment of shared memory. The ticket objects can only be manipulated via atomic primitives that generate events. For example, fetch-and-increment operation (FAI_t) to the i-th t done by CPU *c* generates an event $\boxed{\text{c.FAI\_t i}}$. Note that FAI_t is implemented using instruction xaddl with the lock prefix in x86.

The lock implementation generates a list of events; for example, when CPU *c* acquires the lock i, it continuously generates the event $\boxed{\text{c.get\_n i}}$ (line 15) until the latest n is increased to the ticket value returned by the event $\boxed{\text{c.FAI\_t i}}$ (line 14), and then followed by the event $\boxed{\text{c.pull i}}$ (line 16):



Verifying the linearizability and starvation-freedom of the ticket lock is equivalent to proving that under a *fair* hardware scheduler ε_{hs}, the ticket lock implementation is a *termination-sensitive* contextual refinement of its atomic specification.[20] There are two main proof obligations: (1) the lock guarantees *mutual exclusion*, and (2) the acq operation eventually succeeds.

The *mutual exclusion* property relies on the fact that, at any time, only the thread whose ticket t is equal to the current serving ticket (i.e., n) can hold the lock, and each thread's ticket t is unique. Here, we must also handle potential integer overflows for t and n. As long as the total number of CPUs (i.e., #CPU) in the machine is less than 2^{32} (determined by the uint type), this uniqueness property can be ensured. Then, it is safe to pull the shared memory associated with the lock i to the local copy at line 16. Before releasing the lock, the local copy is pushed back to the shared memory at line 19.

The *starvation-freedom* property relies on the fairness of the scheduler, that is, any CPU can be scheduled within *n* steps for some *n*. We define invariant INV_{lock} over the environment context to say that environmental lock-holders will release the lock within *m* steps. By enforcing INV_{lock}, we can prove that the while-loop in acq (line 15) terminates in $n \times m \times$ #CPU iterations on a CPU-local machine.

After showing the above two properties, we can build a *certified CPU-local layer*, whose overlay interface contains an atomic specification (L_{acq}) that simply generates an event $\boxed{\text{t.acq i}}$. These per-CPU certified layers can be composed in parallel as long as INV_{lock} holds on each CPU's local execution.

This event-based specification for the spinlock is also general enough to capture other implementations such as the *MCS Lock*. In mC2, we have also implemented a version of MCS locks.[16] The starvation-freedom proof is similar to that of the ticket lock. The difference is that the MCS lock-release operation waits in a loop until the next waiting thread (if it exists) has added itself to a linked list, so we need similar proofs for both acquisition and release.

Shared memory management provides a protocol to share physical pages among different user processes. A physical page can be mapped into multiple processes' page maps. For each page, we maintain a *logical owner set*. For example, a user process k_1 can share its private physical page *i* to another process k_2 and the logical owner set of page *i* is changed from $\{k_1\}$ to $\{k_1, k_2\}$. A shared page can only be freed when its owner set is a *singleton*.

Shared queue library abstracts the queues implemented as doubly linked lists into abstract queue states (i.e., Coq lists). Local enqueue and dequeue operations are specified over the abstract lists. Shared queue operations are protected by spinlocks and are specified by queue events $\boxed{\texttt{t.enQ i e}}$ and $\boxed{\texttt{t.deQ i}}$. These events can be replayed (with the function $\mathbb{R}_{\text{queue}}$) to construct the queue state. For example, if the current log of the *i*-th shared queue is $[\boxed{\texttt{t}_0\texttt{.enQ i 2}}]$, and the event list returned by ε is $[\boxed{\texttt{t}_1\texttt{.enQ i 3}}, \boxed{\texttt{t}_2\texttt{.enQ i 5}}]$, then the resulting log of calling deQ is:

$$\boxed{\texttt{t}_0\texttt{.enQ i 2}} \bullet \boxed{\texttt{t}_1\texttt{.enQ i 3}} \bullet \boxed{\texttt{t}_2\texttt{.enQ i 5}} \bullet \boxed{\texttt{t}_0\texttt{.deQ i}}$$

By replaying the log, the queue state is [3;5] and deQ returns 2.

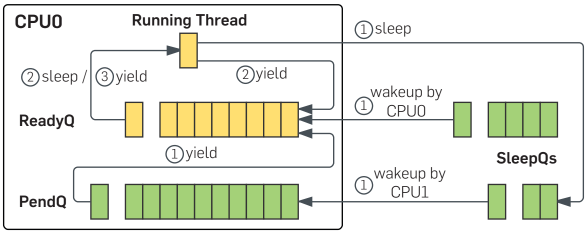
Thread management introduces the thread control block and manages the resources of dynamically spawned threads (e.g., via quotas) and their metadata (e.g., children, thread state). For each thread, one page (4KB) is allocated for its *kernel stack*. We use an external tool[4] to show that the stack usage of our compiled kernel is less than 4KB, so stack overflows cannot occur inside the kernel.

Thread control switches are implemented by the context switch function. This assembly function saves the register set of the current thread and restores the register set of another thread on the same CPU. As the instruction pointer register (EIP) and stack pointer register (ESP) are saved and restored in this procedure, this kernel context switch function does not satisfy the C calling convention and has to be verified at the assembly level. Based on this context switch function and the shared queue library, we can verify three scheduling primitives: yield, sleep, and wakeup (see Figure 7).

Thread-local machine models can be built based on the thread management layers. The first step is to extend the environment context with a *software scheduler* (i.e., abstracting the concrete scheduling procedure), resulting in a new environment context ε_{ss}. The scheduling primitives generate the `t.yield`, `t.sleep i` and `t.wakeup i` events. ε_{ss} responds with the next thread ID to execute. The second step is to introduce the *active thread set* to represent the active threads on the active CPU, and extend ε_{ss} with the *context threads*, that is, the rest of the threads running on the active CPU. The composition structure is similar to the one of Lemma 3. In this way, higher layers can be built upon a thread-local machine with a single active thread on the active CPU (see Figure 5).

Condition variable (CV) is a synchronization object that enables a thread to wait for a change to be made to a shared state. Standard Mesa-style CVs[18] do not guarantee starvation-freedom: a thread waiting on a CV may not be signaled within a bounded number of execution steps. We have implemented a starvation-free CV using condition queues as shown by the popular, most up-to-date OS textbook.[2] However, we have found a bug in the FIFOBBQ implementation as shown in that textbook. Their system can get stuck in two cases: (1) when the destroyed CV is kept inside the remove queue (rmvQ), which will block the insert call to wake up the proper waiter; (2) when multiple CVs are woken up within a short period and the lock-holding CV thread is not the head of rmvQ, that thread will be removed from rmvQ and return to sleep, but will never be woken up again. We fixed this issue by postponing the removal of the CV thread from rmvQ, until woken thread that is allowed to proceed finishes its work; this thread is now responsible for removing itself from rmvQ, as well as waking up the next thread in rmvQ.

Figure 7. Each CPU has a private ready queue ReadyQ and a shared pending queue PendQ. The environmental CPUs can insert threads to the current CPU's PendQ. The mC2 kernel also provides a set of shared sleeping queues SleepQs. The yield primitive moves a thread from PendQ to ReadyQ and then switches to the next ready thread. The sleep primitive simply adds the running thread to a SleepQ and runs the next ready thread. The wakeup primitive contains two cases. If the thread to be woken up belongs to the current CPU, it will be added to the corresponding ReadyQ. Otherwise, the thread is added to PendQ of the CPU it belongs to.



5. EVALUATION

5.1. Proof effort and cost of change

Overall, our certified mC2 kernel consists of 6500 lines of C and x86 assembly. The concurrency extensions were completed in about two person-years. The new concurrency framework (to specify, build, and link certified concurrent abstraction layers) took about one person-year to develop. We extended the certified sequential mCertiKOS kernel[5, 8, 10] (which took another two person-years to develop in total) with various features, such as dynamic memory management, container support for controlling resource consumption, Intel hardware virtualization support, shared memory IPC, two-copy synchronous IPC, ticket and MCS locks, new schedulers, condition variables, etc. Some of these features were initially added in the sequential setting but later ported to the concurrent setting. The verification of these features was completed around one person-year. During this development process, many of our certified layers underwent many modifications and extensions. The CertiKOS framework allows such incremental development to take place much more smoothly. For example, certified layers in the sequential kernel can be directly ported to the concurrent setting if they only access private state. We have also adapted the work by Chen et al.[5] on interruptible kernels with device drivers to our multicore model.

Regarding the proof effort, there are 5249 lines of additional specifications for the various kernel functions, and about 40K LOC used to define auxiliary definitions, lemmas, theorems, and invariants. Additionally, there are 50K lines of proof scripts for proving the newly added concurrency features.

5.2. Bugs found

Other than the FIFOBBQ bug, we have also found a few other bugs during verification. Our initial ticket-lock implementation contains a particularly subtle bug: the spinning loop body (line 15 in Figure 1) was implemented as while(get_n() < my_t). This passed all our tests, but during the verification, we found that it did not satisfy the atomic specification as the ticket field might overflow. For example, if the next ticket number t is $(2^{32}–1)$, an overflow will occur in acq (line 14 in Figure 1) and the returned ticket my_t will equal to 0. In this case, current-serving number n is not less than my_t and acq gets the lock immediately, violating the mutual exclusion property.

5.3. Performance evaluation

Although performance is not the main emphasis of this work, we have run a number of micro and macro benchmarks to measure the speedup and overhead of mC2, and to compare mC2 with existing systems such as KVM and seL4. All experiments have been performed on a machine with one Intel Xeon CPU with four cores running at 2.8 GHz. As the power control code has not been verified, we disabled the turbo boost and power management features of the hardware during experiments.

5.4. Concurrency overhead

The runtime overhead introduced by concurrency in mC2 mainly comes from *the latency of spinlocks*.

The mC2 kernel provides two kinds of spinlocks: ticket lock and MCS lock. They have the same interface and thus are interchangeable. In order to measure their performance, we put an empty critical section (payload) under the protection of a single lock. The latency is measured by taking a sample of 10000 consecutive lock acquires and releases (transactions) on each round.

Figure 8a shows the results of our latency measurement. In the single-core case, ticket locks impose 34 cycles of overhead, whereas MCS locks impose 74 cycles as shown in the line chart. As the number of cores grows, the latency increases rapidly. As the slowdown should be proportional to the number of cores, to show the actual efficiency of the lock implementations, we normalize the latency against the baseline (single core) multiplied by the number of cores ($n*t_1/t_n$). As can be seen from the bar chart, efficiency remains about the same for MCS locks, but decreases for ticket locks.

Now that we have compared MCS locks with ticket locks, we present the remaining evaluations in this section using only the ticket lock implementation of mC2.

5.5. Hypervisor performance

To evaluate mC2 as a hypervisor, we measured the performance of some macro benchmarks on Ubuntu 12.04.2 LTS running as a guest. We ran the benchmarks on Linux as guest in both KVM and mC2, as well as on the bare metal. The guest Ubuntu is installed on an internal SSD drive. KVM and mC2 are installed on a USB stick. We use the standard 4KB pages in every setting—huge pages are not used.

Figure 8b contains a compilation of standard macro benchmarks: unpacking a Linux 4.0-rc4 kernel archive, compiling the Linux 4.0-rc4 kernel source, running Apache HTTPerf on loopback, and the DaCaPo Benchmark 9.12. We normalize the running times of the benchmarks using the bare metal performance as a baseline (100%). The overhead of mC2 is moderate and comparable to KVM. In some cases, mC2 performs better than KVM; we suspect this is because KVM has a Linux host and thus has a larger cache footprint. For benchmarks with a large number of file operations, such as Uncompress Linux source and Tomcat, mC2 performs worse. This is because mC2 exposes the raw disk interface to the guest via VirtIO (instead of passing it through), and its disk driver does not provide good buffering support.

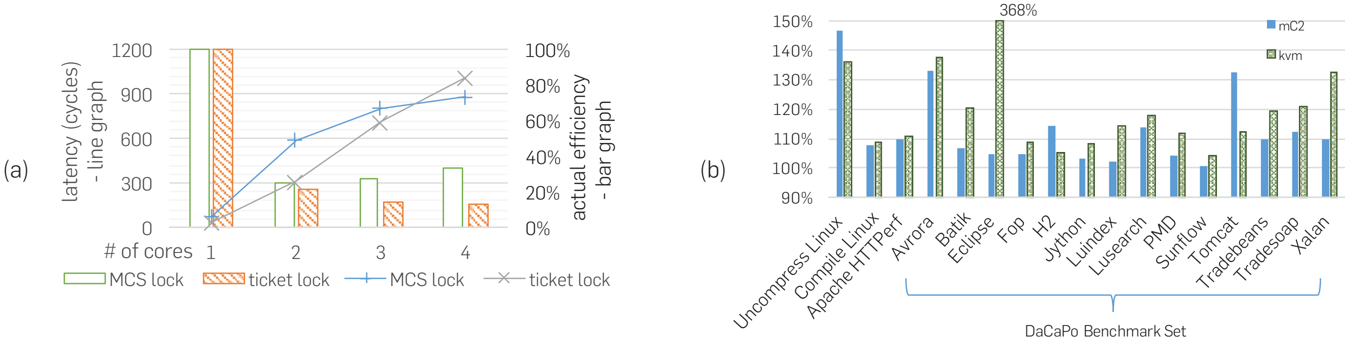
6. RELATED WORK

Dijkstra[9] proposed to "realize" a complex program by decomposing it into a hierarchy of linearly ordered abstract machines. Based on this idea, Gu et al.[10] developed new languages and tools for building certified abstraction layers with *deep* specifications, and showed how to apply the layered methodology to construct certified (sequential) OS kernels in Coq. Costanzo et al.[8] showed how to prove security properties over a deep specification of a certified OS kernel, and then propagate these properties from the specification level to its correct assembly-level implementation. Chen et al.[5] extended the layer methodology to build certified kernels and device drivers running on multiple *logical* CPUs. They treated the driver stack for each device as if it were running on a logical CPU dedicated to that device. Logical CPUs do not share any memory, and are all eventually mapped onto a single physical CPU. None of these systems, however, can support shared-memory concurrency with fine-grained locking.

The seL4 team[17] was the first to verify the functional correctness and security properties of a high-performance L4-family microkernel. The seL4 microkernel, however, does not support multicore concurrency with fine-grained locking. Peters et al.[22] and von Tessin[23] argued that for an seL4-like microkernel, concurrent data accesses across multiple CPUs can be reduced to a minimum, so a single *big kernel lock (BKL)* might be good enough for achieving good performance on multicore machines. von Tessin[23] further showed how to convert the single-core seL4 proofs into proofs for a BKL-based clustered multikernel.

The Verisoft team[1] applied the VCC framework[7] to formally verify Hyper-V, which is a widely deployed multiprocessor hypervisor consisting of 100 kLOC of C code and 5 kLOC of assembly. However, only 20% of the code is verified[7]; it is also only verified for function contracts and type invariants, rather than the full functional correctness property. CIVL[14] uses the state-machine approach with support for atomic actions and movers to reduce the proof burden for concurrent programs. It is implemented as an extension to Boogie and has been used to verify a concurrent garbage collector. However, CIVL can only be used to reason about safety rather than liveness. There is a large body of other work[6, 13, 24] showing how to build verified OS kernels,

Figure 8. (a) The comparison between actual efficiency of ticket lock and MCS lock implementations in mC2; (b) normalized performance for macro benchmarks running over Linux on KVM versus Linux on mC2; the baseline is Linux on bare metal; a smaller ratio is better.



hypervisors, file systems, device drivers, and distributed systems, but they do not address the issues of shared memory concurrency.

7. CONCLUSION

We have presented a novel extensible architecture for building certified concurrent OS kernels that not only have an efficient assembly implementation, but also machine-checkable contextual correctness proofs. OS kernels developed using our layered methodology also come with a clean, rigorous, and layered specification of all kernel components. We show that building certified concurrent kernels is not only feasible but also quite practical. Our layered approach to certified concurrent kernels replaces the hardware-enforced "red line" with a large number of abstraction layers enforced via formal specification and proofs. We believe this will open up a whole new dimension of research efforts toward building truly reliable, secure, and extensible system software.

Acknowledgments

We would like to acknowledge the contribution of many former and current team members on various CertiKOS-related projects at Yale, especially Tahina Ramananandro, Shu-Chun Weng, Liang Gu, Mengqi Liu, Quentin Carbonneaux, Jan Hoffmann, Hernán Vanzetto, Bryan Ford, Haozhong Zhang, and Yu Guo. We thank Xupeng Li, John Zhuang Hui, Xuguang Patrick Dai, members of the VeriGu lab at Columbia, and anonymous referees for helpful comments and suggestions that improved this research and the implemented tools. This research is based on the work supported in part by NSF grants 1065451, 1521523, and 1715154 and DARPA grants FA8750–12–2–0293, FA8750–16–2–0274, and FA8750–15-C-0082. It is also supported in part by Qtum Foundation and Baidu USA. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

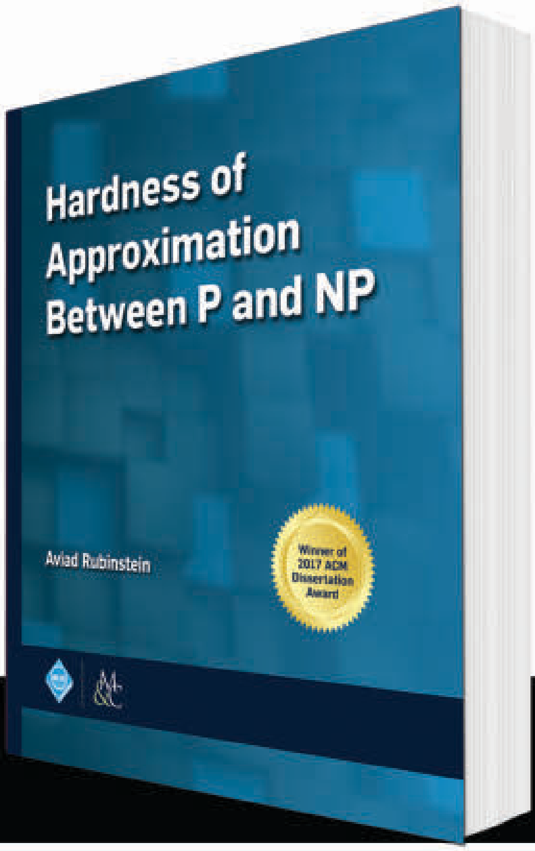
References

1. Alkassar, E., Hillebrand, M.A., Paul, W.J., Petrova, E. Automated verification of a small hypervisor. In *Proceedings of 3rd International Conference on Verified Software: Theories, Tools, Experiments (VSTTE)* (2010), 40–54.
2. Anderson, T., Dahlin, M. *Operating Systems Principles and Practice.* Recursive Books, 2011 (Figure 5.14).
3. Belay, A., Bittau, A., Mashtizadeh, A., Mazières, D., Kozyrakis, C. Dune: Safe user-level access to privileged CPU features. In *Proceedings of 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI'12)* (2012), 335–348.
4. Carbonneaux, Q., Hoffmann, J., Ramananandro, T., Shao, Z. End-to-end verification of stack-space bounds for C programs. In *Proceedings of 2014 ACM Conference on Programming Language Design and Implementation (PLDI'14)* (2014), 270–281.
5. Chen, H., Wu, X., Shao, Z., Lockerman, J., Gu, R. Toward compositional verification of interruptible OS kernels and device drivers. In *Proceedings of 2016 ACM Conference on Programming Language Design and Implementation (PLDI'16)* (2016), 431–447.
6. Chen, H., Ziegler, D., Chajed, T., Chlipala, A., Kaashoek, M.F., Zeldovich, N. Using Crash Hoare logic for certifying the FSCQ file system. In *Proceedings of 25th ACM Symposium on Operating System Principles (SOSP)* (2015), 18–37.
7. Cohen, E., Dahlweid, M., Hillebrand, M., Leinenbach, D., Moskal, M., Santen, T., Schulte, W., Tobies, S. VCC: A practical system for verifying concurrent C. In *Proceedings of 22nd International Conference on Theorem Proving in Higher Order Logics* (2009), 23–42.
8. Costanzo, D., Shao, Z., Gu, R. End-to-end verification of information-flow security for C and assembly programs. In *Proceedings of 2016 ACM Conference on Programming Language Design and Implementation (PLDI'16)* (2016), 648–664.
9. Dijkstra, E.W. The structure of the "THE"-multiprogramming system. *Commun. ACM*, (1968), 341–346.
10. Gu, R., Koenig, J., Ramananandro, T., Shao, Z., Wu, X., Weng, S.-C., Zhang, H., Guo, Y. Deep specifications and certified abstraction layers. In *Proceedings of 42nd ACM Symposium on Principles of Programming Languages (POPL'15)* (2015), 595–608.
11. Gu, R., Shao, Z., Chen, H., Wu, X.N., Kim, J., Sjöberg, V., Costanzo, D. Certikos: An extensible architecture for building certified concurrent OS kernels. In *Proceedings of 12thUSENIX Symposium on Operating Systems Design and Implementation (OSDI'16)* (2016), 653–669.
12. Gu, R., Shao, Z., Kim, J., Wu, X.N., Koenig, J., Sjöberg, V., Chen, H., Costanzo, D., Ramananandro, T. Certified concurrent abstraction layers. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation* (2018), ACM, 646–661.
13. Hawblitzel, C., Howell, J., Lorch, J.R., Narayan, A., Parno, B., Zhang, D., Zill, B. Ironclad apps: End-to-end security via automated full-system verification. In *Proceedings of 11thUSENIX Symposium on Operating Systems Design and Implementation (OSDI'14)* (2014), 165–181
14. Hawblitzel, C., Petrank, E., Qadeer, S., Tasiran, S. Automated and modular refinement reasoning for concurrent programs. In *International Conference on Computer Aided Verification* (2015), Springer, 449–465.
15. Herlihy, M., Shavit, N. *The Art of Multiprocessor Programming.* Morgan Kaufmann, 2008.
16. Kim, J., Sjöberg, V., Gu, R., Shao, Z. Safety and liveness of MCS lock—Layer by layer. In *Asian Symposium on Programming Languages andSystems* (2017), Springer, 273–297.
17. Klein, G., Elphinstone, K., Heiser, G., Andronick, J., Cock, D., Derrin, P., Elkaduwe, D., Engelhardt, K., Kolanski, R., Norrish, M., Sewell, T., Tuch, H., Winwood, S. seL4: Formal verification of an OS kernel. In *Proceedings of 22nd ACM Symposium on Operating Systems Principles (SOSP)* (2009), ACM, 207–220.
18. Lampson, B.W. Experience with processes and monitors in Mesa. *Commun. ACM 23*, 2 (1980).
19. Leroy, X. Formal verification of a realistic compiler. *Commun. ACM 52*, 7 (2009), 107–115.
20. Liang, H., Feng, X. A program logic for concurrent objects under fair scheduling. In *Proceedings of 43rd ACM Symposium on Principles of Programming Languages (POPL'16)* (2016), 385–399.
21. Mellor-Crummey, J.M., Scott, M.L. Algorithms for scalable synchronization on shared-memory multiprocessors. *ACM T. Comput. Syst. 9*, 1 (1991), 21–65.
22. Peters, S., Danis, A., Elphinstone, K., Heiser, G. For a microkernel, a big lock is fine. In *APSys '15 Asia Pacific Workshop on Systems, Tokyo, Japan* (2015).
23. von Tessin, M. *The clustered multikernel: An approach to formal verification of multiprocessor operating-system kernels.* PhD thesis, School of Computer Science and Engineering, The University of New South Wales (2013).
24. Xu, F., Fu, M., Feng, X., Zhang, X., Zhang, H., Li, Z. A practical verification framework for preemptive OS kernels. In *Proceedings of 28th International Conference on Computer-Aided Verification (CAV), Part II* (2016), 59–79.

Ronghui Gu (ronghui.gu@columbia.edu), Columbia University, New York, NY, USA..

Zhong Shao, Hao Chen, Jieung Kim, Jérémie Koenig, Xiongnan (Newman) Wu, and David Costanzo ({zhong.shao,hao.chen,jieung.kim,jeremie.koenig, xiongnan.wu,david.costanzo}@yale.edu), Yale University, New Haven, CT, USA.

Vilhelm Sjöberg (vilhelm.sjoberg@certik.org), CertiK, Cambridge, MA, USA.


Hardness of Approximation Between P and NP
Aviad Rubinstein
Winner of 2017 ACM Dissertation Award

CAREERS

California State University, San Bernardino

Tenure-Track Positions at the Assistant Professor Level

The School of Computer Science and Engineering at California State University, San Bernardino invites applications for TWO (2) tenure-track positions at the Assistant Professor level, beginning August 2020. All areas of Computer Science will be considered, including software engineering, parallel computing, algorithms, theory of computation, networking, and databases.

The School of CSE offers the programs of B.S. in Computer Science (ABET accredited), B.S. in Computer Engineering (ABET accredited), B.S. in Bioinformatics, B.A. in Computer Systems, and M.S. in Computer Science.

California State University, San Bernardino (CSUSB) is located in San Bernardino in the Inland Empire, 60 miles east of Los Angeles and operates a satellite campus in Palm Desert located in Coachella Valley. CSUSB serves approximately 20,000 students, of which 81% are first-generation college students, and graduates about 5,000 students annually. As a designated Hispanic Serving Institution, CSUSB reflects the dynamic diversity of the region and has one of the most diverse student populations of any university in the Inland Empire, and the second highest Hispanic enrollment of all public universities in California. CSUSB employs 467 full-time faculty and offers 48 undergraduate, 35 graduate, and 1 doctoral degree programs and 14 academic programs with national accreditation.

At CSUSB, diversity, equity and inclusion are values central to our mission. We recognize that diversity and inclusion in all its forms are necessary for our institutional success. By fully leveraging our diverse experiences, backgrounds and insights, we inspire innovation, challenge the status quo and create better outcomes for our students and community. As part of CSUSB's commitment to hire, develop and retain a diverse faculty, we offer a variety of networking, mentoring and development programs for our junior faculty. We are committed to building and sustaining a CSUSB community that is supportive and inclusive of all individuals. Qualified applicants with experience in ethnically diverse settings and/or who demonstrate a commitment to serving diverse student populations are strongly encouraged to apply. We also strongly encourage women and members of other underrepresented groups to apply.

Typical Activities
The two positions are to support the B.S. and M.S. programs in Computer Science. The candidate must display potential for excellence in teaching and scholarly work. The candidate is expected to supervise student research at both the undergraduate and graduate levels, and to actively participate in other types of academic student advising. The candidate will actively contribute to the School's curriculum development. The candidate will serve the School, College and University, as well as the community and the profession.

Minimum Qualifications
A Ph.D. in Computer Science or a closely related field is required by time of appointment

APPLICATION PROCESS:
Please submit the following documents at https://www.schooljobs.com/careers/csusb/jobs/2535257.

Formal review of applications will begin Nov 1st, 2019 and continue until the position is filled.

1) Curriculum Vitae
2) Cover Letter that includes:
   a. A statement of your teaching philosophy.
   b. A description of your research interests.
   c. A statement of how you might contribute to CSUSB's Strategic Plan (available at https://www.csusb.edu/strategic-plan).
3) If available, evidence of teaching effectiveness such as teaching portfolios, reports on teaching observations, and/or student evaluations of teaching.
4) Unofficial copies of all postsecondary degree transcripts (official transcripts will be required prior to appointment).
5) Reference List - names, telephone numbers, and email addresses of three (3) referees whom we may contact to obtain letters of recommendation.
6) A Diversity Statement, which may include your interpretation of diversity, equity, and inclusion, and must include specific examples of how your background and your educational and/or professional experiences have prepared you for this role at California State University, San Bernardino (maximum 1,000 words).

Questions about this position can be directed to Dr. Haiyan Qiao, Director of School of Computer Science and Engineering, at hqiao@csusb.edu.

Massachusetts Institute of Technology

Faculty Positions

The *Massachusetts Institute of Technology (MIT)* Department of Electrical Engineering and Computer Science (EECS) seeks candidates for faculty positions starting in July 1, 2020, or on a mutually agreed date thereafter. Appointment will be at the assistant or untenured associate professor level. In special cases, a senior faculty appointment may be possible. Faculty duties include teaching at the undergraduate and graduate levels, research, and supervision of student research. Candidates should hold a Ph.D. in electrical engineering and computer science or a related field by the start of employment. We will consider candidates with research and teaching interests in any area of electrical engineering and computer science.

Candidates must register with the EECS search website at https://school-of-engineering-faculty-search.mit.edu/eecs/, and must submit application materials electronically to this website. Candidate applications should include a description of professional interests and goals in both teaching and research. Each application should include a curriculum vitae and the names and addresses of three or more individuals who will provide letters of recommendation. Letter writers should submit their letters directly to MIT, preferably on the website or by mailing to the address below. Complete applications should be received by December 1, 2019. Applications will be considered complete only when both the applicant materials and **at least three letters of recommendation are received**.

It is the responsibility of the candidate to arrange reference letters to be uploaded at https://school-of-engineering-faculty-search.mit.edu/eecs/ by December 1, 2019.

Send all materials not submitted on the website to:

Professor Asu Ozdaglar
Department Head, Electrical Engineering
and Computer Science
Massachusetts Institute of Technology
Room 38-403
77 Massachusetts Avenue
Cambridge, MA 02139

M.I.T. is an equal opportunity/affirmative action employer.

Oregon State University

College of Engineering

Faculty Positions in Artificial Intelligence

The School of Electrical Engineering and Computer Science at Oregon State University invites applications for full-time, nine-month, tenure-track faculty positions in Artificial Intelligence to begin in Fall 2020. Candidates with a strong research record in any areas of Artificial Intelligence including Natural Language Processing, Computer Vision, Machine Learning, and Automated Planning will be considered.

Appointment is anticipated at the Assistant Professor rank, but candidates with exceptional qualifications may be considered for appointment at the rank of Associate or Full Professor. Applicants must hold a doctorate degree in Artificial Intelligence, Machine Learning, Computer Science, or a closely related field by the start of employment. Applicants should demonstrate a strong commitment and capacity to initiate new funded research as well as to expand, complement, and collaborate with existing research programs in the OSU College of Engineering and beyond. As part of the position, applicants have to regularly perform graduate and undergraduate teaching duties, including developing new courses related to their research expertise. Applicants are expected to mentor students and promote equitable outcomes among learners of diverse and underrepresented identity groups.

The university is located in Corvallis, at the heart of Oregon's Willamette Valley and close to Portland's Silicon Forest with numerous collaboration opportunities. The College of Engineering (CoE) boasts of strong graduate programs in Robotics and AI and a newly established Collaborative Robotics and Intelligent Systems Institute

(CoRIS). Corvallis has been ranked # 1 on a list of "Best Places for Work-Life Balance" and is within easy reach of the Cascade Mountains and the Oregon Coast.

Oregon State University has a strong institutional commitment to diversity and multiculturalism, and provides a welcoming atmosphere with unique professional opportunities for leaders from underrepresented groups. We are an Affirmative Action/Equal Opportunity employer, and particularly encourage applications from members of historically underrepresented racial/ethnic groups, women, individuals with disabilities, veterans, LGBTQ community members, and others who share our vision of an inclusive community.

Apply online at http://jobs.oregonstate.edu/postings/79556 (Posting #: P03142UF) with the following documents: A letter of interest; vita; a two-page statement of research interests; a one-page statement of teaching interests; a one-page statement on efforts towards equity and inclusion; and names and contact information for at least three references.

To be assured full consideration, applications must be received by December 1, 2019.

Oregon State University

College of Engineering

Multiple Faculty Positions in Cybersecurity

The School of Electrical Engineering and Computer Science at Oregon State University invites applications for two or more full-time, nine-month, tenure-track faculty positions in any area of cybersecurity including but not limited to systems security (operating systems, distributed systems, networked systems, embedded systems, real-time systems, cyber-physical systems, and energy delivery systems), hardware security, software security, privacy, cryptography and usable security. Appointment will start in Fall 2020 and is anticipated at the Assistant Professor rank, but candidates with exceptional qualifications may be considered for appointment at the rank of Associate or Full Professor. Applicants must hold a Ph.D. degree in Computer Science, Electrical and Computer Engineering, or closely related discipline, and should demonstrate a strong commitment and capacity to initiate new funded research as well as to expand, complement, and collaborate with existing research programs in the OSU College of Engineering and beyond. Furthermore, applicants should demonstrate a strong commitment to undergraduate and graduate teaching, including developing new courses related to their research expertise. Applicants are also expected to mentor students and promote equitable outcomes among learners of diverse and underrepresented identity groups.

Corvallis has been ranked #1 on a list of "Best Places for Work-Life Balance", and is within easy reach of Portland, Eugene, the Cascade mountain range, and the Oregon Coast. Oregon State University has a strong institutional commitment to diversity and multiculturalism, and provides a welcoming atmosphere with unique professional opportunities for leaders from underrepresented groups. OSU seeks diversity as a source of enrichment for our university community. We are an Affirmative Action/Equal Opportunity employer, and particularly encourage applications from members of historically underrepresented racial/ethnic groups, women, individuals with disabilities, veterans, LGBTQ community members, and others who share our vision of an inclusive community. The College of Engineering ranks high nationally in terms of the percentage of women faculty, and the university supports dual-career applications.

Apply online at https://jobs.oregonstate.edu/postings/80403 (posting #P03157UF) with the following documents: A letter of interest; vita; a two-page statement of research interests; a one-page statement of teaching interests; a one-page statement on efforts towards equity and inclusion; and names and contact information for at least three references. To be assured full consideration, applications must be received by November 15, 2019.

Southern University of Science and Technology (SUSTech)

Faculty Positions 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.

The Department of Computer Science and Engineering at SUSTech was founded in 2016. It has 33 professors, all of whom hold doctoral degrees or have years of experience in overseas universities. Among them, three are IEEE fellows; one IET fellow. The department is expected to grow to 50 tenure track faculty members eventually, in addition to teaching-only professors and research-only professors.

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.

University of Minnesota

Two Tenure-Track Faculty Positions

The Department of Industrial and Systems Engineering at the University of Minnesota invites applications for two full-time, tenure-track faculty positions starting in fall 2020. Applicants at all ranks will be considered.

We seek candidates with a strong methodological foundation in Operations Research (OR) and Industrial Engineering (IE). Individuals whose interests and backgrounds are at the intersection of OR/IE with other fields such as Computer Science, Statistics, and Economics, are encouraged to apply. Applicants should have a demonstrated interest in applications including, but not limited to: business analytics, energy systems and the environment, the sharing economy and digital marketplaces, healthcare delivery and medical decision making, smart transportation and connected cities, logistics, supply chains and manufacturing operations, data-driven decision making, and data science.

Applicants should hold or expect to complete by fall 2020 a Ph.D. in Industrial Engineering, Operations Research, Operations Management, or a related discipline and have demonstrated the potential to conduct a vigorous and significant research program as evidenced by their publication record and supporting letters from recognized leaders in the field. The candidate's expertise and documented research activities must demonstrate a strong potential to enhance both the department's research and teaching missions. Successful candidates are expected to build strong, externally funded, highly visible research programs and to become recognized leaders in their field.

The University of Minnesota is located in the heart of the vibrant Minneapolis-St. Paul metropolitan area, which is consistently rated as one of America's best places to live and is home to many leading companies. The Department of Industrial and Systems Engineering is within the College of Science and Engineering at the University of Minnesota.

Applicants are encouraged to apply by November 1, 2019 Review of applications will begin immediately and will continue until the position is filled. Additional information and application instructions can be found at http://www.isye.umn.edu. We particularly welcome applications from candidates from diverse cultures and communities because we believe that diversity helps broaden perspectives and enriches classroom and research experiences within the department and University. The University of Minnesota is an equal opportunity educator and employer.

[CONTINUED FROM P. 104] If Marie takes A1, then Joan gets all of B or 1 1/3 kilograms. If Marie declines to choose, then Joan gets all 2/3 from A and then Joan divides B equally into 2/3 kilogram for B1 and 2/3 kilogram for B2. This second case yields 2/3 + 2/3 for Joan or 1 1/3 kilograms. Of course, 1 1/3 > 1 1/4, so Joan is better off with this division.

Question: Prove this is optimal for Joan.

Solution: To see this is optimal for Joan, suppose A weighed less than 2/3 of a kilogram. In that case, Marie could decline to choose among A1 and A2 and then get half of pile B. Pile B would weigh more than 4/3 kilograms, so Marie would end up with more than 2/3 kilograms, leaving Joan with less than 1 1/3.

Now suppose A weighs w = 2/3 + e kilograms. In that case, if either A1 or A2 weighed more than 2/3 of a kilogram, Marie would take it, leaving Joan with less than 1 1/3. If both A1 and A2 weigh less than 2/3 of a kilogram, then Marie would decline to choose and would receive some portion we will call w2. Then Marie would receive w2 plus at least half of B. Because B weighs 2 - w, this comes out to w2 + 1/2 (2 - w). Joan then receives (w - w2) + 1/2 (2 - w). Now, we know that w = 2/3 + e, where e is a positive weight and we know that w - w2 <= 2/3 (because otherwise Marie would have chosen the subpile weighing more than 2/3). So, w - w2 = 2/3 + e - w2 <= 2/3. Rewriting, we get e <= w2. Now let's look at Joan's inheritance (w - w2) + 1/2 (2 - w) = (2/3 + e - w2) + 1 - (1/3 + e/2) = 1 1/3 + e/2 - w2. Because w2 >= e, w2 > e/2 and so the expression e/2 - w2 is negative. Thus, Joan receives less than 1 1/3. Let's check this with the case where e = 1/3 (so the two piles are equal). Joan would receive 1 1/3 + 1/6 - 1/4 = 1 1/2 - 1/4 = 1 1/4.

Upstart: How does this generalize to k kilograms and k piles where Marie gets to choose (a) 1 time, (b) k–1 times, or (c) 1 < m < k–1 times?

All are invited to submit their solutions to upstartpuzzles@cacm.acm.org; solutions to upstarts and discussion will be posted at http://cs.nyu.edu/cs/faculty/shasha/papers/cacmpuzzles.html

Dennis Shasha (dennisshasha@yahoo.com) is a professor of computer science in the Computer Science Department of the Courant Institute at New York University, New York, USA, as well as the chronicler of his good friend the omniheuristic Dr. Ecco.

DOI:10.1145/3356582

Dennis Shasha

Upstart Puzzles
Dust Wars

Considering willful approaches to a golden opportunity.

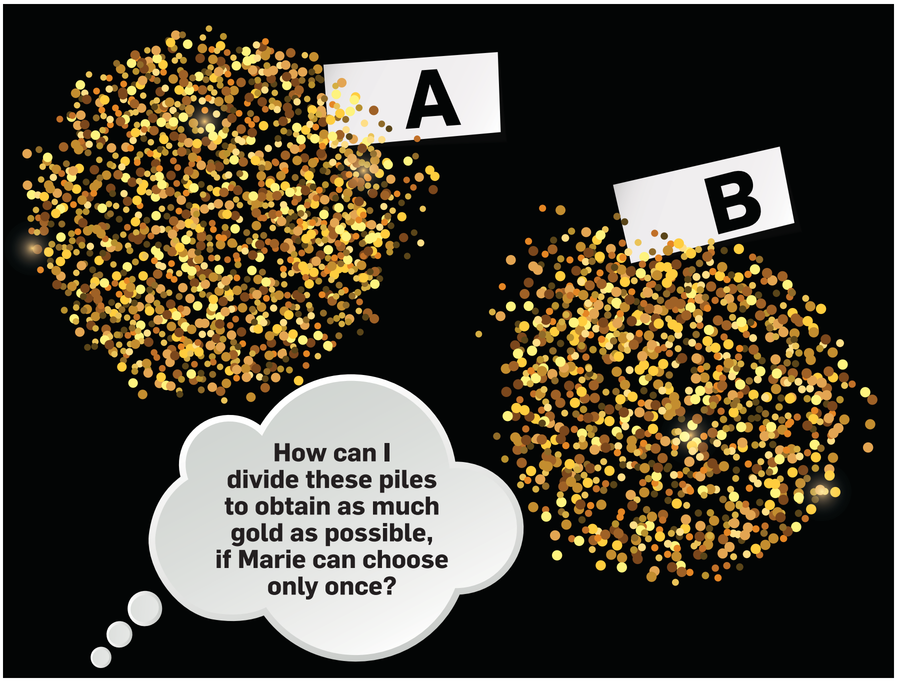

A
B
How can I divide these piles to obtain as much gold as possible, if Marie can choose only once?

IMAGE BY ANDRIJ BORYS ASSOCIATES

WHEN THEIR PARENTS die in a tragic accident, daughters Joan and Marie read their parents' Last Will and Testament. The Will is very short, because there is only one asset: two kilograms of gold dust.

The Will states that Joan, as the elder sister, should divide the dust into two piles: we will call those piles A and B. Then she is to cut pile A into two smaller piles that we will call A1 and A2. Marie can decide to choose one of A1 or A2 or not. If Marie chooses, then Joan takes the other smaller pile and all of pile B. If Marie does not choose between A1 and A2, then Joan can choose one of them (presumably the larger one) and give Marie the other one and then Joan must cut pile B into B1 and B2 and Marie can choose which one she wants.

Joan and Marie, though clever mathematicians, have never gotten along. Each wants as much gold dust as possible while obeying the rules of the Will.

Warm-Up: Suppose Joan divides the two kilograms equally (as depicted in the figure), so pile A weighs one kilogram as does B. How much can Joan be sure to receive as part of her inheritance no matter how clever Marie is?

Solution to Warm-Up: If piles A and B each weighs 1 kilogram, then Joan can guarantee to get 1 1/4 kilograms of gold dust. Here is how: she divides the A pile into subpile A1 consisting of 3/4 kilogram and subpile A2 consisting of 1/4 kilogram. If Marie chooses A1, then Joan gets A2 and all of B, thus 1 1/4 kilograms. If Marie declines to choose either A1 or A2, then Joan gives Marie A2 and divides pile B into B1 consisting of 1/2 kilogram and B2 also consisting of 1/2 kilogram. No matter which one Marie chooses, Marie will get 1/4 +1/2, leaving Joan with 1 1/4.

Question: Prove Joan cannot do any better if the two piles are equal.

Solution: To see intuitively that Joan cannot do any better, suppose she divides up pile A into more unequal portions, say 7/8 in A1 and 1/8 in A2. In that case, Marie takes A1 and Joan receives only 1 1/8 in total. Suppose Joan divides pile A into less unequal portions, say 5/8 in A1 and 3/8 in A2. In that case, Marie declines to choose among A1 and A2. Joan then takes A1 but now Joan must divide up B into equal portions (otherwise Marie will take the larger portion). So, Joan would accumulate only 5/8 + 1/2 = 9/8 = 1 1/8.

Joan realizes she is not obligated to divide up the two kilograms into two equal piles. Clearly, she does not want them to be too unequal. For example, if A were tiny, then Marie would simply not choose between A1 and A2 and get 1/2 of B yielding her nearly 1 kilogram. That would leave Joan with approximately the same amount of gold dust.

Question: Can she do better by dividing them in some other way?

Solution: However, suppose Joan divides the two so that A is 2/3 kilograms and B is 4/3 (or 1 1/3) kilograms. Then Joan divides A into A1 consisting of all 2/3 kilograms and A2 consisting of one piece of dust.

[CONTINUED ON P. 103]

DREAM ZONE!

SIGGRAPH ASIA 2019 BRISBANE

The 12th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia

Conference 17 - 20 November 2019
Exhibition 18 - 20 November 2019

Brisbane Convention & Exhibition Centre (BCEC), Brisbane, Australia

Sponsored by:

acm

Organized by

koelnmesse
we energize your business | since 1924