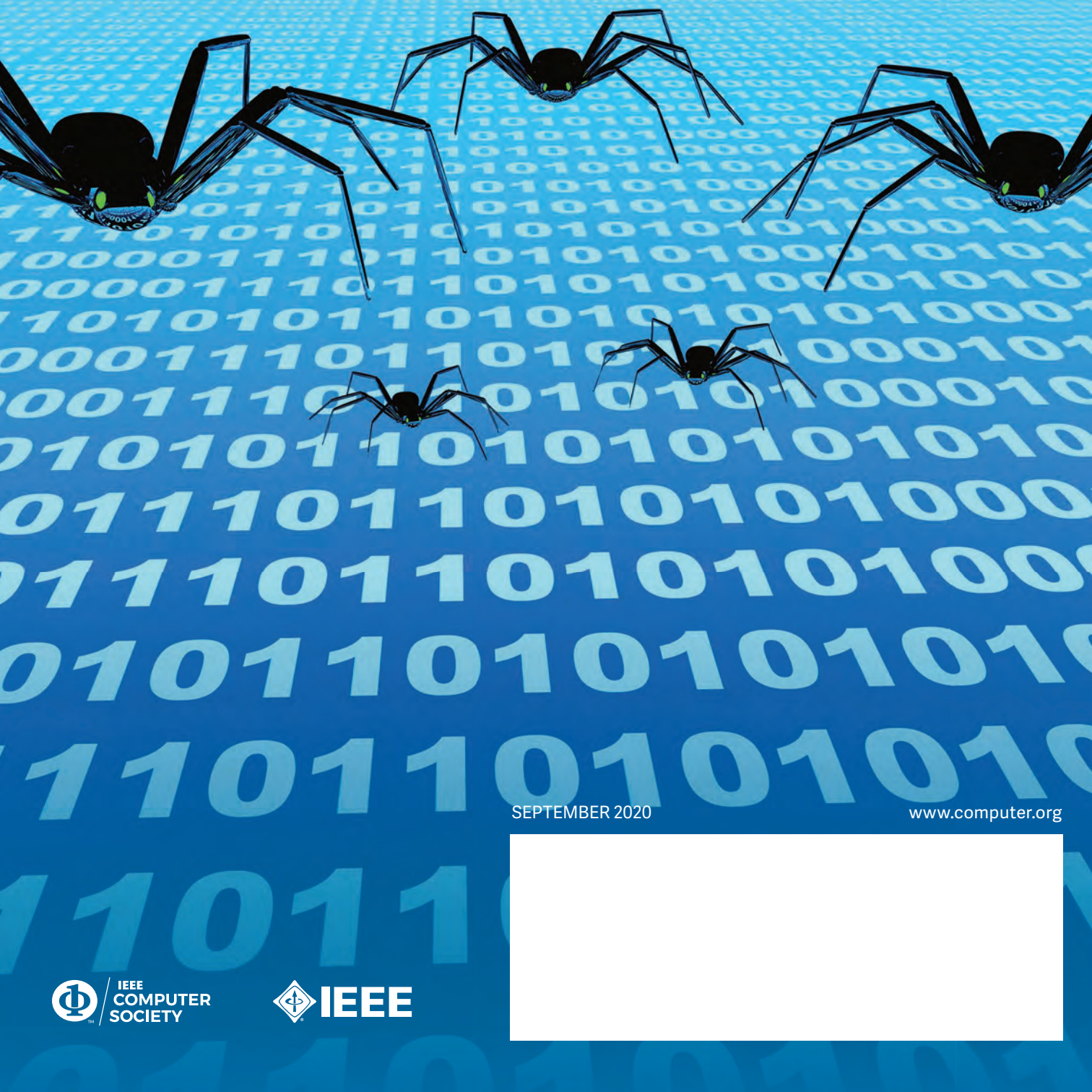


COMPUTING edge

- Software
- Machine Learning
- Multimedia
- Quantum Computing



SEPTEMBER 2020

www.computer.org

IEEE Computer Society Has You Covered!

WORLD-CLASS CONFERENCES — Stay ahead of the curve by attending one of our 200+ globally recognized conferences.

DIGITAL LIBRARY — Easily access over 780k articles covering world-class peer-reviewed content in the IEEE Computer Society Digital Library.

CALLS FOR PAPERS — Discover opportunities to write and present your ground-breaking accomplishments.

EDUCATION — Strengthen your resume with the IEEE Computer Society Course Catalog and its range of offerings.

ADVANCE YOUR CAREER — Search the new positions posted in the IEEE Computer Society Jobs Board.

NETWORK — Make connections that count by participating in local Region, Section, and Chapter activities.

Explore all of the member benefits at www.computer.org today!



STAFF

Editor

Cathy Martin

Publications Operations Project Specialist

Christine Anthony

Production & Design Artist

Carmen Flores-Garvey

Publications Portfolio Managers

Carrie Clark, Kimberly Sperka

Publisher

Robin Baldwin

Senior Advertising Coordinator

Debbie Sims

Circulation: *ComputingEdge* (ISSN 2469-7087) is published monthly by the IEEE Computer Society, IEEE Headquarters, Three Park Avenue, 17th Floor, New York, NY 10016-5997; IEEE Computer Society Publications Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; voice +1 714 821 8380; fax +1 714 821 4010; IEEE Computer Society Headquarters, 2001 L Street NW, Suite 700, Washington, DC 20036.

Postmaster: Send address changes to *ComputingEdge*-IEEE Membership Processing Dept., 445 Hoes Lane, Piscataway, NJ 08855. Periodicals Postage Paid at New York, New York, and at additional mailing offices. Printed in USA.

Editorial: Unless otherwise stated, bylined articles, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *ComputingEdge* does not necessarily constitute endorsement by the IEEE or the Computer Society. All submissions are subject to editing for style, clarity, and space.

Reuse Rights and Reprint Permissions: Educational or personal use of this material is permitted without fee, provided such use: 1) is not made for profit; 2) includes this notice and a full citation to the original work on the first page of the copy; and 3) does not imply IEEE endorsement of any third-party products or services. Authors and their companies are permitted to post the accepted version of IEEE-copyrighted material on their own Web servers without permission, provided that the IEEE copyright notice and a full citation to the original work appear on the first screen of the posted copy. An accepted manuscript is a version which has been revised by the author to incorporate review suggestions, but not the published version with copy-editing, proofreading, and formatting added by IEEE. For more information, please go to: http://www.ieee.org/publications_standards/publications/rights/paperversionpolicy.html. Permission to reprint/republish this material for commercial, advertising, or promotional purposes or for creating new collective works for resale or redistribution must be obtained from IEEE by writing to the IEEE Intellectual Property Rights Office, 445 Hoes Lane, Piscataway, NJ 08854-4141 or pubs-permissions@ieee.org. Copyright © 2020 IEEE. All rights reserved.

Abstracting and Library Use: Abstracting is permitted with credit to the source. Libraries are permitted to photocopy for private use of patrons, provided the per-copy fee indicated in the code at the bottom of the first page is paid through the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923.

Unsubscribe: If you no longer wish to receive this *ComputingEdge* mailing, please email IEEE Computer Society Customer Service at help@computer.org and type "unsubscribe *ComputingEdge*" in your subject line.

IEEE prohibits discrimination, harassment, and bullying. For more information, visit www.ieee.org/web/aboutus/whatis/policies/p9-26.html.

IEEE Computer Society Magazine Editors in Chief

Computer

Jeff Voas, *NIST*

Computing in Science & Engineering

Lorena A. Barba (Interim),
George Washington University

IEEE Annals of the History of Computing

Gerardo Con Diaz, *University
of California, Davis*

IEEE Computer Graphics and Applications

Torsten Möller,
Universität Wien

IEEE Intelligent Systems

V.S. Subrahmanian,
Dartmouth College

IEEE Internet Computing

George Pallis, *University
of Cyprus*

IEEE Micro

Lizy Kurian John, *University
of Texas at Austin*

IEEE MultiMedia

Shu-Ching Chen, *Florida
International University*

IEEE Pervasive Computing

Marc Langheinrich, *Università
della Svizzera italiana*

IEEE Security & Privacy

David Nicol, *University
of Illinois at
Urbana-Champaign*

IEEE Software

Ipek Ozkaya, *Software
Engineering Institute*

IT Professional

Irena Bojanova, *NIST*

COMPUTING
edge



18

Code Mining

24

When
Multimedia
Meets Fashion

30

Learning to
Synthesize and
Manipulate
Natural Images



Software

8 The Rode0day to Less-Buggy Programs

ANDREW FASANO, TIM LEEK, BRENDAN DOLAN-GAVITT, AND JOSH BUNDT

14 Better Code Reviews With Design by Contract

GEORGE FAIRBANKS

Machine Learning

18 Code Mining

GERARD J. HOLZMANN

24 When Multimedia Meets Fashion

SIJIE SONG AND TAO MEI

Multimedia

30 Learning to Synthesize and Manipulate Natural Images

JUN-YAN ZHU

40 Multimedia for Disaster Information Management

SHU-CHING CHEN

Quantum Computing

42 The Quantum Moonshot

ERIK P. DEBENEDICTIS

48 The National Quantum Initiative Will Also Benefit Classical Computers

ERIK P. DEBENEDICTIS AND MICHAEL P. FRANK

Departments

4 Magazine Roundup

7 Editor's Note: Improving Code Reviews

64 Conference Calendar



Subscribe to *ComputingEdge* for free at
www.computer.org/computingedge.



Magazine Roundup

The IEEE Computer Society's lineup of 12 peer-reviewed technical magazines covers cutting-edge topics ranging from software design and computer graphics to Internet computing and security, from scientific applications and machine intelligence to visualization and microchip design. Here are highlights from recent issues.

Computer

Data-Centric Edge Computing to Defend Power Grids Against IoT-Based Attacks

The Internet of Things (IoT) introduces new attack surfaces to power grids through Wi-Fi-enabled high-wattage appliances, rendering security mechanisms ineffective. The authors of this article from the May 2020 issue of *Computer* propose a data-centric edge-computing infrastructure to host defend mechanisms in IoT clouds by integrating physical states in decentralized power-grid regions.

computing IN SCIENCE & ENGINEERING

Augmented and Virtual Reality in Surgery

Augmented and virtual reality are transforming the practice of healthcare by providing powerful and intuitive methods of exploring and interacting with digital medical data, as well as integrating data into the physical world to create natural and interactive virtual experiences. These immersive technologies use lightweight

stereoscopic head-mounted displays to place users into simulated and realistic 3D digital environments, unlocking significant benefits from the seamless integration of digital information with the healthcare practitioner and patient's experience. This article from the May/June 2020 issue of *Computing in Science & Engineering* explores some of the current and emerging technologies and applications in surgery; their benefits and challenges around immersion, spatial awareness, and cognition; and their reported and projected use in learning environments, procedure planning, and perioperative contexts in the surgical theatre.

IEEE Annals of the History of Computing

The Advent of Digital Typography

This article from the January–March 2020 issue of *IEEE Annals of the History of Computing* presents an interview conducted with Liz Bond Crews. Crews began her career as a systems analyst for RCA. She soon joined the Xerox Electro-Optical Division. In her

final position at Xerox, she licensed high-quality typeface designs from Mergenthaler Linotype for use on Xerox's high-performance 9700 laser printer. In the early 1980s, the Adobe cofounder John Warnock recruited Crews to Adobe Systems. While at Adobe, she was responsible for bringing high-quality typography to the Adobe PostScript page description language, running corporate marketing, hiring graphics designers, and putting Adobe on a firm footing with professional designers and printers, greatly enhancing the value of their product. After Adobe, she helped found Electronics for Imaging and, after some consulting, retired in 1993.

IEEE Computer Graphics AND APPLICATIONS

High-Fidelity Point-Based Rendering of Large-Scale 3-D Scan Datasets

Digitalization of 3D objects and scenes using modern depth sensors and high-resolution RGB cameras enables the preservation of human cultural artifacts at an unprecedented level of detail. Interactive visualization of these



large datasets, however, is challenging without degradation in visual fidelity. A common solution is to fit the dataset into available video memory by downsampling and compression. The achievable reproduction accuracy is thereby limited for interactive scenarios, such as immersive exploration in virtual reality. This degradation in visual realism ultimately hinders the effective communication of human cultural knowledge. This article from the May/June 2020 issue of *IEEE Computer Graphics and Applications* presents a method to render 3D scan datasets with minimal loss of visual fidelity. A point-based rendering approach visualizes scan data as a dense splat cloud. For improved surface approximation of thin and sparsely sampled objects, the authors propose oriented 3D ellipsoids as rendering primitives.

IEEE Intelligent Systems

Stock Selection Model Based on Machine Learning with Wisdom of Experts and Crowds

Both stock recommendations from sell-side analysts and online user-generated content from crowds have great significance in the stock market. The authors of this article from the March/April

2020 issue of *IEEE Intelligent Systems* examine and compare different effects of analyst attitude and crowd sentiment on stock prices with data from CSMAR. By estimating a multivariate linear regression model, they find that although the wisdom of both experts and crowds has impact on stock prices, the latter's impact on stock prices prevails. They also adopt LightGBM, a novel machine-learning model, to predict stock trends based on empirical results. Portfolio returns of different models also suggest that crowd wisdom is more valuable for creating investment strategy than expert wisdom.

IEEE Internet Computing

Using Feature Fusion Strategies in Continuous Authentication on Smartphones

With the increasing prevalence of mobile devices, people prefer to use smartphones to make payments, take photos, and collect personal vital information. Due to the high possibility of smartphone illegal access, the security and privacy of the devices become more important. In this article from the March/April 2020 issue of *IEEE Internet Computing*, the authors present FusionAuth, a

sensor-based continuous authentication system leveraging the accelerometer, gyroscope, and magnetometer on smartphones to capture users' behavioral patterns. To improve the authentication performance and enhance system reliability, they utilize two feature fusion strategies of serial feature fusion and parallel feature fusion to combine the designed features from the three sensors in the feature extraction module. Based on the trained one-class support vector domain description classifier, they evaluate the authentication performance of FusionAuth in terms of impact of window size and user size, and accuracy on different users.

IEEE micro

RTX on—The NVIDIA Turing GPU

NVIDIA's latest processor family, the Turing GPU, was designed to realize a vision for next-generation graphics, combining rasterization, ray tracing, and deep learning. It includes fundamental advancements in several key areas: streaming multiprocessor efficiency, a Tensor Core for accelerated AI inferencing, and an RTCore for accelerated ray tracing. With these innovations, Turing unlocks both real-time ray-tracing performance

and deep-learning inference in consumer, professional, and data-center solutions. Read more in this article from the March/April 2020 issue of *IEEE Micro*.

IEEE MultiMedia

Glasses-Free 3-D and Augmented Reality Display Advances: From Theory to Implementation

In this article from the January–March 2020 issue of *IEEE MultiMedia*, the authors describe how four of the most viable glasses-free 3D display methods were chosen for their research at NTU’s Advanced Displays Laboratory. The authors developed three spatio-temporally multiplexed super multi-view display systems based on liquid crystal displays, organic light-emitting diodes, and fast projectors. This article explains the background, status of the prototypes, and their performance.



Accounting for Dynamic Diversity Among Child Users of IoT

As IoT becomes increasingly pervasive, children are more regularly encountering IoT. The recent GDPR legislation in Europe goes some of the way toward protecting children as they make use of these IoT technologies, but there remain significant challenges in ensuring that the IoT that pervades children’s lives is socially

responsible. This paper from the January–March 2020 issue of *IEEE Pervasive Computing* explores some of the reasons that the ethical implications of IoT for children are difficult to contain, and suggests several ways that design might make progress toward socially responsible IoT.



Stop Guessing: Using Guessed Passwords to Thwart Online Password Guessing

Practitioners who are seeking to defend against online password-guessing attacks usually find a shortage of techniques. The authors of this article from the May/June 2020 issue of *IEEE Security & Privacy* present an open source system to defend against password-guessing attacks using information ignored previously: the guessed passwords themselves. They examine passwords to detect login failures caused by users mistyping their passwords.



Innovative Practices for Knowledge Sharing in Large-Scale DevOps

Agile development methods and DevOps require adaptation during implementation to meet the needs of a constantly changing software development environment. The emergence of knowledge-sharing practices for large-scale DevOps has not been the subject of much research. This in-depth case study

from the May/June 2020 issue of *IEEE Software*, consisting of 106 interviews at a multinational company operating in a DevOps-at-scale environment, identifies a number of innovative practices.



Food for Thought: Fighting Fake News and Online Disinformation

Fake news is becoming a growing concern for several industrial sectors and society at large. In this article from the March/April 2020 issue of *IT Professional*, the authors explain why people are susceptible to fake news and what types of impact fake news can cause, taking examples from agri-food and other sectors. By providing a concise yet insightful overview in the field, the authors discuss how advances in machine learning and semantic technologies can be utilized to detect fake news and mitigate online disinformation. 🌐

Join the IEEE
Computer
Society

computer.org/join





Editor's Note

Improving Code Reviews

Manual peer code review is the software industry's standard method for catching bugs and design flaws. But too often, mistakes slip through. Recently, software practitioners have turned to new techniques for reviewing code. This issue of *ComputingEdge* presents two ways to improve code reviews—one that incorporates automated techniques and another that inserts more structure and objectivity into the process.

IEEE Security & Privacy's "The RodeOday to Less-Buggy Programs" discusses the advantages of automated bug-finding tools. The authors describe a monthly competition in which teams use these bug finders to catch errors in a corpus of programs, with the goals of evaluating the tools' effectiveness and identifying the best approaches. *IEEE Software's*

"Better Code Reviews With Design by Contract" details the design-by-contract method and argues that it not only makes the code review process smoother but also helps developers improve their design skills.

Machine learning can also be employed to improve software. *IEEE Software's* "Code Mining" considers whether machine learning could be used to evaluate the quality of program code. Meanwhile, *IEEE MultiMedia's* "When Multimedia Meets Fashion" shows how machine learning is contributing to better software in the fashion industry.

Machine learning has many uses in multimedia applications. The authors of "Learning to Synthesize and Manipulate Natural Images," from *IEEE Computer Graphics and Applications*, use machine learning to preserve

realism while editing photographs. "Multimedia for Disaster Information Management," from *IEEE MultiMedia*, posits that machine learning-based analysis of multimedia big data can lead to more effective disaster response.

This *ComputingEdge* issue closes with two *Computer* articles about the relationship between classical and quantum computing and the evolving definition of "computer." "The Quantum Moonshot" explores the shifts in education, training, and research that the computer science field must undergo to leverage quantum computing. "The National Quantum Initiative Will Also Benefit Classical Computers" argues that rethinking computer design to accommodate quantum will lead to advances that affect all of computing. 🌐

DEPARTMENT: SYSTEMS ATTACKS AND DEFENSES

The RodeOday to Less-Buggy Programs

Andrew Fasano, *MIT Lincoln Laboratory and Northeastern University*

Tim Leek, *MIT Lincoln Laboratory*

Brendan Dolan-Gavitt, *New York University*

Josh Bundt, *Army Cyber Institute and Northeastern University*

Despite their best efforts, computer programmers consistently fail to build the safe and reliable programs they imagine; rather, they typically build systems that work well until something unexpected happens. When presented with unexpected inputs, systems may reveal bugs in their code, which cause incorrect behavior or crashes. Every programmer knows that bugs are a persistent and costly issue in computer programs, but the path to getting rid of them is less clear.

Straightforward approaches such as manual code review are difficult to scale and often unsuccessful. Increasingly, developers are turning to automated bug-finding tools.

Static analyzers may automatically flag suspicious-looking code patterns or use more sophisticated analyses to discover potential vulnerabilities in large codebases. Another approach is to find bugs through dynamic testing (i.e., passing a large number of diverse inputs to a program and fixing any bugs that are discovered). This technique is overwhelmingly favored for both offensive and defensive security research, because this method is easy to perform with off-the-shelf tools, and it gives concrete results without false positives. Anyone can download an industrial-strength, coverage-guided fuzzer such as “American fuzzy lop” (AFL), and it will likely be able to test his or her programs. More importantly, this type of testing absolutely works; despite its easy setup, AFL

(<http://lcamtuf.coredump.cx/afl/>) often succeeds at finding bugs.

But as scientists and practitioners, we would like to know not just that a tool like AFL works, but how well it works. We want to gauge bug-finding merit and improvement to guide the use and development of these tools. Is AFL better than KLEE, which uses symbolic execution and Satisfiability Modulo Theory solving to find bugs? Is this new version of AFL better than the last one? To answer such questions, we propose the following approach to testing bug-finding systems:

1. Create a corpus of buggy programs.
 - › A corpus should contain multiple real-world programs.
 - › Each of the programs should contain diverse and realistic bugs.
 - › Each bug should manifest at a known location.
 - › An input to trigger each bug should be generated.
2. Conduct an evaluation of bug finders.
 - › At the start of an evaluation, all bug finders should be provided with the buggy programs from the same bug corpus. Other data in the corpus (e.g., triggering inputs and so on) should not be provided. Prior to the evaluation, this corpus must not be available in any form to developers of systems being evaluated.
 - › Each bug finder should be evaluated based upon its ability to find inputs that trigger distinct bugs in the corpus.

3. After each evaluation, the full corpus of buggy programs should be released to the public with a list of bugs, where they manifest, and the triggering inputs for each.
4. Evaluations should be run repeatedly and frequently.

Over the past year, we have been running Rode0day [a portmanteau of a rodeo and a 0-day vulnerability, pronounced “ro-dee-oh-day” (<http://rode0day.mit.edu>)], a monthly bug-finding competition and our first attempt at this kind of precise bug-finder evaluation. We think this competition satisfies all of our criteria, with the possible exception of realistic bugs.

BUG INJECTION

To evaluate bug-finding systems, we first need buggy programs. In fact, we need tons of buggy programs for every evaluation. Manually creating these programs clearly can’t scale to the demand, but fortunately, recent research has shown that buggy programs can be created automatically. Previously, we created two systems to do just that: LAVA (which stands for “large-scale automated vulnerability addition”) and Apocalypse,³ which both automatically inject bugs into C programs.

LAVA uses a dynamic taint analysis to identify program variables controllable by input data. New code is injected into the program, which, depending on what those variables are set to, can cause a crash. The taint analysis reveals how values in the input file affect program variables; this information is used to generate input files that likely trigger each bug. LAVA combines this analysis with an empirical test to see if an input actually causes a program to crash. LAVA was presented at the IEEE Symposium on Security and Privacy in 2016, and the corpora from this article, LAVA-1 and LAVA-M, were released later the same year. At that time, the combined detection rate for a state-of-the-art fuzzer and symbolic execution bug finder was 2%. Since then, 47% of the bug-finding systems presented at the top four academic computer security conferences have self-evaluated against the LAVA-M corpus. However, the expiration date of LAVA-M seems to have come and gone, as evidenced by the Angora fuzzer¹ finding all of the labeled bugs in this corpus (and a few unlabeled ones, too).

Apocalypse employs symbolic execution and constraint solving as well as program synthesis to achieve a similar end. The bugs inserted by Apocalypse are fewer in number but far more complicated and stateful than are the LAVA ones. Although these bugs are harder to generate and fewer in quantity, they may better reflect the complexity of some bugs in real software.

RODEODAY

With access to LAVA and Apocalypse, we decided to launch a recurring bug-finding competition for which we would release a corpus of buggy programs and challenge teams to find as many of the injected bugs as they could. Each competition would run for a month, during which time teams would try their best to find bugs. At the end of each competition, we would declare a winner and release inputs to trigger each of the injected bugs. Our vision of how this competition improves bug finding is shown in Figure 1.

At the start of each Rode0day, teams are given a collection of Linux programs, sample inputs, and instructions about how to run each program on the sample inputs. Teams use their bug-finding skills to generate inputs that cause target programs to crash. When a team believes they have generated such an input, they submit it to the Rode0day API and are scored immediately. For each new bug a team finds, they are awarded 10 points. If a team is the first to find a given bug, they are awarded a single bonus point. This encourages teams to submit their generated inputs as soon as they find them. As teams submit new inputs to the API, their scores are updated in real time on a scoreboard on the Rode0day website. Although we conceal from competitors the total number of bugs that were injected into a given challenge, the scoreboard does show the number of unique bugs found by all the teams, thus providing a lower bound.

In May 2018, we ran a closed beta Rode0day competition. Using LAVA, we injected 52 bugs into two simple C programs we had written: a trivial file parser and a simple key/value store. On average, our nine competitors found slightly more than half of the injected bugs. The first-place team, itszn, found every injected bug in roughly 5 h, an indication that future competitions should feature more substantial programs.

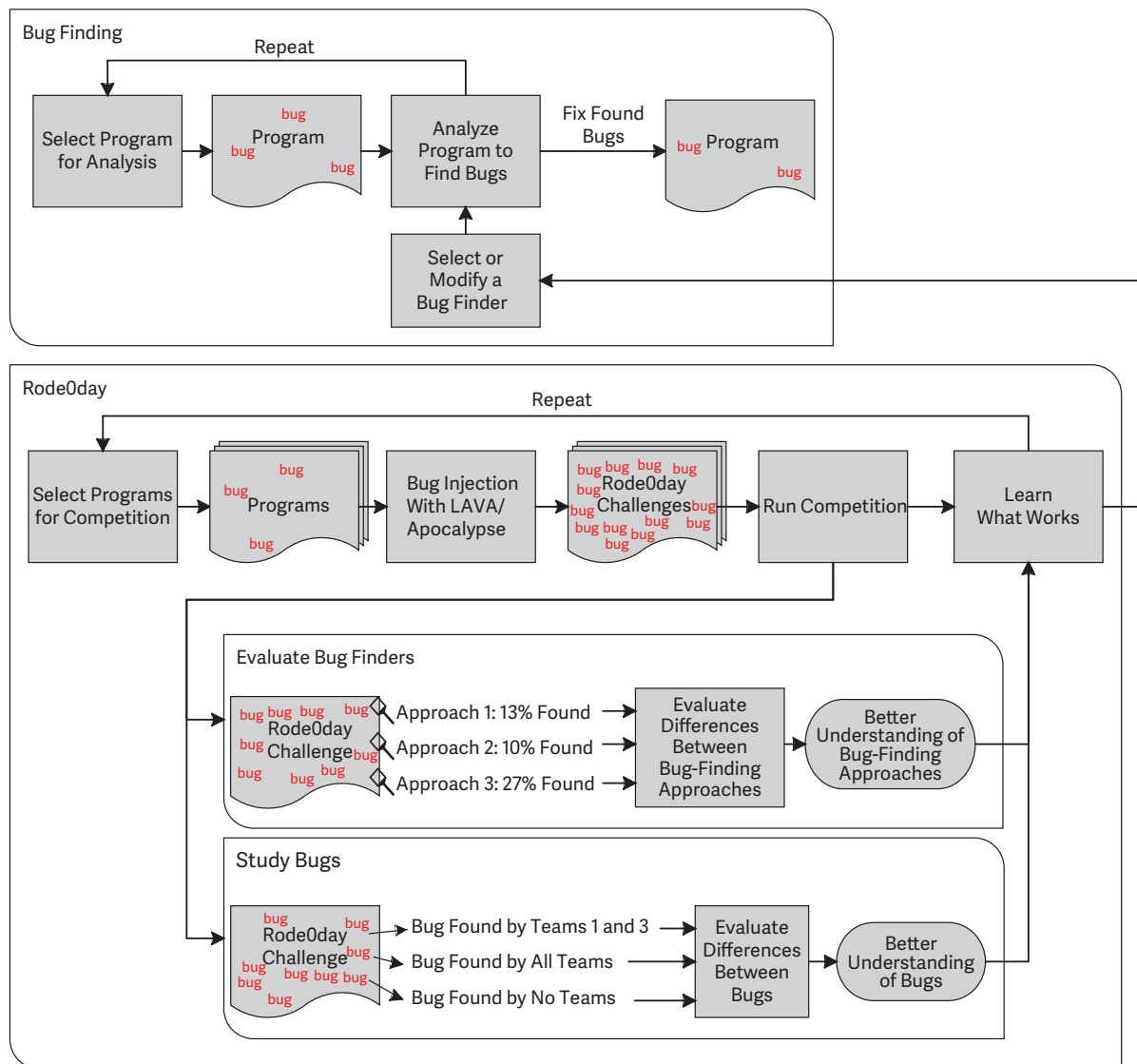


FIGURE 1. An example of how Rode0day improves bug finding.

We launched our first official Rode0day competition in July 2018 and have been running new competitions (nearly) every month since. At the time of this writing, we have run 10 competitions with 44 buggy programs containing a total of 2,103 bugs. These programs are generated from the real software displayed in Table 1. A total of 28 teams have scored points in these competitions.

RESULTS

Administering Rode0day provides us with a wealth of empirical data about what works and what doesn't for finding bugs. It also allows us to conduct controlled

experiments by introducing different types of bugs, varying the size of programs, and using different bug-injection strategies. Although we haven't yet conducted a full analysis of the data we've collected in the past year, in this article, we provide a sampling of some of our more interesting results.

One obvious question is: Which team is the best? Answering this question is not completely straightforward, because not every team has participated in every competition. To assess the relative performance of different teams, we used a version of the Elo rating system, which was developed to rank chess players. Specifically, we use the multiplayer variant of Elo

TABLE 1. The target programs used in Rode0day competitions.

Name	Description	C-SLOC*	Bugs injected	Bugs discovered
File	File parser	15,671	657	431
LibYAML	YAML parser	10,324	231	216
Libjpeg	JPEG parser	20,316	268	268
Bzip2	File compressor	5,820	28	28
Duktape	JavaScript engine	83,793	18	15
Grep	Regular expression matcher	70,030	80	78
Jq	JSON parser	16,897	332	210
Pcre2grep	Regular expression matcher	83,113	184	180
Sqlite	Database	12,902	56	24
Tinyexpr	Math engine	1,241	44	37

*Lines of C code as measured by David A. Wheeler's SLOCCount. SLOC: source lines of code.

described by Tom Kerrigan at http://www.tckerrigan.com/Misc/Multiplayer_Elo. In the Elo system, every player starts out with the same skill rating. When two players play a match against each other, the difference in their scores is used to calculate the expected outcome. The difference between the expected and actual outcome is then used to update their ratings. The rankings for our first year are shown in Table 2. The top-ranked team, afl-lazy, played in the last five competitions and won four of them. Despite being the best team to participate thus far, afl-lazy has found fewer than half the injected bugs across all the Rode0day competitions they participated in. Finding these injected bugs is clearly not a solved problem.

In addition to comparing team performances, we're also interested in studying what makes some bugs easy to find and others difficult. Figure 2 shows which bugs were found in a version of a file used in the November 2018 Rode0day. For this challenge, the 59 LAVA bugs injected were of two types: simple bugs, in which a memory safety violation occurs if a value from the input matches a constant; and complex bugs, in which a memory safety violation occurs if three values from the input satisfy a particular equation. Fifty percent of the 36 injected simple bugs and 70% of the 23 injected complex bugs were found by the six competing teams. Each column in Figure 2 corresponds to a

TABLE 2. The overall rankings for top Rode0day competitors after 10 competitions.

Place	Elo score	Team name
1	1,087	afl-lazy
2	1,069	itszn
3	1,027	H3ku
4	1,017	REDQUEEN
5	1,062	NU-AFL-QSYM

discovered bug. The color of each square indicates the time it took to find a bug: green squares indicate bugs found within 24 h, light blue squares indicate bugs that took just over a day, and darker blue squares took longer (up to 25 days).

Of these discovered bugs, some are clearly easier to find than others, as evidenced by multiple teams finding the same bugs within a day.

Of the 34 discovered bugs shown, the rightmost column represents a bug found by a single team after 25 days of searching. To understand some of the challenges involved in finding this bug, let's walk through what needs to occur for it to be triggered. The bug requires three separate program variables (i.e., *A*, *B*, and *C*) to be set to values copied from separate portions of the input file. If the values of *A*, *B*, and *C* satisfy the equation $((A * B) - C) == 0xf14ec3$, an index into an array is offset by the value from *B*, so it is possible (though difficult) to generate an input that will lead to an out-of-bounds array access. The input file also determines the path taken by the program, and this path must visit the locations in the code where *A*, *B*, and *C* are assigned values from the input. However, the sample input provided to competitors did not execute this code for *A* and *C*, nor did it reach the potentially invalid array access. Competitors had to find an input that could reach these parts of the program.

But these challenges are not unique to this bug; other bugs required solving similar equations and executing new parts of the program. So why was this bug so much harder to find than others? And why could NU_AFL_QSYM find it and other teams could not? As

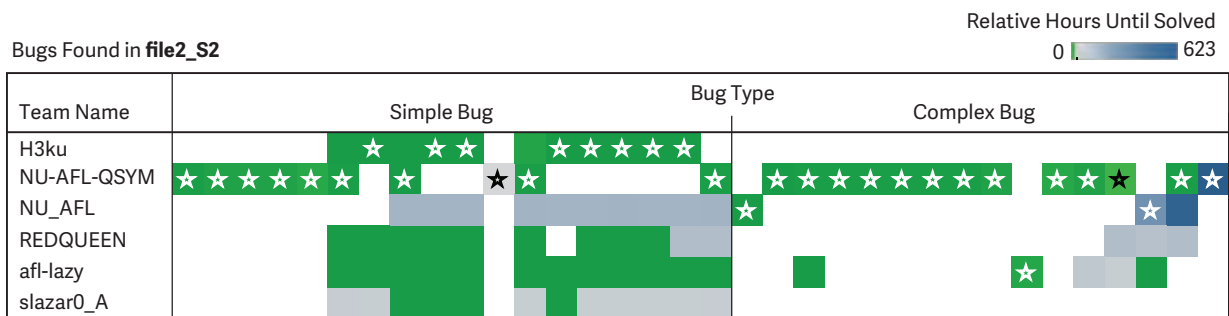


FIGURE 2. A visualization of when teams found bugs in *file2_S2* during the November 2018 Rode0day. Green indicates the bugs found within 24 h of a team's first score. The stars denote the first team to find a bug.

we continue to run our Rode0day competitions, we hope to collect enough data to answer questions like these.

By finding a way to quantify how difficult a bug is to find with a given approach to bug finding, we hope to gain an understanding of how different bug-finding strategies can be used in combination or how new bug-finding systems can be developed to find and fix more bugs in real programs.

The first year of Rode0day has been a success, but we see several areas where it can be improved. Most pressing, we have yet to ensure that the bugs used in the competition are representative of the naturally occurring bugs we'd like our bug finders to uncover. We plan to do this by 1) looking at many real-world bugs and trying to get LAVA to match their properties in ways that matter and 2) validating that the systems that perform well in our competition are also successful in the real world. We would also like to open up Rode0day to other bug-injection systems aside from LAVA and Apocalypse, such as hand-created bug corpora. This would allow others to contribute bugs that they think are more realistic or that cover new bug classes (e.g., use after free) that we haven't had the time to implement.

Another limitation is that most static analysis tools cannot currently participate because they do not provide a triggering input that demonstrates they have found a given bug. For each of the injected bugs, however, we know exactly where their "root cause" is, and because our bugs are injected into source code, we can, in principle, evaluate static analyzers as well. This task is slightly more complicated than evaluating fuzzers, both because of the possibility of false

positives and because different tools may reasonably disagree about the location of a bug in the program. For instance, if a pointer is corrupted on one line of a program but does not cause a crash until it is dereferenced on another line, which line contains the bug? We think these challenges can be overcome with careful scoring design and some empirical examination of how current static analyzers report bugs.

Finally, for the time being, we have intentionally made no attempt to place computational restrictions on competitors. This means that a well-funded team could potentially perform better by dedicating thousands of cores to bug finding. It also makes it more difficult to compare the performance of different tools. To make more direct, apples-to-apples comparisons between different tools, we plan to create a yearly, computation-limited competition. For this event, competitors would submit a Docker container or virtual machine containing their bug-finding system; we would then allocate some fixed amount of resources on a standard cloud platform like Amazon EC2 to run the bug-finding systems and declare a winner.

As we continue running Rode0day, we hope to keep improving our competitions to provide the best possible evaluation of bug-finding systems. Along the way, we'll keep collecting, publishing, and analyzing our data in the hope of giving our community a better understanding of bugs and bug finding. 🍌

ACKNOWLEDGMENTS

This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under U.S. Air Force contract FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations

expressed in this material are those of the author(s) and do not necessarily reflect the views of the Under Secretary of Defense for Research and Engineering.

REFERENCES

1. P. Chen and H. Chen "Angora: Efficient fuzzing by principled search," in *Proc. IEEE Symp. Security and Privacy*, 2018, pp. 711–725.
2. B. Dolan-Gavitt et al., "LAVA: Large-scale automated vulnerability addition," in *Proc. IEEE Symp. Security and Privacy*, 2016, pp. 110–121.
3. S. Roy, A. Pandey, B. Dolan-Gavitt, and Y. Hu, "Bug synthesis: Challenging bug-finding tools with deep faults," in *Proc. ACM Joint Meeting European Software Engineering Conf. Symp. Foundations of Software Engineering*, 2018, pp. 224–234.

ANDREW FASANO is a member of the research staff at the MIT Lincoln Laboratory, Lexington, Massachusetts, and a

Ph.D. student at Northeastern University, Boston, Massachusetts. Fasano received a bachelor's degree from Rensselaer Polytechnic Institute, Troy, New York. Contact him at fasano@mit.edu.

TIM LEEK is a member of the research staff at the MIT Lincoln Laboratory, Lexington, Massachusetts. Leek received a master's degree from the University of California San Diego. He is a member of USENIX and the Association for Computing Machinery. Contact him at tlee@ll.mit.edu.

BRENDAN DOLAN-GAVITT is an assistant professor at New York University. Dolan-Gavitt received a Ph.D. from the Georgia Institute of Technology, Atlanta. He is a Member of the IEEE. Contact him at brendandg@nyu.edu.

JOSH BUNDT is a Ph.D. student at Northeastern University, Boston, Massachusetts. Bundt received a master's degree from the Naval Postgraduate School, Monterey, California. He is a member of USENIX. Contact him at jmb@ccs.neu.edu.

ADVERTISER INFORMATION

Advertising Coordinator

Debbie Sims
Email: dsims@computer.org
Phone: +1 714-816-2138 | Fax: +1 714-821-4010

Advertising Sales Contacts

Mid-Atlantic US:
Dawn Scoda
Email: dscoda@computer.org
Phone: +1 732-772-0160
Cell: +1 732-685-6068 | Fax: +1 732-772-0164

Southwest US, California:
Mike Hughes
Email: mikehughes@computer.org
Cell: +1 805-208-5882

Northeast, Europe, the Middle East and Africa:
David Schissler
Email: d.schissler@computer.org
Phone: +1 508-394-4026

Central US, Northwest US, Southeast US, Asia/Pacific:
Eric Kincaid
Email: e.kincaid@computer.org
Phone: +1 214-553-8513 | Fax: +1 888-886-8599
Cell: +1 214-673-3742

Midwest US:
Dave Jones
Email: djones@computer.org
Phone: +1 708-442-5633 Fax: +1 888-886-8599
Cell: +1 708-624-9901

Jobs Board (West Coast and Asia), Classified Line Ads

Heather Bounadies
Email: hbonadies@computer.org
Phone: +1 623-233-6575

Jobs Board (East Coast and Europe), SE Radio Podcast

Marie Thompson
Email: marie.thompson@computer.org
Phone: +1 714-813-5094

DEPARTMENT: THE PRAGMATIC DESIGNER

Better Code Reviews With Design by Contract

George Fairbanks

Design by contract (DBC) is a technique that improves the quality of your team's code. It yields code with both a logical and a procedural nature, where the contracts state declaratively what will happen, and the implementations procedurally cause the desired effect. The team can reason either logically, by using the contracts, or procedurally, by following the code line by line, but the former allows them to reason about far larger programs. It also creates conditions for deliberate practice so developers using DBC grow their design skills faster.

Teams that are looking for ways to improve their code should seriously consider DBC. It is a technique for designing software in which each method has a contract, much like a legal document, stating what the caller is responsible for and what the method body must do. It was introduced as a term and adapted to object-oriented design in the 1980s by Bertrand Meyer¹ and traces its roots to the late 1960s with the work of Robert Floyd, Tony Hoare, and Edsger Dijkstra on reasoning logically about procedural programs.

DBC is not a magic elixir that guarantees great programs. It's more like standard sentence mechanics in an essay. When sentences are awkward, it's hard for a paragraph or essay to succeed. There are authors who have written great essays while breaking grammar and style rules, but they have done so after they've mastered them. DBC ensures that methods and functions are simple and easy to understand and are therefore great building blocks for a whole program.

Many teams do code reviews so that all code changes are reviewed by a teammate. Before being sent for review, however, the author has already tested the code and knows it works. So, code reviews check

not whether it works but if it's well designed. Clean code usually has a contractual nature while spaghetti code does not, even if its author is not consciously following DBC. Who has not sighed in frustration when encountering something like `"void process() { /* 1kLOC elided */ }`?

DBC and code reviews are a great combination. Reviewers may themselves write clean code but struggle to guide others to do the same. They can say how they would write the code but not articulate why that is better. Such advice can devolve into a battle of opinions. The opportunity with DBC is that when both the author and reviewer agree to a goal of writing code with clear contracts, they can look out for the DBC practices being followed (or not) in the code being reviewed. A reviewer can point the author to the relevant DBC practice and, fingers crossed, help the author improve the design. This article includes a list of DBC practices that you can add to your code review style guide.

HUNGRY FOR CONTRACTS

What are contracts exactly? Let's introduce this concept using an example that's familiar to everyone—buying a sandwich—and apply some practices to decide on a contract. A sandwich seller might tell us that buying a sandwich can be broken down into a series of steps: take payment, give change, collect the ingredients (bread, peanut butter, and jelly), spread the peanut butter on one piece of bread using a knife, spread jelly on the other piece using a knife, assemble the sandwich, and deliver the sandwich. Let's call those steps the implementation of the `buySandwich` method.

DBC asks us to add a contract to that implementation, for example (given in U.S. dollar): If I give you \$5, you will give me a sandwich. That contract states the buyer's responsibility and what the seller will provide in return. We can make that a bit clearer by saying "buyer" and "seller" instead of "me" and "you." We can also

state what you probably assumed, i.e., that the seller takes the money. Here's an improved version of that contract: If the buyer gives the seller \$5, the seller will give the buyer a sandwich and keep the \$5.

If you had read that contract as a code comment above the method `buySandwich`, you'd be able to understand that method fairly well without reading the code body. Notice that the contract isn't a trans-

lation of the procedure into natural language. The contract doesn't talk about how the sandwich is made (e.g., using a knife) or the sequence of its operations. Instead, it states what happens before and after the method.

You may have noticed that the contract talks about what happens using procedural language: The seller gives the buyer \$5. We can change that to declarative language like this: The buyer has \$5. Now that it's stated declaratively, we can use it as the precondition for the `buySandwich` method. The postcondition becomes: The seller has the buyer's \$5, and the buyer has a sandwich. The contract and procedure are shown in Figure 1.

Perhaps you are thinking that this is just a mild improvement, as there could have been a comment on the `buySandwich` method saying something similar. Besides, everyone knows how buying a sandwich works. What is different is that callers know what they can depend on. Consider a few details of the implementation that do not appear in the contract:

- › Making change: The method body says that the buyer will get change, but the contract doesn't guarantee that. You have probably seen similar best efforts, say, from a vending machine that might not have exact change or a public bus.
- › Assembly with a knife
- › Peanut butter before the jelly
- › Sandwich type: I'm sure there are readers who were already questioning whether peanut butter and jelly is an acceptable sandwich at any price.

Without the contract, you could read the method name and implementation and then guess at the contract, but it's easy to infer the wrong things. You

buySandwich

Precondition: The buyer has \$5.

- Take payment.
- Give change.
- Collect the ingredients (bread, peanut butter, and jelly).
- Spread the peanut butter on one piece of bread using a knife.
- Spread jelly on the other piece using a knife.
- Assemble the sandwich.
- Deliver the sandwich.

Postcondition: The seller has \$5, and the buyer has a sandwich.

FIGURE 1. An example of contract and implementation.

could convince yourself that any of these details are something a caller may depend on. If so, how will we ever fix bugs or change the implementation to run faster? Stating the contract removes the guesswork. As an implementer, writing a contract leads you to think about what the caller can rely on and separate that from how the method is implemented. As a caller, a contract tells you what's safe to depend on.

CLEAN CODE USUALLY HAS A CONTRACTUAL NATURE WHILE SPAGHETTI CODE DOES NOT, EVEN IF ITS AUTHOR IS NOT CONSCIOUSLY FOLLOWING DBC.

LOGICAL REASONING

Without the contract, you can reason through a method procedurally, animating the source code line by line in your head like a little machine, and draw conclusions about how it will behave. When methods have contracts, you can still use procedural reasoning if you want to, but you can also apply logical reasoning.

In this `buySandwich` example, you know that, before calling the method, you are rich and hungry, and afterward, you are poor and full. That's consistent with reasoning procedurally about the implementation, but it's different. It lets you employ formal logic, which is why contracts are used in automated program analysis, such as when your IDE warns you that a value in your program might be null.

Contracts also let humans reason informally with logic, and we do that all the time. Imagine an implementation of `buySandwich` that uses two helper methods:

- Contracts state what must be true about the inputs.
- Contracts state what will be true about the outputs.
- Contracts use declarative, not procedural, language.
- Contracts omit implementation choices, including sequence.
- Contracts state when a subset of a type is used.
- Contracts state how nulls are handled.
- Contracts state what inputs or states trigger predictable failures.
- Contracts identify any side effects.
- Callers can understand the contract without reading the implementation.
- State ubiquitous conditions as invariants.
- Prefer simple contracts over complex ones.
- Omit contracts only when caller cannot infer the wrong contract.
- Align contracts with the contours of the problem.
- Separate predicates, queries, and commands.

FIGURE 2. The design-by-contract practices.

collectMoney and makeSandwich. Does that work? Your logical intuition says yes. But consider different helper methods blandly named *A* and *B*. Now your intuition is less sure. You actually don't have any more information about collectMoney than you do about *A*, but your mind inferred a logical contract such as "At

ONCE DEVELOPERS START THINKING ABOUT DBC, IT CHANGES HOW THEY WRITE EVERY METHOD.

the end of the method collectMoney, the seller has the \$5." You couldn't have been reasoning procedurally through the implementation because there isn't one.

We all reason through programs procedurally, but there's a size limit, and it's not big. Can you keep 10,000 lines in your head and reason about it? Using procedural reasoning, that's shaky, but using logical reasoning it's pretty easy. Consider this: I bet you recall the postcondition for buySandwich, but do you recall every step in the implementation?

To me, the ability to scale our reasoning is the great benefit of DBC. When you structure your program with clear contracts on methods, you can always fall back to procedural reasoning, but you also unlock your ability to reason logically and can keep larger programs in your head.

HOW TO GET STARTED

Every code review starts with the author thinking that the proposed change is a good idea, so we should be

looking for ways to guide authors in advance, not just during the review. Authors can shoot at a known target as they write code by using a list of DBC practices and the overall guiding metaphor of a contract that is usable by callers.

I was unable to find a checklist suitable for use in code reviews, so I created the list shown in Figure 2, and I think it's consistent with DBC literature, such as in Mitchell and McKim.² You can think of the list as an extension of the team's

coding style. Like any style guidance, you should discuss and tweak it for your project.

I find that once developers start thinking about DBC, it changes how they write every method. There are always choices about how to decompose a problem, and they will gravitate toward methods where the contract is easy to state. When introducing DBC to an existing team or codebase, it's better to do it gradually and skip contracts that you think callers can reliably guess. Overall, state contracts as terse comments that help callers. DBC is something that you can practice on your own but it's even better if the whole team adopts it.

Low-hanging fruit is the easiest to pick and coworkers are unlikely to protest, so I suggest starting with methods that represent predicates, such as `isActive` or `hasAddress`. Contracts for these methods can be stated as one-line comments of the form "Returns true iff ...," where "iff" is short for "if and only if." If the codebase already has methods like these, the change is just stating the contract, but if the methods don't exist, then you also benefit from making the code read better by reducing in-line logic.

Next, turn your attention to query methods, like `getStatus` or `getAddress`. Because a comment saying "Returns the status" is unhelpful noise, think about the corner cases and write a contract if you discover anything interesting, i.e., how it handles null, whether the method checks for invalid data, or whether the values are only a subset of the declared type (especially for primitives such as string or integer). Also consider whether the accessors should exist, as they could be coupling the caller to the current implementation unnecessarily.

When you write contracts for commands (i.e., transactional methods that are impure), look for opportunities to split the method. Broad methods such as `updateCustomer` tend to have long contracts that cover all of the corner cases. It can be easier to write the contract for a narrower method that does less, and it's easier for clients to understand. A set of commands likely has an obligation to maintain invariants on the data structures it manipulates, so make sure those invariants are clearly stated.

DELIBERATE PRACTICE

As they grow, developers must learn to detect vagueness, incompleteness, and clumsiness in their designs. Years on the job will eventually give them those skills, but mundane experience is less effective than deliberate practice. One nice thing about DBC is that it can turn routine programming into deliberate practice. To understand how, let's look at deliberate practice in another field.

William Zinsser, an English composition teacher, says that in his writing classes, he would not cross out unclear or unnecessary parts of students' sentences but would instead put square brackets around those parts. Rather than simply telling the students what he considered the right answer, his technique encouraged them to wrestle with it themselves. His experience was that, by the end of the semester, they had learned to write terse prose.³

By encouraging his students to wrestle with their work, he created the conditions for deliberate practice. I see the same thing happening with DBC. When I'm just grinding out code, I'm not deliberately practicing. But when I force myself to state the contracts for each method, I notice when an idea is fuzzy, when the contract is rambling, or when my code makes unstated assumptions about a data structure.

The act of stating contracts creates the conditions for deliberate practice. It makes the unstated visible, like the square brackets that direct attention in an essay. The contracts let me see my code from a different perspective, revealing design flaws, and nudging me toward clearer designs. If DBC is able to accelerate the careers of developers by helping them learn to detect vagueness, incompleteness, and clumsiness in their designs, it is worth trying for that reason alone.

DBC is a technique taught to computer science undergraduates at many universities, including the Massachusetts Institute of Technology in 6.031 and Carnegie Mellon University in 17-241. It encourages designs where you know what must be true when a method completes, rather than designs where a method does a bunch of stuff and you squint to infer what exactly it means.

If you are a technical lead or manager who wants to improve your system's code, you could just wait several years until the team has more experience. If you want to do something today, however, there are only a few techniques that are easy to teach and offer the benefits that DBC does. DBC helps if just one person on the team applies it, and it helps more with each additional person.

There are other ways to arrive at elegant designs, but DBC is a particularly good fit for code reviews because a reviewer can point the author to a practice that the code does not yet follow. What's more, DBC leads the team to think about the abstractions that the contracts refer to, so it's a gateway to other helpful techniques like precise modeling.

Having contracts on methods is like having an owner's manual in your car's glove box. The most loved owner's manuals are the ones that are never opened because the design is simple and obvious. Everyone wants software that is simple and probable, but wishing does not make it so. DBC provides an early warning about awkward designs, shows where complexity still lives, and often leads to methods that callers understand without reading the contract. 🧐

REFERENCES

1. B. Meyer, *Object-Oriented Software Construction*. Reading, MA: Addison-Wesley, 1994.
2. R. Mitchell and J. McKim, *Design by Contract, by Example*. Reading, MA: Addison-Wesley, 2001.
3. W. Zinsser, *On Writing Well: The Classic Guide to Writing Nonfiction*. New York: Harper Perennial, 2016.



GEORGE FAIRBANKS is a software engineer at Google. Contact him at gfa@georgefairbanks.com.

Code Mining

Gerard J. Holzmann

This article originally
appeared in
Software
vol. 36, no. 2, 2019

Machine-learning techniques are starting to achieve some impressive feats. The successes are due, in part, to the availability of astonishingly large amounts of data that can be used for training. Show a suitably equipped system a billion images of cats and a billion images of lawn mowers, and then see whether it can figure out what the common patterns are. If this succeeds, we can then show the same system a picture of our own cat or lawn mower, and maybe it can tell which is which. Of course, if you show the same system an image of a house, it wouldn't have a clue beyond saying that it's neither a cat nor a lawn mower. Basically, what we're leveraging are statistics, not intelligence, but, nonetheless, the results can be impressive.

Could we somehow use the same type of approach in software analysis to find likely bugs? Just superficially, you can imagine an experiment in which we show some "suitably equipped" system a billion examples of good programs and ask it to deduce common patterns. Then we show it a billion examples of programs known to be buggy and again ask it to spot some common characteristics among these. Next, we show the now-learned system our own code and ask it to classify the code as either good or bad. Finding the input sets would not be very difficult. There are large enough databases of reasonably "good" programs available. I'll conveniently sidestep the landmine here of trying to give examples, but we can likely all think of a few that could fit the bill. We also have substantial repositories of examples of bad code, such as the fragments of code collected in the ever-growing database of common weakness enumerations that you can mine at <https://cwe.mitre.org>. I think you'll agree that this approach is unlikely to work. I'll try to explain why I think so, although I would be delighted if someone could prove me wrong.

Digital Object Identifier 10.1109/MS.2018.2884867

Date of publication: 22 February 2019



FIGURE 1. A picture of a smiling cat that, with some luck, a machine-learning system could be trained to recognize. My daughter, Tessa, drew this when she was 10 years old, deviously adding orange splotches that she must have known could complicate recognition.

MINING CATS

How do we know, or, how could our learned system know, that we're looking at a picture of a cat like in Figure 1? Clearly there are clues in the image. The outline of the cat is one such hint. Other clues are the two eyes, the pointy ears, and other characteristics common to cats. Both we and our systems can learn to recognize those patterns. The patterns in the image can, of course, be deduced with relatively simple image processing techniques until we get what is basically a fingerprint or signature of each image. I'm not a machine-learning expert, but I can imagine that some clever hashing techniques could be used to group similar images, which can then be used to gain a confidence rating of the closeness of a match. Figure 1 should then score a lot better against our fingerprints of cat images than against those of lawn mower images.

But how would that process work if we wanted to distinguish good from bad software? Program code is

rife with patterns that are obvious to a human reader. But are they also easy to discover by a learning algorithm? A few examples can illustrate the degree of difficulty.

A person may realize, for instance, that we don't see many appearances of the word "goto" in well-written code, but he or she may also notice that, whenever we do see it, it is most often followed by a name, and that same name appears somewhere else in the same code followed by a colon. That name, finally, will rarely be followed by an equal sign in the same fragment of code. Of course, the name in question is quite arbitrary, which means that, although the pattern is real enough, it might be very hard for an algorithm to discover it. Furthermore, we should add that even if our algorithm would discover it and could detect deviations from the pattern in bad code, that ability is not too useful because every halfway decent compiler can do the same.

CODE AS TEXT

It would be a little more useful if our algorithm could deduce that a name, when it is followed by an open round brace and preceded by a word other than "void" (in "good" programs, e.g., in the definition of a function), will almost never be preceded by either a colon or a semicolon when it appears elsewhere in the code (again, followed by an open round brace). That is, the value returned by the function is effectively used by the caller in an expression or an assignment.

As you can see, trying to describe these code patterns as just textual coincidences leads to some tortured language, but the statistics should definitely bear out these types of correlations. Whether they are also discoverable by a machine-learning algorithm without further guidance is another matter. I have my doubts. Unfortunately, it is generally easy to record patterns that are present but much harder to record patterns that are rare or absent. An example of a well-known pattern that is absent from well-written English prose, for instance, is that a preposition is rarely followed by a period. Our teachers made sure that we all follow that rule, at least most of the time.

CODE FORENSICS

Although the patterns I mentioned so far are somewhat dubious, it is not too hard to think of correlations that could be worth mining. For example, consider a

"for" statement in the C language. It doesn't take much imagination to note that if the condition part of the for-statement contains a < or <= operator, then the increment part of the statement will most likely contain an increment operator, such as ++ or +=. Similarly, if the condition part contains a > or >= operator, then the increment part will most likely contain a < or <= operator. Intuitively, this is fairly obvious. We should be on our guard now, because sometimes things that seem obvious are actually not true. So let's make sure.

I don't have a machine-learning algorithm handy that can give me statistics for large code bases, but it is easy enough to run some tests, say, on a couple of million lines of source code from one of the Linux distributions. I'll use the 14.9 million lines of code from the Linux 4.3 distribution for this check.

Figure 2 shows the correlation of operators used in the condition part of for-loops with operators used in the increment part of the same loop. So, here, we're ignoring everything else that might appear within the control part and just look at the operators. It's fair to say that there is, indeed, a correlation of the type we suspected, and it's a pretty strong one at that. The most common way to write a for-loop is to use a < operator in the condition and a ++ operator for the loop increment. Table 1 shows the most interesting combinations in more detail.

The more unexpected combinations, for instance, the use of a < or <= operator and a decrement operator, are quite rare but curiously nonzero. Those cases, being the statistical outliers, could well warrant more scrutiny and may contain either mistakes or the use of obscure code that may be prone to misunderstanding. A good example of the former could be this line of code in file `./drivers/net/ethernet/qlogic/netxen/netxen_nic_hw.c`, line 2,335:

```
for (k = 0; k < read_cnt; k--) {
```

and an example of the latter could be this line of code in file `mm/memtest.c`, line 108:

```
for (i = memtest_pattern - 1; i < UINT_MAX; --i) {
```

The good thing is that, once we know the statistics, the number of outliers is usually sufficiently small that we can inspect each one in detail and form an opinion

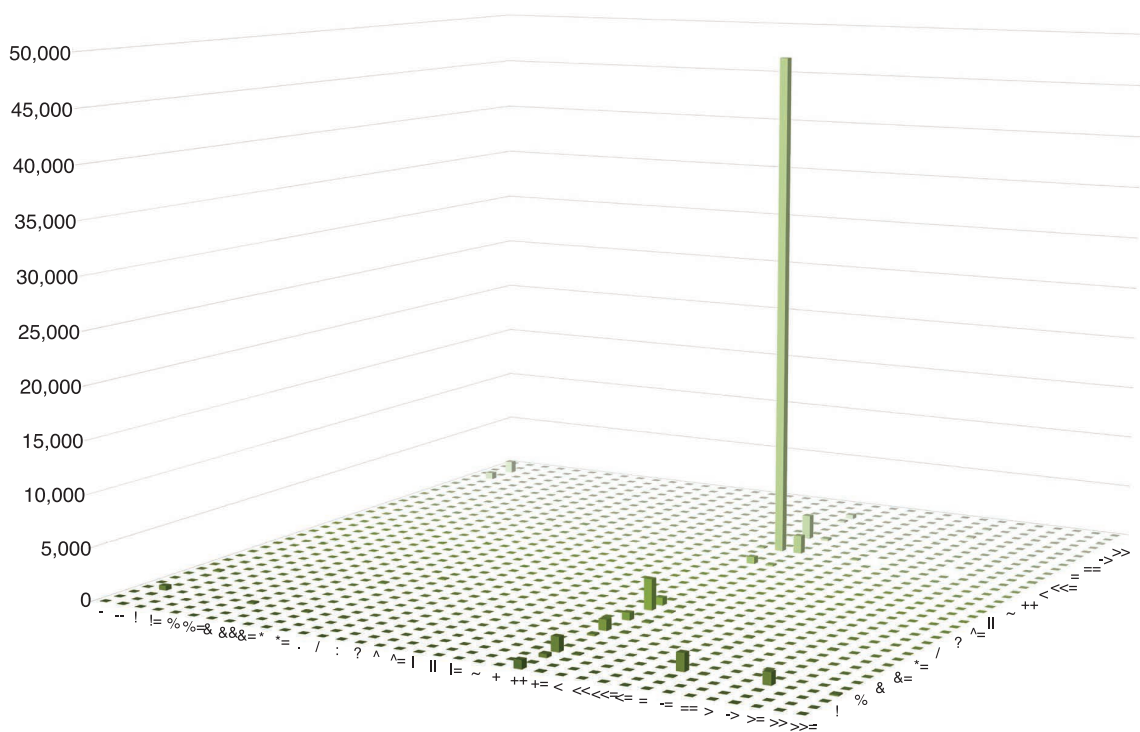


FIGURE 2. A graphic showing the correlation between operators used in the condition part of a for-loop and the operators used in the increment part of the same for-loop. The peak of 48,138 cases is for the combination of operator < in the condition part and operator ++ in the increment part of the same for-loop. The statistics are for 14.9 million lines of the Linux 4.3 distribution. Symbols at the base of the graphic represent operators. Numbers along the left side represent number of cases.

TABLE 1. Some of the most frequent operator combinations used in for-loops.

Condition\increment	++	+=	--	-=
<	48,138	1,726	26	7
<=	2,301	209	5	0
>	154	48	666	123
>=	35	20	1,157	98

about the quality of that code. A static analyzer could take advantage of these types of patterns, but the question is still: How can we systematically discover them with machine-learning techniques or otherwise?

CODE EXPLORATIONS

I haven't said yet how I generated the data shown in Figure 2 and Table 1. You can, of course, try to use simple text processing tools to extract the code fragments of interest. For example, we can try to use the tool `grep` to find all occurrences of the keyword "for"

and then filter out the operators in the control part that follows in round braces. Often, the complete control part of the for-loop appears on the same line as the initial keyword but not always. There are enough exceptions to the one-line rule here that we may want to try another way.

A code-browsing tool comes in handy at this point, especially if it allows us to specify a pattern of interest as a regular expression over lexical tokens rather than plain characters. The tool I've used for my check is called *Cobra* (<http://spinroot.com/cobra>), and it supports such a pattern-matching language. So when we say, for instance,

```
$ cobra -pat "for ( .* ) " *.c
```

we're not restricted to matches that appear on a single line, since the match is on the token sequence, which is insensitive to white space.

Unfortunately, this still doesn't give us the statistics we need. We will have to postprocess the code to

extract the two groups of operators. Yet even though the input to the pattern matcher above isn't sensitive to white space, the output (the matching lines) will still contain all those line breaks. So, in postprocessing, we must still deal with that circumstance.

I found it easier to enact the entire process with a sequence of code-querying commands, issued interactively in a code-querying session over all of the 14.9 million lines of code. I used those commands to home in on the various parts of each for-loop and then write it out in two groups of operators per line, one for the operators used in the condition part and the other for operators used in the increment part. A few small postprocessing steps then gave me the data for Figure 2. But we can do more.

For example, the following is a sequence of Cobra commands I used to find a relatively small set of deviations from the general rule that we can find in the code base. The pound sign (#) is a comment delimiter of the query-command language:

```
mark for (      # mark all keywords "for" followed
                by a round brace
next           # move the mark forward to the
                token following "for"
contains/`<=?$  # keep only those marks where the
                range of tokens that is
                # enclosed in the round braces
                contains an < or <= operator
contains/`-[-]=$ # keep only those marks where the
                range of tokens also
                # contains at least one decrement
                operator
contains no ++  # and no increment operators
= "nr matches:" # report the number of matches
display        # and display them
```

The first command in this set marks all places in the code where we find the keyword "for" followed by an open round brace. Remember that white space doesn't matter since we're matching on lexical tokens here. The matches will point at the locations in the code where the "for" itself appears. With the next command, we now move those match points forward to the open round brace. That open round brace is part of a pair that the tool knows about, and the range of tokens in between the matching open and closing

round brace can be interrogated as a set. So the third command checks that set and narrows the selections to only those sets that contain tokens matching the regular expression shown. The regular expression matches only on token texts that start with < and may contain a subsequent = character and nothing else. Now we have all for-loops that contain either a < or a <= operator. Next, we check whether those same for-loops contain a decrement operator, which is either a - or a -= token, again, with a regular expression. We refine the set of matches a little more by next also saying that the same clause does not also contain the increment operator ++, and we then display the small set of remaining matches.

The whole sequence can be abbreviated by using single-letter shorthand for each query command, and we can pass the entire command sequence in a command-line invocation of the tool rather than typing them in one at a time in an interactive session, but you get the idea. Of course, this whole process of querying the code base would be of little use if it took more than a few seconds, so speed is important.

FAST APPROXIMATIONS

We can get the tool to work fast because the matching algorithm is simple and, therefore, trivial to parallelize. It can, in principle, be executed independently on each separate file, so given enough CPUs, the 21,987.c files in the Linux distribution we looked at could be scanned within a fraction of a second. On my own 32-core system, the processing takes roughly 20 s, which includes approximately 9 s for preparing the lexical token stream itself. I should add that this last code-mining method I used is not precise. For example, I did not bother to distinguish among the three different parts of the for-loop. The final set of matches was small enough, even for a code base of this size, that I could afford to overshoot the final set by a small amount and then peruse the matches for the little gems I was really seeking.

This brings to mind a quote from the late John Tukey, a Bell Labs mathematician who, in the late 1940s and 1950s, worked with both John von Neumann and Claude Shannon. Tukey wrote in 1962:

The most important maxim for data analysis is to heed, and one which many statisticians seem to

have shunned, is this: “Far better an approximate answer to the right question, which is often vague, than an exact answer to the wrong question, which can always be made precise.”¹

The observation was meant as a critique of the cookie-cutter approach that some of Tukey’s colleagues were using to perform data analysis at the time: spending little or no time carefully checking whether the underlying assumptions for those methods were valid in each case at hand. Perhaps the same can be said about our use of static source-code analysis techniques today. Although most tools allow the user to customize the properties checked for to a given code base, these capabilities are rarely used.

Several obstacles stand in the way of the routine use of custom properties in the most commonly used static source-code analyzers. One is the difficulty of specifying new types of queries. A person has to be very determined to learn the internal details of the tool, which are, of course, different for each one. Then the average query resolution time for new types of static analysis can be substantial if the code base grows to a few million lines. The long waits discourage experimentation and targeted approximation to home in on cases of interest.

CASTING DOUBT

Query approximations that can be quickly executed can have real value. Consider how we may try to determine how often different types of pointer arithmetic are used in a large code base. If you’re brave enough to try to express this in one of the customization languages of the leading static analysis tools, you are a better person than I. Here’s how I would do it with a Cobra query that can both be written and executed in a few seconds:

```
$ cobra -pat “\* ) ( .* + .* ) ” *.c
```

The pattern I used attempts to match type casts, enclosed in round braces, that end with a star operator, indicating that whatever follows is cast to a pointer. Next, the pattern checks that the expression in round braces that follows contains a plus operator, indicating that an addition is being performed. The syntax for the pattern expression we used here is a simplified version

of a regular expression. The initial star is escaped so that it cannot be misinterpreted as a metasymbol. The dot matches any token text, and the combination.* is used to indicate an arbitrary sequence of tokens. Cobra makes sure that the round braces in the pattern match, which forces the text in between the braces to refer to a single expression. In an interactive session with the Linux source code, matches to this query pattern are found in roughly 1 s on a multicore system. If you’re curious, there are approximately 6,000 such matches in the 14.9 million lines of Linux code. Here’s the first one, found in file mm/slub.c, online 247:

```
return *(void **)(object + s->offset);
```

It is, of course, not much harder to now home in on more egregious types of pointer arithmetic that use a sequence of additions, subtractions, and, perhaps, even multiplications. Once we have the initial set of matches, those refinements are simple to add interactively by querying the range of tokens contained in the braced expression, similar to what we did before when querying for-loops. If we do so, we are rewarded with roughly 70 matches. The following cast, which contains an addition, a subtraction, and a multiplication, is found in file drivers/video/console/fbcon.c on lines 2697–2698 (note that the pattern spans multiple lines):

```
return (u16 *) (vc->vc_origin + offset -
               softback_lines * vc->vc_size_row);
```

Both the ease with which we can phrase these queries and the speed with which they can be resolved make a difference. So far though, it’s most helpful to those old-fashioned types of learning systems: us. 🙄

REFERENCE

1. J. W. Tukey, “The future of data analysis,” *Ann. Math. Statist.*, vol. 33, no. 1, pp. 1–67, Mar. 1962.



GERARD J. HOLZMANN works on developing stronger methods for the design and analysis of safety-critical software as a consultant and researcher at Nimble Research. Contact him at gholzmANN@acm.org.



PURPOSE: The IEEE Computer Society is the world's largest association of computing professionals and is the leading provider of technical information in the field.

MEMBERSHIP: Members receive the monthly magazine *Computer*, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.

COMPUTER SOCIETY WEBSITE: www.computer.org

OMBUDSMAN: Direct unresolved complaints to ombudsman@computer.org.

CHAPTERS: Regular and student chapters worldwide provide the opportunity to interact with colleagues, hear technical experts, and serve the local professional community.

AVAILABLE INFORMATION: To check membership status, report an address change, or obtain more information on any of the following, email Customer Service at help@computer.org or call +1 714 821 8380 (international) or our toll-free number, +1 800 272 6657 (US):

- Membership applications
- Publications catalog
- Draft standards and order forms
- Technical committee list
- Technical committee application
- Chapter start-up procedures
- Student scholarship information
- Volunteer leaders/staff directory
- IEEE senior member grade application (requires 10 years practice and significant performance in five of those 10)

PUBLICATIONS AND ACTIVITIES

Computer: The flagship publication of the IEEE Computer Society, *Computer* publishes peer-reviewed technical content that covers all aspects of computer science, computer engineering, technology, and applications.

Periodicals: The society publishes 12 magazines and 18 journals. Refer to membership application or request information as noted above.

Conference Proceedings & Books: Conference Publishing Services publishes more than 275 titles every year.

Standards Working Groups: More than 150 groups produce IEEE standards used throughout the world.

Technical Committees: TCs provide professional interaction in more than 30 technical areas and directly influence computer engineering conferences and publications.

Conferences/Education: The society holds about 200 conferences each year and sponsors many educational activities, including computing science accreditation.

Certifications: The society offers three software developer credentials. For more information, visit www.computer.org/certification.

BOARD OF GOVERNORS MEETING

24 – 25 September 2020 in McLean, Virginia, USA

EXECUTIVE COMMITTEE

President: Leila De Floriani

President-Elect: Forrest Shull

Past President: Cecilia Metra

First VP: Riccardo Mariani; **Second VP:** Sy-Yen Kuo

Secretary: Dimitrios Serpanos; **Treasurer:** David Lomet

VP, Membership & Geographic Activities: Yervant Zorian

VP, Professional & Educational Activities: Sy-Yen Kuo

VP, Publications: Fabrizio Lombardi

VP, Standards Activities: Riccardo Mariani

VP, Technical & Conference Activities: William D. Gropp

2019–2020 IEEE Division VIII Director: Elizabeth L. Burd

2020–2021 IEEE Division V Director: Thomas M. Conte

2020 IEEE Division VIII Director-Elect: Christina M. Schober

BOARD OF GOVERNORS

Term Expiring 2020: Andy T. Chen, John D. Johnson, Sy-Yen Kuo, David Lomet, Dimitrios Serpanos, Hayato Yamana

Term Expiring 2021: M. Brian Blake, Fred Douglass, Carlos E. Jimenez-Gomez, Ramalatha Marimuthu, Erik Jan Marinissen, Kunio Uchiyama

Term Expiring 2022: Nils Aschenbruck, Ernesto Cuadros-Vargas, David S. Ebert, William Gropp, Grace Lewis, Stefano Zanero

EXECUTIVE STAFF

Executive Director: Melissa A. Russell

Director, Governance & Associate Executive Director: Anne Marie Kelly

Director, Finance & Accounting: Sunny Hwang

Director, Information Technology & Services: Sumit Kacker

Director, Marketing & Sales: Michelle Tubb

Director, Membership Development: Eric Berkowitz

COMPUTER SOCIETY OFFICES

Washington, D.C.: 2001 L St., Ste. 700, Washington, D.C. 20036-4928; **Phone:** +1 202 371 0101; **Fax:** +1 202 728 9614;

Email: help@computer.org

Los Alamitos: 10662 Los Vaqueros Cir., Los Alamitos, CA 90720;

Phone: +1 714 821 8380; **Email:** help@computer.org

MEMBERSHIP & PUBLICATION ORDERS

Phone: +1 800 678 4333; **Fax:** +1 714 821 4641;

Email: help@computer.org

IEEE BOARD OF DIRECTORS

President: Toshio Fukuda

President-Elect: Susan K. "Kathy" Land

Past President: José M.F. Moura

Secretary: Kathleen A. Kramer

Treasurer: Joseph V. Lillie

Director & President, IEEE-USA: Jim Conrad

Director & President, Standards Association: Robert S. Fish

Director & VP, Educational Activities: Stephen Phillips

Director & VP, Membership & Geographic Activities:

Kukjin Chun

Director & VP, Publication Services & Products: Tapan Sarkar

Director & VP, Technical Activities: Kazuhiro Kosuge

When Multimedia Meets Fashion

Sijie Song, *Peking University*

Tao Mei, *JD.com*

Through transforming visual fashion into computational images, multimedia technologies are transforming fashion industry at a faster pace than before. The huge demand for fashion analytics triggers promising applications and theoretical research in this area. The future of fashion is being reshaped by multimedia, and researchers are working toward computational fashion.

Even without realizing that artificial intelligence (AI) is no longer a prerogative in research laboratories of computer science, it is gradually getting well prepared for our everyday lives in a wide range of fields, including health-care, transportation, education, and even fashion. Multimedia, as an important part in AI, provides powerful tools for analyzing, understanding, and predicting fashion. With rich data posted in the fashion communities and online stores from the Internet, multimedia technologies are trying to teach machine to rethink fashion, cultivate fashion sense, and creativity. The collision of fashion and multimedia technologies would definitely bring a new revolution to the fashion industry.

DEMANDS AND STARTUPS IN FASHION

Fashion is an integral part of our everyday life. It is about not only what people wear, but also a mirror of people's attitude toward life, reflections of culture, arts, and even economics. Fashion is really a huge industry of which the total global value is up to 3T dollars, about 2% of the world's gross domestic product.¹ In general, there are usually three roles involved in the fashion industry, including merchandisers, designers, and customers. With the demands and business opportunities from the three roles, fashion gradually becomes a hot topic, not to mention the increasing attention from many startup companies. We will next demonstrate the needs from various aspects and some promising startups in the fashion industry.

In recent years, fashion industry has developed mature pipeline from the process of design to production. The rapid increase in fast fashion brands such as Zara, H&M, and the development in e-commerce giants, such as Amazon, JD.com, etc., have led the fashion industry to find tools to accurately locate market orientation and spot potential customers. For these merchandisers, they are dedicated to predicting the fashion trend to guide their production and advertisement spending. They need to know what consumers want to wear, before the consumers even know themselves. With the technologies in multimedia, the accurate prediction could be achieved through collecting and analyzing large amounts of digitized fashion-related data. The customer's browsing and shopping history are rich sources for retailers to align the product and demand. On the other hand, merchandisers pay more attention to provide personalized and diverse services to consumers by analyzing their shopping preference. JD.com, the leading e-commerce platform, foresees the fashion future powered by multimedia technology. It could provide a personalized webpage layout in the app according to users' preference and fashion style. An online retail assistant was also launched to offer professional style suggestions to customers, as an attempt to digitize the retail world.

Designers, the creator in the fashion chain, need to sense the emerging trends from numerous fashion shows. They are always supposed to remember insights from thousands of pictures or videos and,

TABLE 1. Landscape of companies in the fashion industry.*

Functions Industry	Item detection	Item attribute	Color analysis	Style classification	Visual search	Outfit recommendation	Fashion Trend
Markable	✓	✓	✓		✓		
Visenze	✓	✓	✓	✓	✓	✓	
ScopeMedia	✓	✓	✓	✓		✓	✓
SenseTime	✓	✓	✓		✓		
ProductAI	✓	✓	✓	✓			
Vue	✓	✓		✓		✓	
Alibaba	✓	✓		✓	✓	✓	
JD.com	✓	✓	✓	✓	✓		

*Markable: <https://markable.ai/>; Visenze: <https://www.visenze.com/>; ScopeMedia: <https://scopemedia.com/>; SenseTime: <https://www.sensetime.com/>; ProductAI: <https://www.productai.cn/home>; Vue: <https://vue.ai/>; Alibaba: <https://www.alibabacloud.com/zh/campaign/fashionai>; JD.com: <https://www.jd.com/>.

then, discover trending elements in their fashion collections. The collaboration of multimedia and fashion design is expected to explore how multimedia technologies can take inspirations from the fashion market as early as possible, and inject creativity into the designer's daily work. For example, the multimedia technology can replace the repetitive and low-level design works for the designers to improve their productivities. The team consisted of students from the Fashion Institute of Technology and researchers from IBM AI Research managed to use multimedia algorithms to combine shapes, colors, and patterns as a fashion basis for future design.²

Consumers care more about what to wear at a specific occasion and how to be in a fashionable outfit. The key to this problem lies in the harmonious clothing matching. It reminds the startups of the potential market in personal stylist. With fashion images from online social media such as Pinterest, or online

shopping websites, it is much easier for multimedia algorithms to learn the rules of outfit matching from the large amounts of online fashion data. Amazon and JD.com have provided intelligent speakers to evaluate different outfits captured by snapshots, and they are able to return a result indicating which outfit won out over the other. In addition, consistently improving the shopping experiences is another promising direction in fashion industry. Markable, ViSenze, etc., the startup companies for visual e-commerce, deliver intelligent visual recognition solutions, enabling shoppers to search and discover products they are already inspired to buy. Besides, JD.com has respectively launched an AR Styling Station and a three-dimensional (3-D) try-on fitting room to help consumers test makeup or clothes virtually with the convenience of online shopping.

Overall, startups are trying their best to occupy the fashion market. To give an overview of the fashion

industry, we list several top companies and the fashion functions they are working toward more practical applications in Table 1.

TOWARD COMPUTATIONAL FASHION

From catwalk to street style, fashion is basically conveyed and perceived by vision. Thus, it is an excellent domain for applying technologies from computer vision and multimedia, where how to convert fashion into computational data is the key issue. Essentially, fashion images are 2-D arrays of numbers known as pixels. The first step toward computational fashion is to interpret these basic pixel values, and then analyze them to achieve further tasks.

At the early stage, images are extracted by hand-crafted descriptors, which are designed by humans, such as image gradients, edges, or other image statistics calculated based on the pixel values. Later on, more complex feature descriptors such as local binary pattern, histogram of oriented gradient, and scale-invariant feature transform were developed to better capture the essential components in images, regardless of the variation in viewpoints, scale, and even brightness. With lots of features from images in a specific category, various tools are employed to discover the pattern for recognition. Typical tools in pattern recognition include K-nearest neighbors, support vector machine, and so on.

In the recent years, researchers started to train machines to learn the world as humans do. Human perceive images through the retina in the eye made of millions of individual cells. Our brain assembles all these points and processes the inherent signals and structures. For example, after looking lots of blouse images, our brain is able to conclude the pattern of blouses, which usually consist of a collar, two sleeves, and body. Thus, we are able to recognize the given image as a blouse or not. Researchers design deep neural networks to simulate densely interconnected cells, imitating how our brain recognizes patterns and makes decisions. The models are learning all by itself, just like a brain. That is the wave of deep learning.

Deep learning methods need to be trained with enough learning examples. Thus, content tagging is required to provide guidance to teach the machine to see and sense fashion. We have to tell machine what

a jumpsuit look like, or what kind of outfits belong to the style of hipster. When the machine learns to recognize what is in the image, they are taught to parse the image and understand the content. We expect the machine to achieve finer tasks such as marking the human body in pixel level, or even locate where the hands or feet are. Of course, each pixel in the training images is supposed to be annotated to indicate if it is in the human body or in the background. To go a further step, automatic relation discovery is also an important course for the deep learning machines to take. For instance, machines have seen too many images that a shirt always go with pants, so that they would not match two pairs of pants as an outfit.

Besides, vision is not the only way for humans to sense the environment. Other sources from audition, olfactory, and text contribute a lot to provide cues in the colorful world. Therefore, it might not be enough to equip multimedia technology with fashion sense solely from fashion images. In the fashion area, currently another accessible information is text, which can be easily obtained from many fashion blogs. By relating text and images, it gives extra guidance to better understand images from semantic perspectives.

INSPIRING RESEARCH

The development in multimedia has provided researchers with basic tools to deal with the problems in fashion. However, there are still particular issues in fashion domain, including the detection of clothes attributes and landmark, as well as how to utilize them to benefit further fashion analysis. Here, we would like to introduce the progress in fashion research, which would likely accelerate the conversion from laboratory to startups in the fashion industry, and even impact people's lifestyle in the near future.

The topics of fashion research in the literature of multimedia are summarized in Figure 1. One of the branches in fashion research is low-level pixel computation, which aims to label each pixel in the scene (i.e., fashion parsing, landmark detection, and pose estimation). Supported by the low-level pixel computation, mid-level fashion understanding emerges, including style classification, item detection, and attribute prediction. More recently, high-level fashion analysis blossoms in recommendation (including outfit

matching and retrieval), fashion synthesis, and fashion trend prediction, which lead us a step closer to a fashion intelligent assistant.

Low-Level Pixel Computation

Fashion parsing refers to generate pixel-level labels on the image, including hair, head, upper clothes, pants, etc., which understands images in a finer granularity. It is a very challenging problem, since the number of garment types, the variation in configuration, and appearance are enormous. Currently, fashion parsing is considered to a restricted domain, where the potential labels are predefined.

Besides, due to the clothes structure, clothes key points are helpful to extract discriminative representations to understand fashion images. These key points are referred as fashion landmarks, which were first introduced by the research team in the Chinese University of Hong Kong with a well-annotated fashion landmark dataset DeepFashion.³ Landmark detection is to localize these key points located at functional regions. Regression models are usually employed to achieve landmark detection.

Another topic in low-level pixel understanding lies in pose estimation. Different from landmark detection, human pose estimation is the process of inferring 2-D or 3-D human body part positions, such as head, torso, shoulders, etc. The results of pose estimation and fashion could benefit each other and be refined iteratively.⁴

Mid-Level Fashion Understanding

Based on pixelwise labels, fashion images can be understood at a higher level, making machine to predict what is in the fashion images. First, the machine needs to locate the fashion items. Item detection aims to detect various fashion items that a person in the image is wearing or carrying. To achieve this task, item proposals are generated and classified with many state-of-the-art object detection methods including Faster-RCNN, SSD, YOLO, etc.

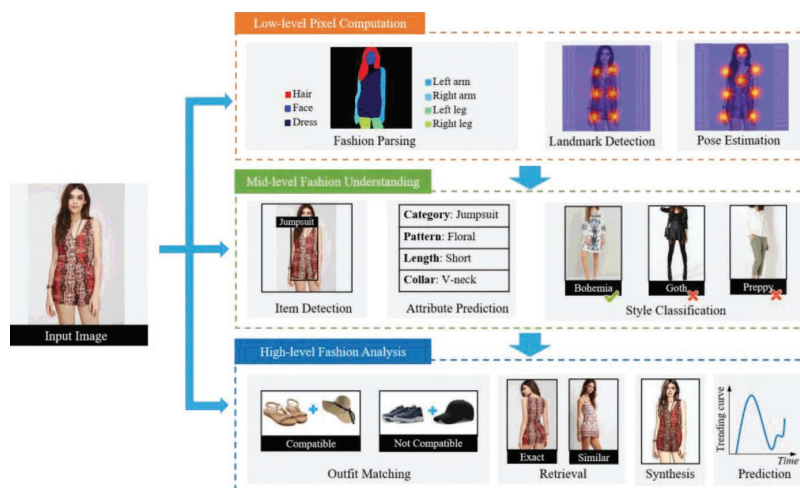


FIGURE 1. The research topics in fashion (model images are from DeepFashion Dataset³ and FashionStyle14 Dataset⁵).

Then attribute prediction can be applied to resolve the detected fashion items. The attributes predefined by human usually include category, texture, fabric, shape, part, and style, which are associated with a local patch in the images. Thus, the attribute prediction depends on the localization of landmarks or human skeletal joints. Guided by the key points, information in the local patches is explicitly or implicitly extracted. Combined with the deep neural network,³ we are able to infer predefined attributes with lots of annotated data. However, annotating attributes is quite a tedious work that costs much time and efforts. Recently, automatic attribute discovery has attracted researcher's attention. These methods usually explore the visual semantic relationships from the fashion image and corresponding text. The visual attributes are discovered according to the neural activations.⁶

In addition, style classification plays a key role in mid-level fashion understanding. The clothes people wear reveal the personal style, bohemian or pinup, Goth or preppy. The pipeline for style classification is extracting features from images and employing classifiers to identify the specific style.⁵ How to discover elements that define a style becomes the key issue.

High-Level Fashion Analysis

When it comes to fashion analysis, many people first think of understanding people's preference to benefit personalized services.⁷ With the help of mid-level fashion understanding, it is possible to realize customized

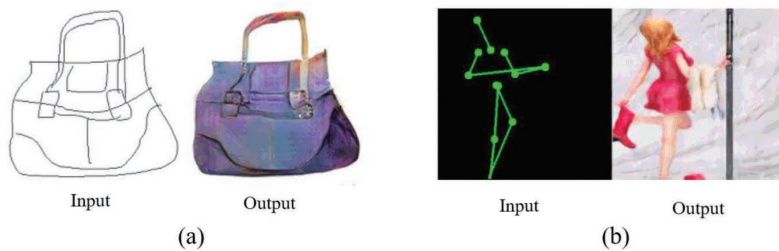


FIGURE 2. Fancy works from GAN. (a) Sketch to image from Pix2Pix.⁹ (b) Pose to image from DAGAN.¹⁰

recommendation. There are mainly two aspects in recommendation: outfit matching and retrieval. Outfit matching aims to find compatible candidates when a particular item is given. For example, jeans are more desired if a white T-shirt is provided. Currently, some researchers deal with the task of outfit matching by searching visual semantic embedding space,⁸ where compatible items are supposed to stay closely. For retrieval, earlier research works focus on similar retrieval. The returned item is only required to be similar to the query in color or style. Exact retrieval to find the exact clothing item is also well developed recently. This task is more challenging but also more demanding in practice.

In the meantime, the success of generative adversarial network (GAN) encourages researchers to devote themselves for fashion synthesis. As shown in Figure 2, GAN is like an amazing brush that could transfer desired textures from arts or other sources to the target items such as coats, shoes, and bags,⁹ or even generate images according to text or skeleton.¹⁰ The application of GAN in fashion field could benefit virtual try-on, product-human transfer, and ultimately fashion design.

Fashion trend analysis is another rising research topic. With the big data from Internet, researchers are able to investigate the world's fashion landscape from a geographic perspective,¹¹ or predict the future of fashion from temporal analysis.¹² Relying on the statistics of social media, they analyze the visual elements shared across images. Finally, researchers track the styles' spatial or temporal trajectories and provide the future trends.

FUTURE DIRECTIONS

We are excited to see that multimedia technologies

have gradually shown their power in the fashion industry. Merchandisers, designers, and customers could all work alongside technology. Statistical models have been developed using large amounts of data to help merchandisers make decisions in marketing. Several applications from startups have been seamlessly embedded in the products for customers. However,

the demanding of designers is far from being satisfied. Intelligent design is still a challenge so far. Fortunately, there have been successful examples showing that the line between technology and creativity is gradually blurring. Machines today can write poems, compose music, and create artwork that no one can tell is machine generated. So the fashion design is surely not beyond the capabilities of multimedia. Fashion designers armed with multimedia technology will be able to transfer their radical new ideas into products more efficiently. Intelligent design will amplify designer's creativity rather than replace it. It will become an indispensable tool for designers, ultimately improving and fueling the creative processes. To realize technology-assisted fashion design, here we list some promising research directions that might be helpful.

► Fashion generation conditioned on text:

Automatic synthesis of realistic images from text would be interesting and important in the design process. Although recent years have seen the progress in this area, the generation results are only satisfying in limited objects such as birds, flowers, etc. The diverse attributes of fashion images in color, pattern, style, and even fabric make it challenging to generate realistic fashion images from text. For example, the conditional text could be "long sleeve supple leather jacket in light blue with zip closure at front." Research works on how to handle such complex conditions as well as data sources should be inspired.

► Fashion assessment:

Currently the evaluation metric for fashion generation is based on inception score or human preference score. However, inception score focuses more on the image

quality, regardless of the aesthetic factors. Human preference score obtained from a small group can be easily influenced by the users' personal preference or the environment. Thus, it is eager to build a novel fashion assessment metric that is objective and robust.

- ▶ *Unsupervised or weakly supervised fashion research:* Many fashion tasks rely heavily on annotated data, such as fashion parsing, pose estimation, landmark detection, item detection, etc. But densely labelling is a tedious and time-consuming process. More efforts on unsupervised or weakly supervised learning in fashion domain are necessary to relieve the labelling burden. One possible solution is to utilize text information along with the visual data.
- ▶ *2D/3D virtual try-on:* Although there are much advancement in fashion virtual try-on, current methods are still far from this goal. On the one hand, clothing is much more difficult to render due to deformation and occlusion. On the other hand, 3-D human body modeling for arbitrary people is still challenging. The virtual try-on is particularly important for real applications. New models and learning methods need to be developed to capture personalized shape detail and clothing geometry.

The collision of multimedia technologies and fashion is still in its infancy. There is a long way to go. But the sparkles have lighted up our passion to trigger the revolution of fashion. We are looking forward to the transforming future of fashion and multimedia. 🌈

REFERENCES

1. "Global fashion industry statistics—International apparel," 2018. [Online]. Available: <https://fashionunited.com/global-fashion-industry-statistics>
2. A. Joseph, "IBM Watson+Tommy Hilfige +FIT: The AI upgraded designers," Jan. 26, 2018. [Online]. Available: <https://futur404.com/elevated-designer/>
3. Z. Liu, et al., "Deepfashion: Powering robust clothes recognition and retrieval with rich annotations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1096–1104.
4. K. Yamaguchi, et al., "Parsing clothing in fashion photographs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3570–3577.
5. M. Takagi, et al., "What makes a style: Experimental analysis of fashion prediction," in *Proc. IEEE Int. Conf. Comput. Vis. Workshop*, 2017, pp. 2247–2253.
6. S. Vittayakorn, et al., "Automatic attribute discovery with neural activations," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 252–268.
7. S. Liu, L. Liu, and S. Yan, "Fashion analysis: Current techniques and future directions," *IEEE Multimedia*, vol. 21, no. 2, pp. 72–79, Apr.–Jun. 2014.
8. X. Song, et al., "Neurostylist: Neural compatibility modeling for clothing matching," in *Proc. ACM Multimedia Conf.*, 2017, pp. 753–761.
9. P. Isola, et al., "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1125–1134.
10. S. Ma, J. Fu, C. W. Chen, and M. Tao, "DA-GAN: Instance-level image translation by deep attention generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5657–5666.
11. Y. T. Chang, et al., "Fashion world map: Understanding cities through streetwear fashion," in *Proc. ACM Multimedia Conf.*, 2017, pp. 91–99.
12. Z. Al-Halah, R. Stiefelhagen, and K. Grauman, "Fashion forward: Forecasting visual style in fashion," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 388–397.

SIJIE SONG is currently working toward the PhD degree with Peking University, Beijing, China. Her research interests include computer vision and image processing. She received the BS degree in computer science. Contact her at ssj940920@pku.edu.cn.

TAO MEI is a technical vice president with JD.com and the deputy managing director of JD AI Research, where he also serves as the director of Vision and Multimedia Lab. His lab is focusing on the research, development, and innovation of computer vision and multimedia, with a broad range of applications in retail, logistics, finance, cloud computing, as well as fashion and design. Prior to joining JD.com in 2018, he was a senior research manager with Microsoft Research Asia in Beijing, China. He received the BE and PhD degrees from the University of Science and Technology of China, Hefei, China. Contact him at tmei@jd.com.

Learning to Synthesize and Manipulate Natural Images

Jun-Yan Zhu, *Massachusetts Institute of Technology*

This article originally appeared in

IEEE **Computer Graphics**
AND APPLICATIONS
vol. 39, no. 2, 2019

EDITOR'S NOTE:

Jun-Yan Zhu received the 2018 SIGGRAPH Outstanding Doctoral Dissertation Award.

Humans are avid consumers of visual content. Every day, people watch videos, play games, and share photos on social media. However, there is an asymmetry—while everybody is able to consume visual data, only a chosen few are talented enough to express themselves visually. For the rest of us, most attempts at creating realistic visual content end up quickly “falling off” what we could consider to be natural images. In this thesis, we investigate several machine learning approaches for preserving visual realism while creating and manipulating photographs. We use these methods as training wheels for visual content creation. These methods not only help users easily synthesize realistic photos but also enable previously not possible visual effects.

Every day, people consume astounding amounts of visual content, as they watch videos, play digital games, and share photos on social networks. For example, Facebook alone reports 3 million photo uploads per day, and YouTube sees 300 hours of video uploaded every single minute. As of this writing, an estimated 4.7 trillion photos have been taken since the invention of photography, of which around 20% are from the past 12 months.

The availability of this big visual data has enabled researchers to develop powerful visual understanding methods, which aim at compressing visual data, such as images or videos, into abstract representations. For example, a machine can convert an image into a single word “street” via scene classification, into multiple words “pedestrians,” “motorcycles,” and “buildings” via

object detection, or even into a human-like sentence such as “A group of people riding motorcycles on a busy city street” via image captioning. However, there is another side of visual intelligence: visual synthesis. It operates in the opposite direction, from compact concepts back into visual data. In this dissertation,¹ I would like to teach machines to imagine the realistic visual world from both low-level visual concepts such as texture and shape, as well as high-level semantic concepts like objects and scenes.

But why would *visual synthesis* be useful? Indeed, it can help address the one-sided nature of communication between humans and vast amounts of visual data. While we all perceive information in the visual form through photographs, paintings, sculpture, videos, etc., only a chosen few are talented enough to effectively express themselves visually. This imbalance manifests itself even in the most mundane tasks. Consider an online shopping scenario. A user looking for shoes has found a pair that mostly suits them, but

Digital Object Identifier 10.1109/MCG.2019.2891309

Date of current version 22 March 2019.

perhaps they would like them to be a little taller, or wider, or in a different color. How can the user communicate their preference to the shopping website? How can we recreate the vivid visual world representing the user's mental picture? If the user is an artist, then a few minutes with an image editing program will allow them to transform the shoe into the desired one, and then a simple image-based search can find it. However, for most of us, even a simple image manipulation in Photoshop presents insurmountable difficulties. One reason is the lack of "safety wheels" in visual synthesis and editing: any less-than-perfect edit immediately makes the image look completely unrealistic. To put it another way, the classic visual synthesis and manipulation paradigm does not prevent the user from "falling off" the set of natural images.

In this dissertation, I investigated several data-driven visual synthesis approaches for preserving visual realism while creating and manipulating photographs. Most prior works rely on low-level visual cues, such as color and texture, for modeling visual realism or hand-crafted engineering to reduce artifacts for individual applications. Unlike these methods, we propose to model visual realism directly from large-scale collections of natural images. We then use the learned models as training wheels for visual content creation. We define a class of image synthesis and manipulation operations, constraining their outputs to look realistic according to the learned visual realism models.

We first build our methods on two classic machine learning paradigms: discriminative learning and generative modeling. For discriminative learning, we train a classifier to predict visual realism and aesthetics. The classifier can then be used for choosing the best-looking photos from a collection, as well as computing the optimal parameters of an image editing program. For generative modeling, we directly enforce the synthesized results to look realistic, as characterized by the learned image generation models, while satisfying user constraints. We explore several variants of this idea, from a 19th-century-old image averaging model² to a state-of-the-art deep generative model.³ We present real-time applications such as visual data exploration and image editing.

In the above-mentioned methods, visual realism modeling and image synthesis algorithms serve as two independent system components which are

designed and optimized separately. To take advantage of end-to-end learning, we combine the realism classifier and image synthesis program into a single image-to-image translation pipeline. Inspired by a recently proposed method known as generative adversarial networks (GANs),³ we train an image generation network to translate inputs (such as user sketches) directly to output results, while simultaneously teaching a realism classifier to distinguish the synthesized results from natural photographs. Through many qualitative results and human perceptual studies, we demonstrate that our proposed methods help users easily synthesize more visually appealing photos, compared to the previous state-of-the-art. We also show that our methods enable many new visual effects not possible before, such as turning a running horse video into a zebra video, generating real photographs from Impressionist paintings, and converting an image captured at night into day images with different types of lighting, sky, and clouds.

DISCRIMINATIVE LEARNING OF VISUAL REALISM AND AESTHETICS

A key open problem in data-driven image synthesis is how to make sure that the synthesized image looks realistic, i.e., if it lies in the set of natural images. The human ability to very quickly decide whether a given image is "realistic" is impressive. In contrast, characterizing this in closed form for a computer is very difficult, and this is precisely what makes good computer graphics and photographic editing so difficult. So many things must be "just right" for a human to perceive an image as realistic, while a single thing going wrong will likely hurtle the image down the Uncanny Valley.

So what makes an image appear realistic? This is one of the most long-standing problems in computer vision. Back in 1999, Huang and Mumford⁶ studied the local statistics of images, ranging from a simple histogram of intensity values to the joint distribution of texture features, such as wavelets. But until now, no existing machine learning method has been shown to reliably tell whether a given image looks natural. This is because the spectrum of unrealistic images is much larger than the spectrum of natural ones. If this were not the case, photorealistic computer graphics would have been solved long ago.

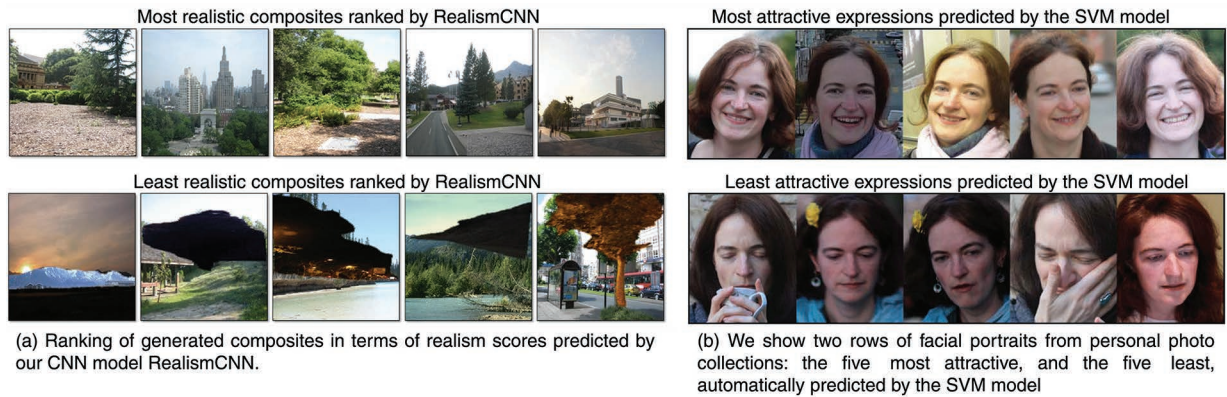


FIGURE 1. Learning to predict visual realism and photo aesthetics. (a) CNNs are trained to predict if an image composite looks realistic or not.⁴ (b) A support vector machine (SVM) learns to predict the attractiveness of facial expressions for a given subject.⁵ This model can automatically mine attractive photos (top row) from a personal photo collection.

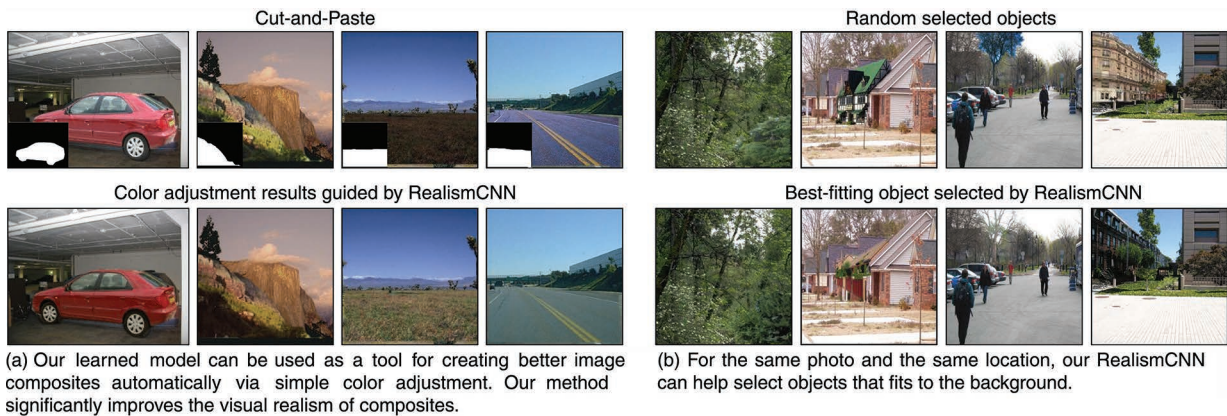


FIGURE 2. Our model RealismCNN⁴ can help create more realistic composites. Our model can (a) improve color compatibility between an object and its background and (b) choose the best object that fits a given background.

Predicting and Improving Visual Realism

In our work, RealismCNN,⁴ we took a discriminative approach to address a particular instance of the above problem, training a high-capacity discriminative model, a convolutional neural network (CNN),⁷ to estimate the realism of spliced image composites. Since it is difficult to obtain enough human-labeled training data to learn what looks realistic, we instead learned to classify between real images and automatically generated composites. Surprisingly, the resulting classifier can actually predict how realistic a new composite would look to a human, as shown in Figure 1(a). Interestingly, the model appears to be picking up on cues about visual realism, as demonstrated

by its ability to rank image composites by their perceived realism. Our model mainly characterizes visual realism regarding color, lighting, and texture compatibility.

Moreover, we demonstrated that our learned model can be used as a tool for creating better image composites automatically via simple color adjustment. Given a low-dimensional color mapping function, we directly optimized the visual realism score predicted by our CNN model. Our method outperforms previous color adjustment methods on a large-scale human subject study (see Figure 2(a)). We also used our model to choose an object from a category that best fits a given background at a specific location (see Figure 2(b)).

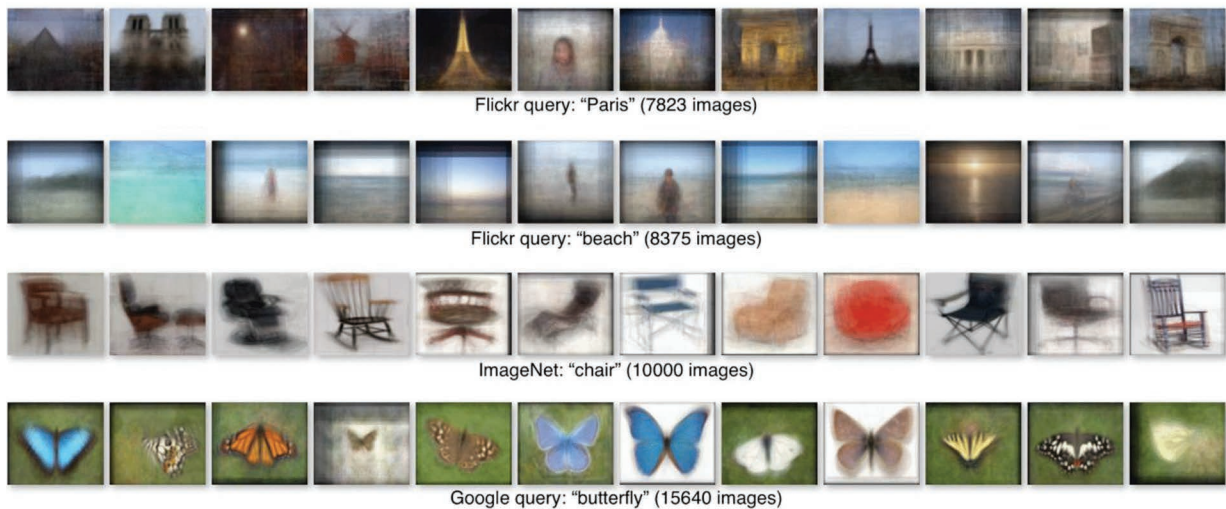


FIGURE 3. Examples of interactively discovered modes in the data using AverageExplorer.⁸ See our interactive interface and more vision and graphics applications in our supplemental video.

Modeling and Improving Photo Aesthetics

Along the same line of thought, we developed a discriminative learning approach for modeling and improving the photograph aesthetics of portraits.⁵ We used crowdsourcing to score the portraits for their attractiveness. We then used these scores to train an SVM that can automatically predict attractiveness of different expressions of a given person. As shown in Figure 1(b), the model can evaluate the attractiveness of facial expressions and can automatically mine attractive photos from personal photo collections. Furthermore, we demonstrated a training app that helps people learn how to mimic their best expressions. See our training demo⁵ for more details.

GENERATIVE MODELING FOR VISUAL EXPLORATION AND SYNTHESIS

We have demonstrated that discriminative classifiers can learn to accurately predict the photorealism of the result regarding a single factor, such as color compatibility⁴ or facial expressions.⁵ However, many factors play a role in the perception of realism. While a learned classifier picks up on one visual cue such as color, others such as lighting, semantics, scene layout, perspective, etc., are also important. Learning a classifier for all the factors is challenging due to the lack of training data. Besides, to produce a single result, we have to first generate many candidate results and rank them

with a classifier. Therefore, the running time is exponential to the number of the parameters in an image editing program, which makes it computationally prohibitive for real-time graphics applications. We can perhaps search for the best parameters if the image editing program is differentiable, but the search space is still huge.

To address the issues above, we directly model the natural image prior via generative models and constrain the output of an image manipulation tool to look realistic according to the learned prior. Generative models can directly synthesize a realistic image given a low-dimensional vector as input, often in real-time. This advantage allows us to build interactive data-driven exploration and editing interfaces.

Visual Exploration via Image Averaging

We first studied a simple and old image generation model: image averaging, pioneered by Sir Francis Galton² in 1878. The image averaging model constrains the output result to be a convex combination of the database images. Building on this simple model, we presented AverageExplorer, an interactive framework⁸ that allows a user to rapidly explore and visualize a large image collection using the medium of average images. Our interactive, real-time system provides a way to summarize large amounts of visual data by weighted averages of an image collection, with the weights reflecting user-indicated importance. As shown in Figure 3, we not only captured the mean of

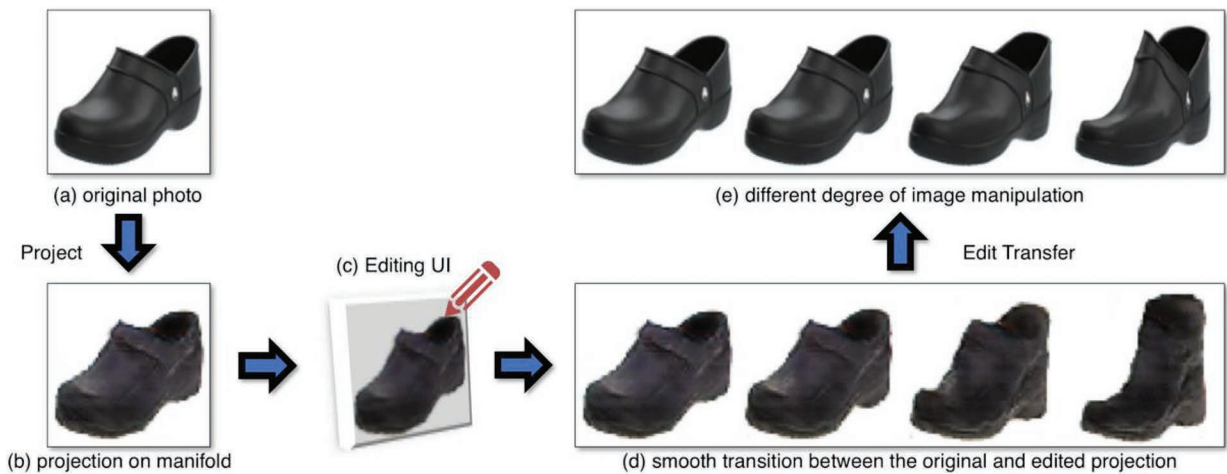


FIGURE 4. Interactive image editing with GANs^{3,9}: we first project an original photo (a) onto a low-dimensional representation (b) by regenerating it using GAN. A user can then modify the color and shape of the generated image (d) using various brush tools (c) (for example, dragging the top of the shoe). Finally, our edit transfer algorithm applies the same amount of geometric and color changes to the original photo to achieve the final result (e). See our interactive image editing video for more details.

the distribution, but also a set of modes, discovered via interactive exploration. We posed this exploration in terms of a user interactively “editing” the average image using various types of strokes, brushes, and warps, with each user interaction providing a new constraint for updating the average. Together, these tools allow the user to simultaneously perform two fundamental operations on visual data, user-guided clustering and user-guided alignment, within the same framework. The supplemental video shows that our system is useful for various computer vision and graphics applications.

Visual Manipulation With Deep Generative Models

Simple image averaging models² can serve as an intuitive and artistic medium for visual data exploration. However, the generated results often look blurry and far from realistic, even with the alignment algorithm implemented in AverageExplorer.⁸ Furthermore, novel results cannot always be represented as linear combinations of dataset images. To produce more natural and diverse results, we turned to recently developed deep generative models due to the quality and complexity of their samples. In the last two years, there has been rapid advancement, fueled mainly by the development of the GANs.³ In particular, recent work has shown visually impressive samples drawn from the

model.⁹ However, two reasons prevent these advances from being useful in practical applications at this time. First, the generated images, while impressive, are still not quite photo-realistic. Second, more importantly, these generative models are set up to produce images by sampling a vector, typically at random. So, these methods are not able to create or manipulate visual content in a user-controlled fashion.

To improve the quality and user control, we used the GANs to learn the prior of natural images, but we did not actually employ it for image generation. Instead, we used it as a constraint on the output of various image manipulation operations, to make sure the results follow the learned prior at all times. This idea enables us to reformulate several editing operations, specifically color and shape manipulations, in a natural and data-driven way. The model automatically adjusts the output keeping all edits as realistic as possible (see Figure 4). The supplemental video shows three image editing applications based on our proposed interface iGAN¹⁰: (1) manipulating an existing photo based on an underlying generative model to achieve a different look (shape and color), (2) “generative transformation” of one image to look more like another, (3) generating a new image from scratch based on user’s scribbles and warping UI.

Figure 4 illustrates our approach. Given a real photo, we first projected it onto a low-dimensional

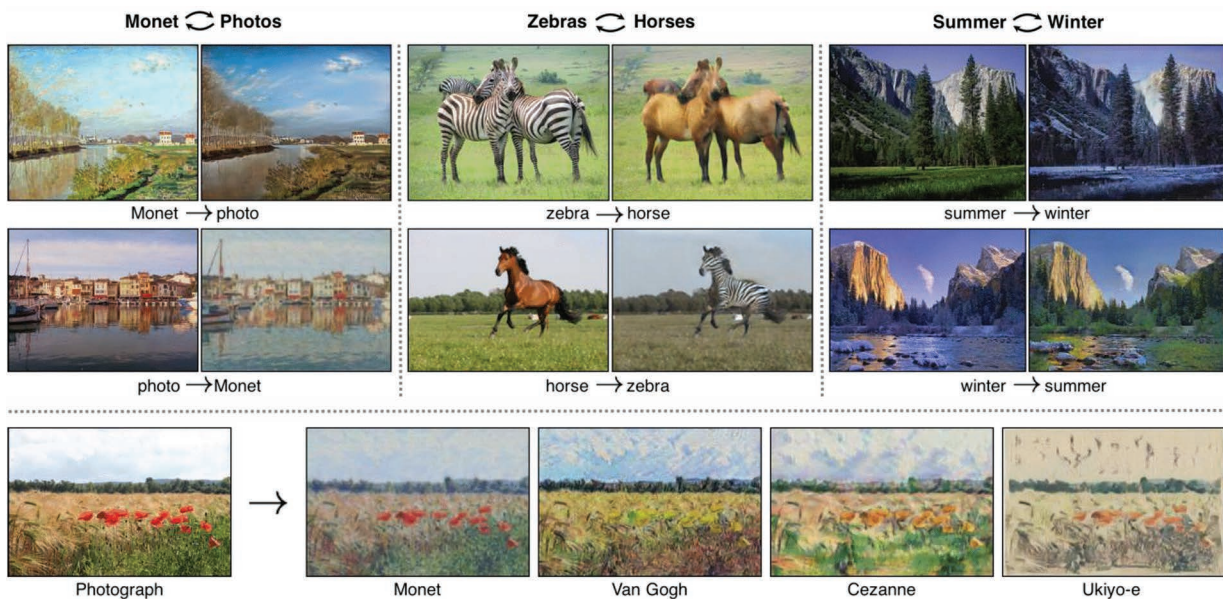


FIGURE 5. Given any two unordered image collections X and Y , our algorithm Cycle-Consistent Adversarial Networks (CycleGAN)¹¹ learns to automatically “translate” an image from one into the other and vice versa. Example application (bottom): using a collection of paintings of a famous artist, learn to render a user’s photograph into their style.

representation parameterized by GANs. Then, we presented a real-time method for generating a desired image that both satisfies the user’s edits and stays close to the natural image prior. Unfortunately, the generative model usually loses low-level details of the input image. Therefore, we proposed an image alignment method that can transfer the edits from generated image to the original photo and produce the final result.

IMAGE-TO-IMAGE TRANSLATION

In the methods above, the modeling of visual realism and the algorithms for image synthesis are designed and optimized separately. To solve such coupled problems jointly, my coauthors and I proposed a general-purpose image-to-image translation framework, *pix2pix*,¹² where we learned a graphics program to translate inputs (e.g., user sketches) directly to output results, while simultaneously teaching a realism classifier to distinguish the synthesized results from real photos. This method can achieve visually appealing results for many problems that traditionally would require very different formulations, such as synthesizing photos from label maps, turning sketches into pictures, and colorizing images.

Unpaired Image-to-Image Translation

Training a *pix2pix*¹² model requires paired training data, which is often difficult and expensive to collect. For example, only a couple of small datasets exist for tasks like semantic segmentation. Obtaining input-output pairs for graphics tasks such as artistic stylization can be even more difficult since the desired output often requires artistic authoring. To address this issue, we proposed a model, CycleGAN,¹¹ for learning to translate an image x from a source domain to an image y in a target domain, in the absence of paired examples. Our goal is to learn a mapping $G: x \rightarrow y$, such that the distribution of images from $G(x)$ is indistinguishable from the data distribution of target images y using an adversarial loss. Because this mapping is highly under-constrained, we enforced the additional constraint that translation should be “cycle consistent,” in the sense that if we translate, e.g., a sentence from English to French, and then translate it back from French to English, we should arrive back at the original sentence. In practice, we coupled it with an inverse mapping, $F: y \rightarrow x$, and introduce a cycle consistency loss to push $F(G(x)) \approx x$ (and vice versa). Figure 5 shows qualitative results on several tasks where paired data does not exist, including collection style transfer,

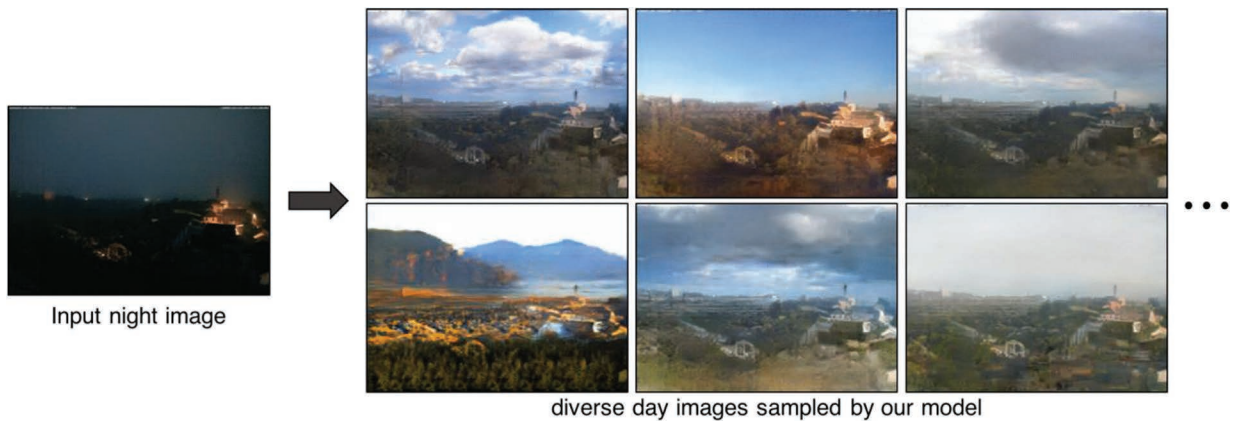


FIGURE 6. Multimodal image-to-image translation¹³: given an input image from one domain (night image of a scene), we can model a distribution of potential outputs in the target domain (corresponding day images), producing both realistic and diverse results.

object transfiguration, season transfer, and photo enhancement.

Multimodal Image-to-Image Translation

Our methods^{11,12} can only produce one plausible result given an input image. However, many image-to-image translation problems are inherently ambiguous, as a single input image may correspond to multiple possible outputs. To address the issue above, my coauthors and I proposed a conditional generative modeling framework¹³ to model a distribution of possible outputs. The ambiguity of the mapping is distilled into a low-dimensional vector, which can be randomly sampled at test time. A generator learns to map the given input, combined with this vector, to the output. We explicitly encouraged the connection between output and the vector to be invertible. This objective helps prevent a many-to-one mapping from the vector to the output during training, also known as the problem of mode collapse and produces diverse results. Our proposed method encourages bijective consistency between the encoding vector and output modes. Figure 6 shows an example result of using our method to generate a day image from a night one. Given the same input night image, our method can synthesize day images with different sky color, various lighting effects on the ground, and different density and shape of clouds.

IMPACT AND APPLICATIONS

We have described a few data-driven approaches for learning visual realism directly from large-scale image collections. We used the learned visual realism models for realistic image synthesis and editing. The presented approaches not only produce more visually appealing results compared to previous methods that rely on hand-crafted engineering or heuristics, but also enable several new visual effects, unachievable by any prior work.

Research

Several of our algorithms above, most notably CycleGAN¹¹, have been used in different fields including modifying the style of images, three-dimensional (3-D) models, and animation (graphics), generating synthetic training data (vision), adapting policy from simulations to the real world (robotics), converting MRIs into CT scans (medical imaging), synthesizing voice and music (audio processing), and changing the sentiment of text (NLP). Even as authors, we are consistently surprised by these creative applications, many of which we never considered ourselves. These individual applications demonstrate the utility of our work as general-purpose unsupervised learning methods for transforming data across different domains.

Education

CycleGAN¹¹ and pix2pix¹² have been taught in several universities worldwide such as Stanford, MIT, UC



FIGURE 7. Image search with a mental picture: our image synthesis system could be potentially used as an image search interface for efficient browsing.

Berkeley, University of Toronto, as well as in leading MOOC platforms such as Udacity and Fast.ai. CycleGAN has also been used as homework and projects in the courses above. CycleGAN and pix2pix have also appeared or will appear in several textbooks.

Software

We have open-sourced many of our research projects and online demos for public use. As a result, many researchers and practitioners have built new models and practical applications based on our work. Among them, three projects were listed as the top 30 machine learning projects in 2017 (out of 8800 projects) according to a recent survey.¹⁴ One of my favorite application is cat and dog transformation, where an Internet user from Japan transformed a dog into a cat using CycleGAN.

Art

A number of our tools have been used widely not just by researchers and developers, but also by visual artists. For example, many professional artists and amateurs have used CycleGAN to perform collection artistic style transfer, including turning a face portrait into anime art, transforming a real landscape photo into stained glass art, or rerendering everyday drink and food pictures with flowers sketches.¹⁵ These results demonstrate exciting, artistic collaborations between humans, machines, and algorithms.

FUTURE DIRECTIONS

Below, I discuss several potential future directions, building on our current image synthesis and manipulation algorithms.

Communication Interfaces Between Humans and Visual Data

We are living in an age of big visual data. In the end,

it is humans, not machines, who are overwhelmed by trillions of photos on the Internet. How can we help humans better understand this vast visual space, to see what is out there? While keywords are often used as a proxy for indexing visual content, the visual world is much richer than what can be named by words; and yet this process is inevitably lossy and noisy. Fortunately, generative models have the potential to provide a useful human-data interface. If users can synthesize their mental pictures using generative models, systems can understand the intent and retrieve results quickly and accurately. We performed a preliminary study with the AverageExplorer system.⁸ As shown in Figure 7, the user may start with a set of black high heels (left), and then decide on a different color. Simply drawing a stroke with the desired red color on the generated image (center) retrieves the available red shoes of similar styles. If the user then wants a heel with more support, adding an edge stroke at the bottom of the shoe switches the heel style (right). In the future, generative modeling will be a promising way to connect a user's mental picture directly to large-scale datasets.

Learning Graphics for Building Autonomous Systems

Historically, graphics research has mainly focused on creating content for humans. However, computer graphics can also be used to create richly annotated virtual environments for training autonomous systems. Unfortunately, the performance of such systems can suffer if the virtual environments used in training look different from the real-world environments in which the systems are used. To bridge this gap, our recent work¹⁷ learns to adapt virtual CG inputs to the appearance of a real-world test domain (see Figure 8), significantly improving the model's generalizability.

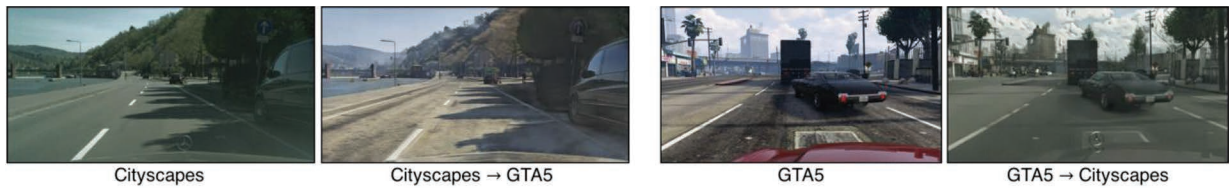


FIGURE 8. CycleGAN¹¹ can learn to bridge the gap between the synthetic virtual world such as the Grand Theft Auto (GTA5) and real-world computer vision datasets such as Cityscapes dataset.¹⁶

Moving forward, I plan to build differentiable graphics pipelines that can easily adapt these methods to a wide range of new datasets and tasks. The parameters of the graphics engine will be learned given an objective, rather than specified by humans. For example, the model would learn to produce training images optimized for a particular vision or robotics application while matching the appearance of a target domain.

Beyond 2-D Image Generation

Deep generative models such as GANs³ have been able to produce visually appealing images, enabling computer vision and graphics applications as mentioned in this paper.^{11,12} However, most of the recent learning-based methods focus on generating 2-D images. Consider a generative model that can synthesize a car. But what will this car look like from a new viewpoint? Can we transfer this car's look to a different 3-D CAD model? What does this car look like when it drives on the road and interacts with other cars and pedestrian? None of the above questions can be answered with the current 2-D image generation models. Besides, it is hard to deploy these 2-D models to many real-world applications such as robotics, virtual reality, and gaming. Therefore, to build intelligent machines that can recreate and imagine our visual world, we have to go beyond 2-D image generation. We need to model the 3-D nature of our visual world as well as recreate its dynamics. We have studied simplified forms of the above problem statements such as synthesizing 3-D textured objects,¹⁸ generating realistic videos conditional on input videos,¹⁹ and synthesizing and interpolating light field videos.²⁰ These methods still work in restricted domains or require additional inputs as guidance. In the future, a full generative model that can synthesize 3-D or 4-D continuous visual experience would be desirable. 🌈

ACKNOWLEDGEMENT

I would like to thank my thesis committee members: A. A. Efros, J. Malik, R. Ng, and M. DeWeese for their guidance. Special thanks to my collaborators as well as students, faculty, and postdocs at UC Berkeley, CMU, and MIT who provided tremendous support for my research. This work was supported in part by a Facebook Graduate Fellowship. This paper has supplementary downloadable material at <http://ieeexplore.ieee.org>, provided by the author.

REFERENCES

1. J. Zhu, *Learning to Synthesize and Manipulate Natural Images*. Berkeley, CA, USA: UC Berkeley, 2017.
2. F. Galton, "Composite portraits made by combining those of many different persons into a single figure," *Nature*, vol. 18, pp. 97–100, 1878.
3. I. Goodfellow et al., "Generative adversarial nets," in *NIPS Proc.*, 2014.
4. J.-Y. Zhu, P. Krahenbuhl, E. Shechtman, and A. A. Efros, "Learning a discriminative model for the perception of realism in composite images," in *Proc. Comput. Vis. Pattern Recognit.*, 2015, pp. 3943–3951.
5. J.-Y. Zhu, A. Agarwala, A. A. Efros, E. Shechtman, and J. Wang, "Mirror mirror: Crowdsourcing better portraits," *ACM Trans. Graph.*, vol. 33, p. 234, 2014.
6. J. Huang and D. Mumford, "Statistics of natural images and models," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 1999, vol. 1, pp. 541–547.
7. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
8. J.-Y. Zhu, Y. J. Lee, and A. A. Efros, "AverageExplorer: Interactive exploration and alignment of visual data collections," *ACM Trans. Graph.*, vol. 33, p. 160, 2014.
9. A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *ICLR*, 2016.

10. J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *Proc. Euro. Conf. Comput. Vis.*, 2016, pp. 597–613.
11. J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. ICCV*, 2017, pp. 2223–2232.
12. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. Comput. Vis. Pattern Recognit.*, 2017, pp. 1125–1134.
13. J.-Y. Zhu et al., "Toward multimodal image-to-image translation," in *NIPS*, 2017.
14. Mybridge. 30 Amazing Machine Learning Projects for the Past Year, 2018.
15. H. Sarin, "Playing a game of GANstruction," 2018.
16. M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. Comput. Vis. Pattern Recognit.*, 2016, pp. 3213–3223.
17. J. Hoffman et al., "CyCADA: Cycle-consistent adversarial domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2018.
18. J.-Y. Zhu et al., "Visual object networks: Image generation with disentangled 3D representations," in *NIPS*, 2018.
19. T.-C. Wang et al., "Video-to-video synthesis," in *NIPS*, 2018.
20. T.-C. Wang, J.-Y. Zhu, N. K. Kalantari, A. A. Efros, and R. Ramamoorthi, "Light field video capture using a learning-based hybrid imaging system," *ACM Trans. Graph.*, vol. 36, 2017, Art. no. 133.

JUN-YAN ZHU is a Postdoctoral Researcher with Massachusetts Institute of Technology, Cambridge, MA, USA. His research interests include graphics, computer vision, and machine learning. He received the Ph.D. degree in EECS from UC Berkeley, Berkeley, CA, USA, in 2017, and the B.E. degree from Tsinghua University, Beijing, China, in 2012. He received a Facebook Fellowship and the 2017-18 David J. Sakrisson Memorial Prize. Contact him at junyanz@mit.edu.

Contact department editor Jim Foley at foley@cc.gatech.edu.



IEEE COMPUTER SOCIETY
Call for Papers

Write for the IEEE Computer Society's authoritative computing publications and conferences.

GET PUBLISHED
www.computer.org/cfp

 IEEE COMPUTER SOCIETY

 IEEE

Multimedia for Disaster Information Management

Shu-Ching Chen, *Florida International University*

The global cost of natural disasters in 2017 increased by almost twice from the ones in 2016. In particular, disastrous events such as wildfires, flood, hurricanes, and earthquakes caused the economic loss of \$306 billion in 2017, which is much higher than the average 10-year loss of \$190 billion.¹ Due to these significant devastations and losses, there is an urgent need for the development of disaster information management systems.

With the proliferation of social media and smart devices, a considerable amount of multimedia data can be collected before, during, and after a disaster. The disaster-based multimedia data include visual, audio, textual, geographical, sensor, and many other data types extracted from social networks, mobile phones, apps, and emergency websites. This valuable information is considered as "big data," which can be further integrated and utilized by the community residents and Emergency Management (EM) responders for better and faster decision-making and for more reliable and accurate situational awareness.

CHALLENGES

Despite the enormous opportunities provided by multimedia big data for different real-world applications, we are still far from having a comprehensive disaster information management system. This system should be able to handle the heterogeneous and large-scale data efficiently, analyze the information automatically, summarize and integrate the critical information, and finally deliver fast responses to residents and emergency responders.

The reliable and timely interactions of emergency services are required to diminish the human and property losses in a disaster recovery situation. Therefore, integrated communication and information systems


are presented in recent years to overcome the following challenges.^{2–4}

- › Time sensitivity and constraint.
- › The heterogeneous nature of data types and sources including static and dynamic data.
- › Trustworthiness of data sources and security.
- › Speed of access to data.

WHAT IS THE ROLE OF MULTIMEDIA AND SOCIAL MEDIA IN DISASTER RESPONSE AND RECOVERY?

Multimedia big data analysis is a key aspect of disaster information management systems, which effectively manages very large-scale data generated in disaster-based circumstances. Techniques such as machine learning and data mining together with multimedia and big data solutions can be utilized to address the aforementioned challenges. In general, the steps of disaster data analysis can be classified into preprocessing, data selection and summarization, predictive analytics, and postprocessing.⁴

Nowadays, social media is becoming a big part of everyone's life. Over the last decade, the impact of social media has been greatly shown in different applications from politics to businesses. Now, the question arises is how social media can enhance the existing disaster management systems. Obviously, it can be considered one of the most valuable sources of information for disaster preparedness, response, and recovery.⁵ It provides amazing opportunities for training, real-time communication, and data collection and analysis. Also, the location-based services in social media can assist emergency responders to save



human lives in catastrophic situations and victims to get help in a faster and more convenient way.

WHAT IS NEXT?

Existing multimedia-aided disaster information management systems utilize few sources of information (e.g., textual, Geographic Information System (GIS), sensors, etc.). It is essential to collect all critical information from various sources, summarize, and integrate them to adequately assist the EM. However, multimodal data models for disaster information management are still in its early stages and more efforts are required to achieve this goal.

It is possible to generate interactive models for disaster monitoring and situational awareness using visualization techniques. Currently, multimedia techniques⁶ and GIS-based models⁷ have been proposed to serve this purpose. However, such systems mostly rely on two-dimensional visualizations, which may not be easily interpretable by the nonexpert users. Therefore, we can utilize advanced three-dimensional visualization of real-life GIS data⁸ to help residents and the community for better preparedness and decision-making.

Today, artificial intelligence and machine learning have been extensively used in real-world applications. New technologies and tools such as drones, sensors, and robots can accurately collect data (damaged buildings, trapped victims, etc.) and provide information that cannot be easily gathered by the humans. Also, new machine learning techniques such as deep learning can be leveraged for understanding the deep knowledge of data for planning effective disaster information management and humanitarian relief.⁹ 🤖

REFERENCES

1. A. Rathi, "Global disasters in 2017 cost \$306 billion—nearly double the cost of 2016," *Quartz*, 2017. [Online]. Available: <https://qz.com/1162791/2017-natural-disasters-and-climate-change-effects-cost-the-world-306-billion/>
2. T. Li *et al.*, "Data-driven techniques in disaster information management," *ACM Comput. Surv.*, vol. 50, no. 1, Mar. 2017, Art. no. 1.
3. V. Hristidis, S.-C. Chen, T. Li, S. Luis, and Y. Deng, "Survey of data management and analysis in disaster situations," *J. Syst. Softw.*, vol. 83, no. 10, pp. 1701–1714, 2010.
4. M. E. P. Reyes, S. Pouyanfar, H. C. Zheng, H.-Y. Ha, and S.-C. Chen, "Multimedia data management for disaster situation awareness," in *Proc. Int. Symp. Sens. Netw., Syst. Secur.*, Springer, 2017, pp. 137–146.
5. C. H. Sherwood, "How social media impacts disaster response," *ZDNet*, 2011. [Online]. Available: <https://www.zdnet.com/article/how-social-media-impacts-disaster-response/>
6. Y. Yang *et al.*, "MADIS: A multimedia-aided disaster information integration system for emergency management," in *Proc. 8th IEEE Int. Conf. Collaborative Comput. Netw., Appl. Worksharing*, Oct. 14–17, 2012, pp. 233–241.
7. GeoVISTA, Pennsylvania State University, University Park, PA, USA, 2010. [Online]. Available: <http://www.geovista.psu.edu>
8. M. E. P. Reyes and S.-C. Chen, "A 3D virtual environment for storm surge flooding animation," in *Proc. 3rd IEEE Int. Conf. Multimedia Big Data*, Apr. 19–21, 2017, pp. 244–245.
9. X. Song, R. Shibasaki, N. J. Yuan, X. Xie, T. Li, and R. Adachi, "DeepMob: Learning deep knowledge of human emergency behavior and mobility from big and heterogeneous data," *ACM Trans. Inf. Syst.*, vol. 35, no. 4, Aug. 2017, Art. no. 41.

SHU-CHING CHEN is an eminent scholar chaired professor with Florida International University. Contact him at chens@cs.fiu.edu.

The Quantum Moonshot

Erik P. DeBenedictis, Zettaflops, LLC

This article reviews the IEEE Computer Society's opportunity and responsibility in the information revolution.

It looks like the world is embarking on a quantum moonshot. So I'll make a projection of how quantum will partially slip into the technology pipeline as a replacement for the transistor, but the effect will grow over time and lead to a new type of computer for solving problems that are difficult to fully anticipate today. Society is in the middle of an information revolution where humanity is developing the explicit means to control information using not only computers but also DNA, the Internet, proteins, and other physical carriers and processors of information.

The exponential growth in computing over the last half-century led humanity to create a pipeline for commercializing increasingly powerful computers. The pipeline had been fed by smaller and smaller transistors. Yet transistors are close to their limits, and the pipeline needs a new technology to fuel progress. Still, quantum is not a drop-in replacement for the transistor.

There are well-founded beliefs that there are profoundly different ways to compute, but building a biological computer, for example, would require invention of a new design hierarchy to replace gates, software, and so on. The key point for the IEEE Computer Society is that the innovators in biological computing call themselves bioengineers instead of computer scientists.

Quantum computers seem destined to fall in the

middle of a continuum bounded by CMOS and DNA. Quantum circuits look like classical circuits, with gate symbols connected by lines and time going left to right. This will allow tools for designing circuits and compilers to be upgraded and the users familiar with existing tools to employ the new ones with little retraining. While the Boolean logic values of false and true can be applied to (some) quantum gates, yielding classical reversible logic, qubits have phase and other properties that go beyond just the bit value. These additional properties change the meaning of what look like normal circuits to give them the essential properties that make quantum computers especially powerful for some problems.

There are efforts worldwide to see how quantum can fit into the existing pipeline that commercializes computers, but I think this is destined to be a story with multiple chapters. The efforts are sometimes called *moonshots*, so let's review the actual moonshot. There were six Apollo moon landings between 1969 and 1972 but none since. I'll argue that the Apollo moonshot was a huge success nonetheless.

The June 1964 issue of *IEEE Spectrum*¹ was a special issue with nine articles focused on semiconductor integrated circuit R&D, which was funded from the economic success of discrete transistor avionics and equipment for the Apollo space program. The business opportunity was leading industry to search for a way to scale integrated circuits, resulting in the 1965 scaling concept later called *Moore's law*. Moore's law powered an information revolution whose ongoing progress supports many IEEE Members, the IEEE Computer



Society, and many of IEEE's publishing and conference endeavors.

Although there were no additional moonshots after 1972, the Apollo moonshot had spinoffs that significantly benefitted the Computer Society and IEEE as a whole. By analogy, we should be prepared for the quantum moonshot to produce an as-yet-unimagined spinoff product that looks as much like current quantum computers as a smartphone looks like the computer in the Apollo lunar lander. While quantum computers can perform general-purpose computing, recent efforts in quantum benchmarking and supremacy have shown that quantum computers can address new applications and problems at a greater scale, such as the chemical simulation of larger molecules, or perform additional functions for some existing applications, such as optimization.

THE IEEE COMPUTER SOCIETY IS A PEOPLE-PROCESSING MECHANISM

I'm suggesting that quantum, if successful, will ultimately change the interaction between the Computer Society and technically oriented individuals over their lifetimes from middle school onward. Young people become interested in computing based on their affinity for building things with well-defined structure and moderate amounts of math. The Computer Society should motivate people with the specific concepts behind both classical and quantum information, and this will mean moving beyond Boolean logic and Turing machines.

Students, young or old, will need classes with textbooks and labs on quantum. To the extent that these materials exist today, they are for advanced undergraduates and graduate students in physics and require an extensive mathematical background. The quantum industry will specialize as it grows, which should make it possible to create mathematically

simplified educational materials for specialties, such as users of applications and frameworks, component suppliers, and so forth.

There are examples of these mathematically simplified abstractions today, yet they must be invented

WE SHOULD BE PREPARED FOR THE QUANTUM MOONSHOT TO PRODUCE AN AS-YET-UNIMAGINED SPOFF PRODUCT THAT LOOKS AS MUCH LIKE CURRENT QUANTUM COMPUTERS AS A SMARTPHONE LOOKS LIKE THE COMPUTER IN THE APOLLO LUNAR LANDER.

and validated by professor-level experts with a deep understanding of both the math and physics. I'm projecting that this will be a new and rigorous aspect of computer science and engineering in the future.

Professors are created from tenure-track faculty who compete on the number of peer-reviewed articles they write, which, in turn, depends on organizations that create and run conferences and journals that handle peer-reviewed research. As a large professional society with integral conference and publications businesses, this puts the IEEE Computer Society at the top of the supply chain.

SCIENCE, TECHNOLOGY, ENGINEERING, AND MATHEMATICS EDUCATION AND IMMEDIATE WORKFORCE AUGMENTATION

There is an immediate need to train a portion of the workforce to be effective in performing the quantum-related R&D that is being funded worldwide. Over time, this immediate need will become an

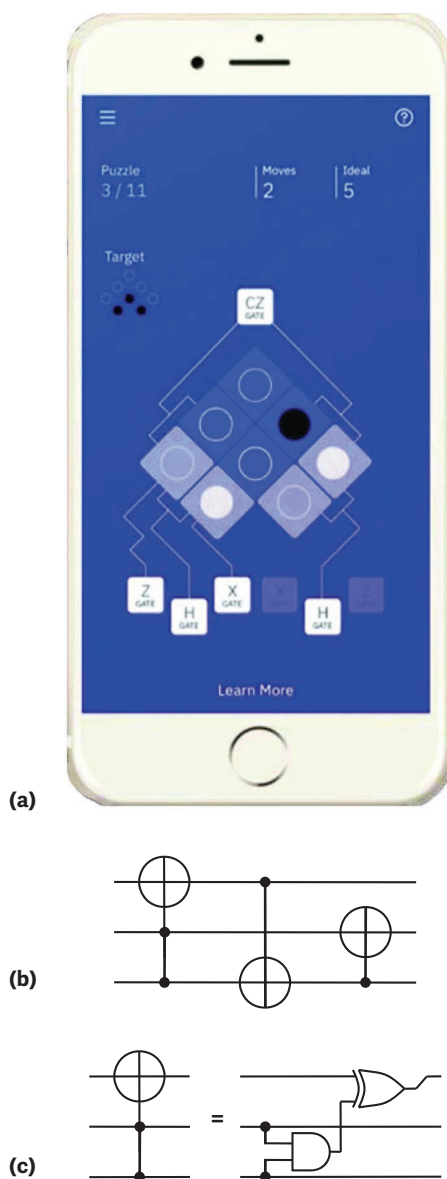


FIGURE 1. (a) The *Hello Quantum* application for teaching a simple quantum concept to a newcomer to the field. (Source: IBM; used with permission.) (b) A three-qubit quantum circuit in conventional digital circuit form. (c) The digital equivalent of the Toffoli gate or symbol. Applying the Toffoli function to its output results in reversal of the computation.

ongoing need to recruit young people to quantum as a part of science, technology, engineering, and math education efforts. This will also be essential to recruiting new IEEE Computer Society members.

The first step in attracting people to quantum is to convince them that the tasks they would perform

are compatible with their interests and abilities. Figure 1(a) is a smartphone game called *Hello Quantum*² from IBM, showing the states of two qubits as eight white, blue, or black circles. Pushbuttons implement gate operations, causing some of the circles to change color. The object of the game is to duplicate the pattern under “Target” along the left edge by pushing the correct sequence of buttons. It is entertaining in the same sense as a Rubik’s Cube but conveys valid information about how qubits differ from bits and how quantum gates work. Many people have created quantum abstractions, such as a baby book,³ musical notes,⁴ Powerball tickets,⁵ and others, but it remains to be seen which ones are effective at both teaching quantum concepts and making quantum interesting and approachable to newcomers.

APPLYING CLASSICAL COMPUTING KNOWLEDGE TO QUANTUM

Physicists invented the concepts in quantum computing from the properties of elementary particles and their interactions with one another and their environment. The math is complex, and the interactions with the environment increase the number of equations and variables.

However, David Deutsch⁶ reinterpreted key aspects of particle physics to the terminology of classical reversible logic, leading to something like a quantum logic family. Figure 1(b) shows an example circuit, with horizontal lines conveying information left to right. Figure 1(c) explains the gate symbol for the three-input, three-output Toffoli gate, after Tommaso Toffoli. The two-input conditional not, or CNOT, gate is a Toffoli gate with one control input removed. While quantum gates are intrinsically analog, the logic family abstraction applies well enough to simplify some tasks.

Technology and educational materials for classical computers have a role in quantum computing, often with minimal adaptation. Students and employees familiar with tools like the SPICE circuit simulator, Visual C++, logic design, and so forth can readily understand how those tools and skills can be adapted to quantum. If a qubit’s value is a bit, that is, a 0 or 1, then the diagram in Figure 1(b) has the same meaning irrespective of whether it is a classical reversible or quantum circuit. If a qubit has a value including phase

and entanglement, like $(|0\rangle + |1\rangle)/\sqrt{2}$, then the circuit must be a quantum circuit to function correctly. There are university classes that teach quantum concepts to electrical engineering students designing digital circuits, such as integrated circuit design.⁷

STEPPING INTO THE UNKNOWN: ANALOG GATES

Analog computers have a role in computer history,⁸ but computers are overwhelmingly digital today. What happened? A modern (nonquantum) analysis shows that analog computers must overcome fundamental noise limitations to get variables to represent just a handful of bits, which usually ends up being too much effort for the small return.

Computing with qubits is more powerful than computing with bits because qubits have phase and multiple qubits can be entangled. These additional properties permit some algorithms to solve polynomially or exponentially larger problems as a function of the number of qubits or quantum gate operations. Controlling noise may have been too much trouble in the 1950s, but the exponential speedup of a single quantum computer chip today allows it to outrun the world's largest supercomputer,⁹ apparently making the effort worthwhile: not by reducing the effort but by increasing the return.

Now there is a new engineering subdiscipline for understanding how to build scalable quantum analog circuits,⁹ which is in the emerging discipline of quantum engineering. The IEEE Computer Society may be able to shepherd the development of analog counterparts to many of the disciplines such as in CAD, compilers, and theory that currently exist for today's implicitly digital computer science discipline.

CONTROL ELECTRONICS

Today's digital computing seems destined to become the control electronics for qubits, but heat dissipation is starting to limit scaling just as much as the qubits themselves. Control signals can be created at room temperature and transmitted to cryogenic quantum components through cables, but the cables conduct heat and noise from room temperature and increase cooling load. If the control signals are generated in the vicinity of the cryogenic qubits, the heat dissipation of the electronics increases cooling load. The remedy will

be to develop an understanding of classical computer architecture covering multiple temperature ranges, which is an exciting and crucial new area of engineering innovation.^{10,11}

MOON LANDING POINT: TURING-COMPLETE QUANTUM COMPUTERS

Today's computers are very much focused on the Turing machine concept and its ability to compute any computable function. (I'm using the phrase Turing machine where the more precise phrase would be bounded-storage machine.) There is a tremendous body of knowledge on making fast Turing-complete microprocessors, such as superscalar and parallel architectures, along with associated compilers and parallel algorithms. Quantum computers obtain high performance in other ways. While a pure quantum computer can be Turing complete, the quantum part alone does not need to be. In fact, a highly capable quantum accelerator coupled to a classical control system containing an Arduino microprocessor could be Turing complete at the system level.

I predict that the moonshot will land as a natively quantum architecture designed to service a class of applications. It is clear from current quantum computer hardware that quantum computers will excel at problems such as optimizations, physical simulations, perhaps machine learning, and probably others. So the moonshot may yield a Turing-complete quantum computer that can handle any computable function at today's speed but is faster than any classical computer at calculating functions that are reliant on optimization or one of the other quantum accelerator examples.

A natively quantum architecture seems inevitable, but the quest has not been formalized so far. Today's quantum computers are typically built from field programmable gate arrays and lab equipment in an apparently ad hoc process for each quantum computer system. There are publications¹² but not many.

The computer industry has taken past advances in stride, such as Moore's law almost seamlessly shifting from bipolar transistors to CMOS in the 1980s. But the past evolution of computers has been at the lower levels of the design hierarchy. The

computer industry has always been committed to Turing machines and their ability to compute any computable function. Quantum concepts are entering at the lower levels but will ultimately change computing at the high levels as well.

The new type of computer would use different techniques to solve problems. These will obviously be within the space of all computable problems, but the new quantum computer would solve problems intractable for a nonquantum computer or, in other words, a classical computer would take longer than the life of the universe to solve the same problem.

Some of the new problems are readily imagined, although obviously we do not have a lot of practical experience solving problems that have not been solved before. It is easy to find examples, such as learning faster, factoring of thousand-digit numbers, and modeling the behavior of proteins, that may exist in our bodies and but are too large to simulate with classical computers.

The best endpoint in my view would be for the IEEE Computer Society to expand the traditional view of computers to include support for digital bits, analog qubits, and cryogenic operation (where needed for certain types of qubits). These expanded computers would be Turing machines, yet able to solve problems effectively in a larger space that includes the capabilities of both today's microprocessors and quantum computers. 🤖

REFERENCES

1. *IEEE Spectrum*, June 1964. [Online]. Available: <https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=6500551&pnumber=6>
2. Hello Quantum. Accessed on: Jan. 31, 2020. [Online]. Available: <https://helloquantum.mybluemix.net>
3. C. Ferrie and Whurley, *Quantum Computing for Babies*. Naperville, IL: Sourcebooks, 2018.
4. "EPiQC: Developing and communicating the CS in quantum computing," Canonlab. Accessed on: Jan. 31, 2020. [Online]. Available: <https://www.canonlab.org/quantumvideos>
5. E. DeBenedictis, "Powerball and quantum supremacy," *Computer*, vol. 52, no. 10, pp. 110–112, 2019. doi: 10.1109/MC.2019.2930166.
6. D. E. Deutsch, "Quantum computational networks," *Proc. Roy. Soc. London. A. Math. Phys. Sci.*, vol. 425, no. 1868, pp. 73–90, 1989. doi: 10.1098/rspa.1989.0099.
7. B. Lacour, presented at the *Quantum Computing Session 01*, 2019 IEEE Int. Conf. Rebooting Computing (ICRC), San Mateo, CA, Nov. 6–7, 2019, start time: 3:28:08. [Online]. Available: <https://ieeetv.ieee.org/conference-highlights/educational-programs-in-qis-at-ut-austin-brian-la-cour-icrc-san-mateo-2019>
8. W. Karplus, *Analog Simulation: Solution of Field Problems*. New York: McGraw-Hill, 1958. [Online]. Available: <https://onlinebooks.library.upenn.edu/webbin/book/lookupid?key=olbp64909>
9. S. Boixo, "Cross entropy benchmarking & quantum supremacy," presented at 2019 IEEE Int. Conf. Rebooting Computing (ICRC), San Mateo, CA, Nov. 6–7, 2019. [Online]. Available: <http://ieeetv.ieee.org/conference-highlights/cross-entropy-benchmarking-quantum-supremacy-sergio-boixo-icrc-san-mateo-2019>
10. S. J. Pauka et al., A cryogenic interface for controlling many qubits. 2019. [Online]. Available: [arXiv:1912.01299](https://arxiv.org/abs/1912.01299)
11. D. P. Franke, J. S. Clarke, L. M. K. Vandersypen, and M. Veldhorst, "Rent's rule and extensibility in quantum computing," *Microprocess. Microsyst.*, vol. 67, pp. 1–7, June 2019. doi: 10.1016/j.micpro.2019.02.006
12. A. Butko et al., Understanding quantum control processor capabilities and limitations through circuit characterization. 2019. [Online]. Available: <https://arxiv.org/abs/1909.11719>

ERIK DEBENEDICTIS is the principal of Zettaflops, LLC, editor-in-chief of *IEEE Transactions on Quantum Engineering*, and a volunteer supporting the IEEE Computer Society, Council on Superconductivity, Rebooting Computing, and Quantum Initiative. Contact him at erikdebenedictis@gmail.com.



WWW.COMPUTER.ORG/COMPUTINGEDGE

Get Published in the New *IEEE Open Journal of the Computer Society*

Submit a paper today to the premier new open access journal in computing and information technology.

Your research will benefit from the IEEE marketing launch and 5 million unique monthly users of the IEEE *Xplore*® Digital Library. Plus, this journal is fully open and compliant with funder mandates, including Plan S.

Submit your paper today!

Visit www.computer.org/oj to learn more.



The National Quantum Initiative Will Also Benefit Classical Computers

Erik P. DeBenedictis and Michael P. Frank, *Sandia National Laboratories*

The U.S. National Quantum Initiative places quantum computer scaling in the same category as Moore's law. While the technical basis of semiconductor scale up is well known, the equivalent principle for quantum computers is still being developed. Let's explore these new ideas.

A regular, or classical, computer delivers value to the user based on the complexity of the calculation it performs. Classical computers are overwhelmingly based on transistors, and Moore's law showed how to make transistors cheaper, faster, and more energy efficient over time. While quantum computers are expected to be built mostly from classical devices as well, the value delivered to the user will depend on the qubits, not the supporting devices. The point of this article is to demonstrate that shifting classical devices from the primary role to a supporting role will transform perspectives about computers.

Let's introduce this new thinking by considering the central 3D structure of a quantum computer, which contains the qubits and a portion of the control system. While nobody knows which type of qubit will go into large-scale production, most candidate qubits require cryogenic operation. Through-silicon vias (TSVs) are a current example of 3D technology. They are microscopic bumps that electrically and mechanically interconnect up to roughly half a dozen integrated circuit die along their planar faces to a 3D module.

The first advance we're seeing is the extension of these approaches to multiple operating temperatures, such as room-temperature layers plus other layers at one or more cryogenic temperatures. The cryogenic

layers allow a second extension to novel devices and circuits, including sensing devices that require low temperatures to function and computing devices to support them.

LOGIC PERFORMANCE AND TEMPERATURE

There was an extensive search for transistor replacement as a potential remedy to the so-called end of Moore's law, ultimately producing Figure 1's scatterplot of energy and delay for many candidate devices.¹ Focusing just on the top group for now, we see that the three data points with whiskers are Josephson junction-based reciprocal quantum logic (RQL) and adiabatic quantum flux parametron (AQFP) logic families. The dots for these cryogenic devices show the sum of device energy per operation plus the additional energy used by the cryogenic refrigerator to move heat from 4 K to room temperature; the whisker shows the result over a refrigerator efficiency range. None of the devices in the top group stands out.

When low-temperature operation is required, all the devices in Figure 1 need refrigeration, and the method of tallying the energy cost of cryogenic refrigeration changes. In low-temperature operation, devices in both the upper and lower groups operate at temperature T and require a cryogenic refrigerator to remove heat to room temperature. A 100% Carnot-efficient refrigerator will multiply the heat by $300\text{ K}/T$ in the process of moving it to the room temperature (300

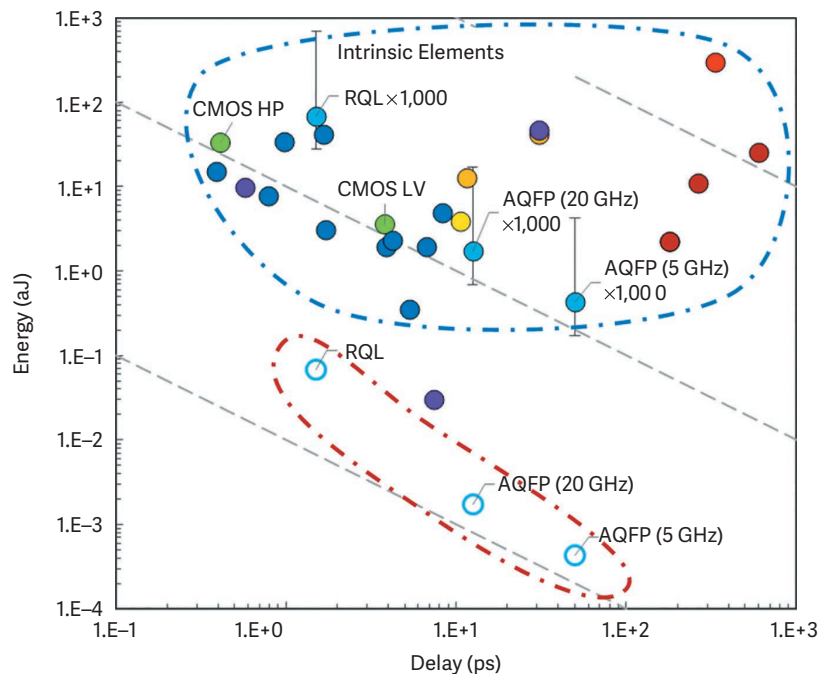


FIGURE 1. An energy-delay plot of a comprehensive set of logic devices at room temperature, including Josephson junctions operating at 4 K in three circuits (RQL and, at two speeds, AQFP). None of the devices stands out at room temperature because only the superconducting ones have refrigeration overhead. However, at 4 K, they all require refrigeration, causing the superconducting devices to stand out. (The outlier purple dot is a BisFET, which is too immature to be considered a serious contender.)

K) environment, and refrigerator inefficiency adds another multiple of 5–20×. However, these factors are the same for all devices, so the best options will be the points closest to the origin of the graph, which are the cryogenic Josephson junction-based devices.

DIFFERENT CIRCUITS BECOME EFFECTIVE AT LOW TEMPERATURES

Everybody wants to save energy, which is why wise engineers consider energy-efficient light bulbs for their homes. However, once they see that the energy-efficient bulb costs ten times as much as an incandescent one, they do a calculation to see if it's a

good deal. If the increase in purchase price is greater than the energy savings over the bulb's lifetime, it's a bad deal. Energy-savings technology for computers, such as reversible and adiabatic logic, have been known for years. However, they trade energy efficiency during operation for more complex circuits that cost more to buy in the first place—leading to the same tradeoff as for light bulbs.

All other things being equal, you become more likely to choose the energy-efficient technology as the price of energy rises. Energy-efficient lights and reversible computers that are bad deals at room temperature become better deals at 4 K because of the effective 1,000× increase in energy cost—and are

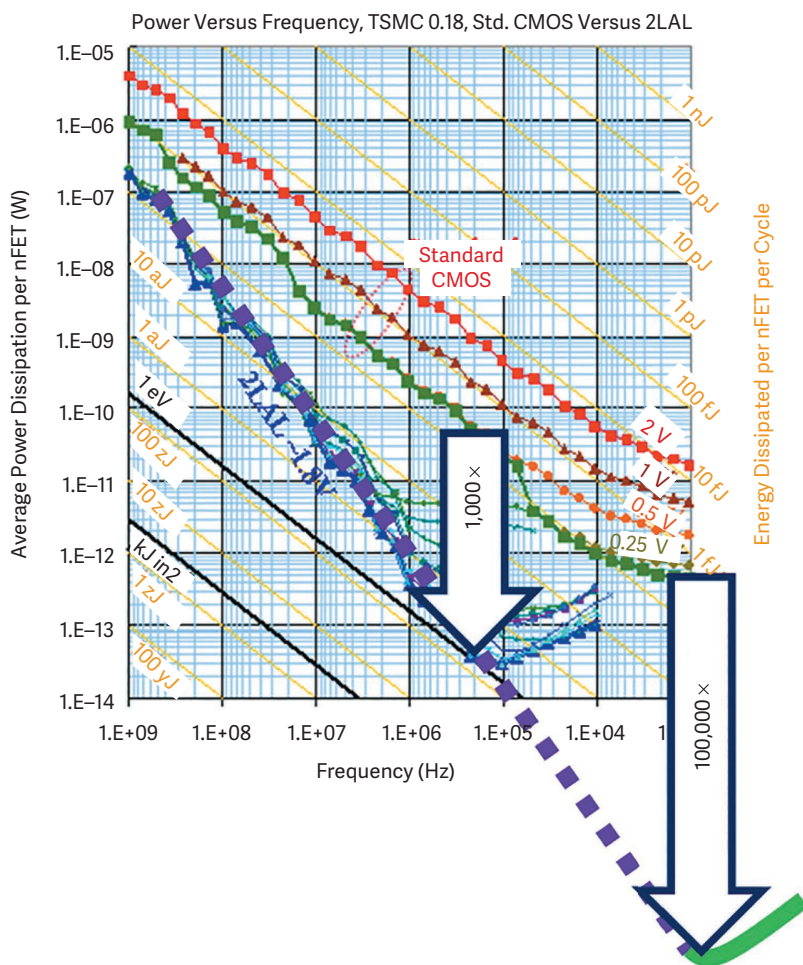


FIGURE 2. A comparison of circuit efficiency for standard CMOS (top line) and a reversible circuit 2LAL, showing a maximum advantage of 1,000× at 200 KHz. However, if the 2LAL is operated at 4 K, downward-sloping curves should extend further, leading to a possible 100,000× energy-efficiency improvement over room-temperature electronics. This may allow a transistorized 2LAL to compete with Josephson junctions in applications where speed is not essential. FET: field-effect transistor; freq.: frequency; TSMC: Taiwan Semiconductor Manufacturing Company, Limited.

no-brainers for systems operating at millikelvins, such as quantum computers, where the effective cost rises 1 million times.

Adiabatic and reversible logic have been studied at room temperature for many years, and the plot in Figure 2 shows an example of the benefits when implemented with standard CMOS. The curves are the result of many simulations and show transistor power declining with clock period for both standard CMOS and a reversible logic family called 2LAL.² On the log-log scale, the power dissipation of CMOS declines with

slope -1 (linear), while 2LAL has slope -2 (quadratic). This creates a widening advantage for 2LAL at lower clock frequencies, which eventually ends because of leakage current I_{off} . However, a 200-kHz microprocessor is not very useful, even if it is energy efficient.

What happens to reversible circuits at cryogenic temperatures? Studies of standard CMOS transistors operated at 4 K show a large reduction in leakage current,³ while other factors stay about the same. Although cryogenic 2LAL has not been systematically analyzed for this article, the qualitative result annotated on top of Figure 2's base graph is an extension of the region with downward slope -2 by an additional factor of perhaps 10,000× before leakage current I_{off} causes the upward sloping behavior. A review of the literature indicates that the advantage should increase a bit more below 4 K and then level off, but further work is indicated.

The exponential growth of Moore's law made semiconductors seem unbeatable for decades. However, recent changes in Moore's law have enabled superconductors to pull ahead for quantum computers. But wait! Now that we take time to think, semiconductors

can play this new game, too, by switching from CMOS circuits to 2LAL or some other adiabatic form. If thinking hard can rejuvenate both semiconductors and superconductors, what could happen if we thought about using both at once?

HETEROGENEOUS INTEGRATION OF SEMICONDUCTORS AND SUPERCONDUCTORS

One of us saw a presentation at a recent conference reporting on the invention of a superconducting

field-effect transistor.⁴ It was kind of an odd device. The application of 90 V would switch it from a 0- Ω superconductor to a resistor of 50 Ω or so. Nobody had any idea what to do with it, but we'll suggest that it could interface between 2LAL and Josephson junction logic in a semiconductor–superconductor hybrid module.

Although the device was experimentally tested with 90-V swings, the authors claim that scaling could reduce the drive voltage to 2.5 V, a typical CMOS voltage swing. This could lead to a structure like the one shown in Figure 3, where CMOS voltage signals are converted to Josephson junction current signals.

The inset in Figure 3 shows a superconducting field-effect transistor (FET).⁴ Ignoring the green structures for the moment, the blue structure is a superconducting wire that will conduct current horizontally with zero resistance. However, a narrow superconducting wire conducts with zero resistance only up to a maximum current, called the *critical current*, above which the device becomes a resistor. A narrow wire of this type is called a *weak link* and is a type of a Josephson junction.

However, the critical current declines in the presence of an electric field, such as the field applied by the green capacitor plate in Figure 3 and the corresponding green structure in the inset. Because superconductor circuits are built from wires with zero resistance, voltages in a superconductor circuit almost always stay between ground and a few millivolts. Theory and experiment for the superconducting FET show that the weak link's critical current changes when the green structure applies few volts or more, positive or negative, which is straightforward for transistorized electronics. Hence, the semiconductor layer would communicate with the superconducting layer through voltages that change critical currents.

Attempts to restore traditional computer performance growth rates, like Moore's law, have taken

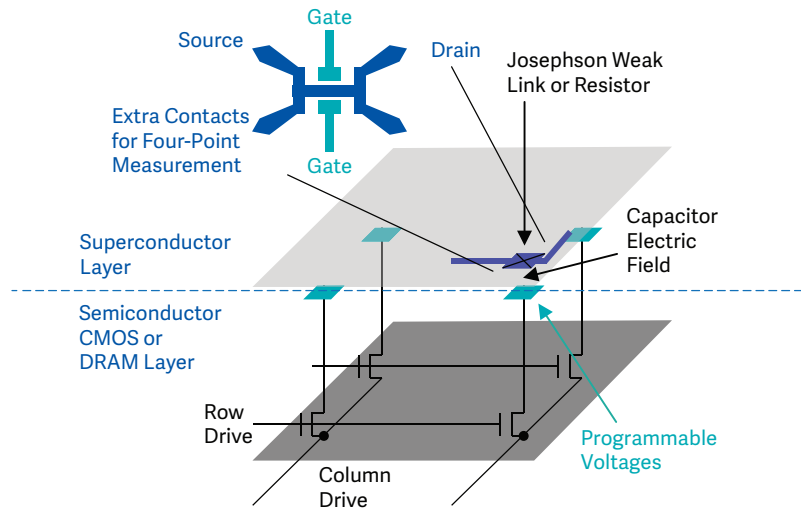


FIGURE 3. A possible semiconductor-superconductor hybrid. The semiconductor layer applies voltage-based signals to the gates of superconducting FETs, which translate the signals into a form readily used by superconducting circuits. The hybrid would be fabricated using a CMOS wafer as a base for depositing superconductor circuits, in lieu of today's method of using a blank silicon wafer as a base. DRAM: dynamic random-access memory.

different directions recently. We've seen how several of these directions can be pursued at once to make a more audacious hybrid than has been seen before.

Running both semiconductors and superconductors at 4 K, with suitable circuits and an efficient, integrated interface between the two, yields a set of building blocks with a unique set of properties:

- › **Semiconductors:** A large number of components are available per unit area, and the technology has achieved high maturity. Signals have 1- to 2-V swings. However, achieving high-energy efficiency requires progressively reducing the clock rate.
- › **Superconductors:** Clock rate remains fast down to the lowest temperatures. These relatively immature devices have large features that would result in low device density. Signal voltages are around a millivolt, but there is a method for conversion from 1- to 2-V signals from semiconductors.

Is there any circuit in common use that could benefit from such an oddball combination of features? Our

first thought is a superconducting field-programmable gate array (FPGA),⁵ but with a new design where the configuration logic is implemented with semiconductors. Because it runs on only power-up, it doesn't have to be fast. After the FPGA has been configured, the Josephson junction logic executes the user-defined logic at high speed.

The standard process for computer design is to create a Turing-complete processor out of universal gates and then add I/O as an afterthought. However, a quantum computer will have two types of universal gates: classical and quantum. Standard computer design would lead to a quantum computer system composed of a quantum processor connected to a classical control system via I/O. But this is not to be. Qubits offer their exotic features only in specific environments, such as cryogenic temperatures, whereas classical electronics offer both density and performance only at room temperature. This article showed how the requirements of each intrude on the design space of the other, destroying the standard abstraction.

So "universal" Boolean logic gates actually have additional properties attached to them (i.e., density and performance at room temperature). The impact of these additional properties was invisible when we only had one example of them. However, mandates to scale up quantum computers force us to deal with quantum gates where the additional properties are incompatible. We must now learn to design with "generalized universal gates," after which we need to revise the standard process for computer design. While quantum computing has the lion's share of public attention right now, the same technology could apply to cryogenic sensor systems that rely on quantum information and which may enable scientific breakthroughs. 🤖

ACKNOWLEDGMENTS

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. The views expressed in the article do not necessarily

represent the views of the U.S. Department of Energy or the United States Government.

REFERENCES

1. D. E. Nikonov and I. A. Young, "Overview of beyond-CMOS devices and a uniform methodology for their benchmarking," *Proc. IEEE*, vol. 101, no. 12, pp. 2498–2533, 2013. doi:10.1109/JPROC.2013.2252317.
2. V. Anantharam, M. He, K. Natarajan, H. Xie, and M. P. Frank. "Driving fully-adiabatic logic circuits using custom high-Q MEMS resonators," in *Proc. Int. Conf. Embedded Systems and Applications and Proc. Int. Conf VLSI (ESA/VLSI)*. Las Vegas, NV, 2004, pp. 5–11.
3. R. M. Incandela, L. Song, H. Homulle, E. Charbon, A. Vladimirescu, and F. Sebastiano. "Characterization and compact modeling of nanometer CMOS transistors at deep-cryogenic temperatures," *IEEE J. Electron Devices Soc.*, vol. 6, pp. 996–1006, Apr. 2018.
4. G. De Simoni, F. Paolucci, P. Solinas, E. Strambini, and F. Giazotto. "Metallic supercurrent field-effect transistor," *Nature Nanotechnology*, vol. 13, no. 9, pp. 802–805, 2018. doi:10.1038/s41565-018-0190-3.
5. N. K. Katam, O. A. Mukhanov, and M. Pedram, "Superconducting magnetic field programmable gate array," *IEEE Trans. Appl. Supercond.*, vol. 28, no. 2, pp. 1–12, 2018. doi:10.1109/TASC.2018.2797262.

ERIK P. DEBENEDICTIS is a technical staff member at Sandia National Laboratories Center for Computing Research. He is a Senior Member of the IEEE and a member of ACM and APS. Contact him at epdeben@sandia.gov.

MICHAEL P. FRANK is a senior technical staff member at Sandia National Laboratories Center for Computing Research. He is a Member of the IEEE. Contact him at mpfrank@sandia.gov.



Evolving Career Opportunities Need Your Skills

Explore new options—upload your resume today

Changes in the marketplace shift demands for vital skills and talent. The **IEEE Computer Society Jobs Board** is a valuable resource tool to keep job seekers up to date on the dynamic career opportunities offered by employers.

Take advantage of these special resources for job seekers:



JOB ALERTS



TEMPLATES



WEBINARS



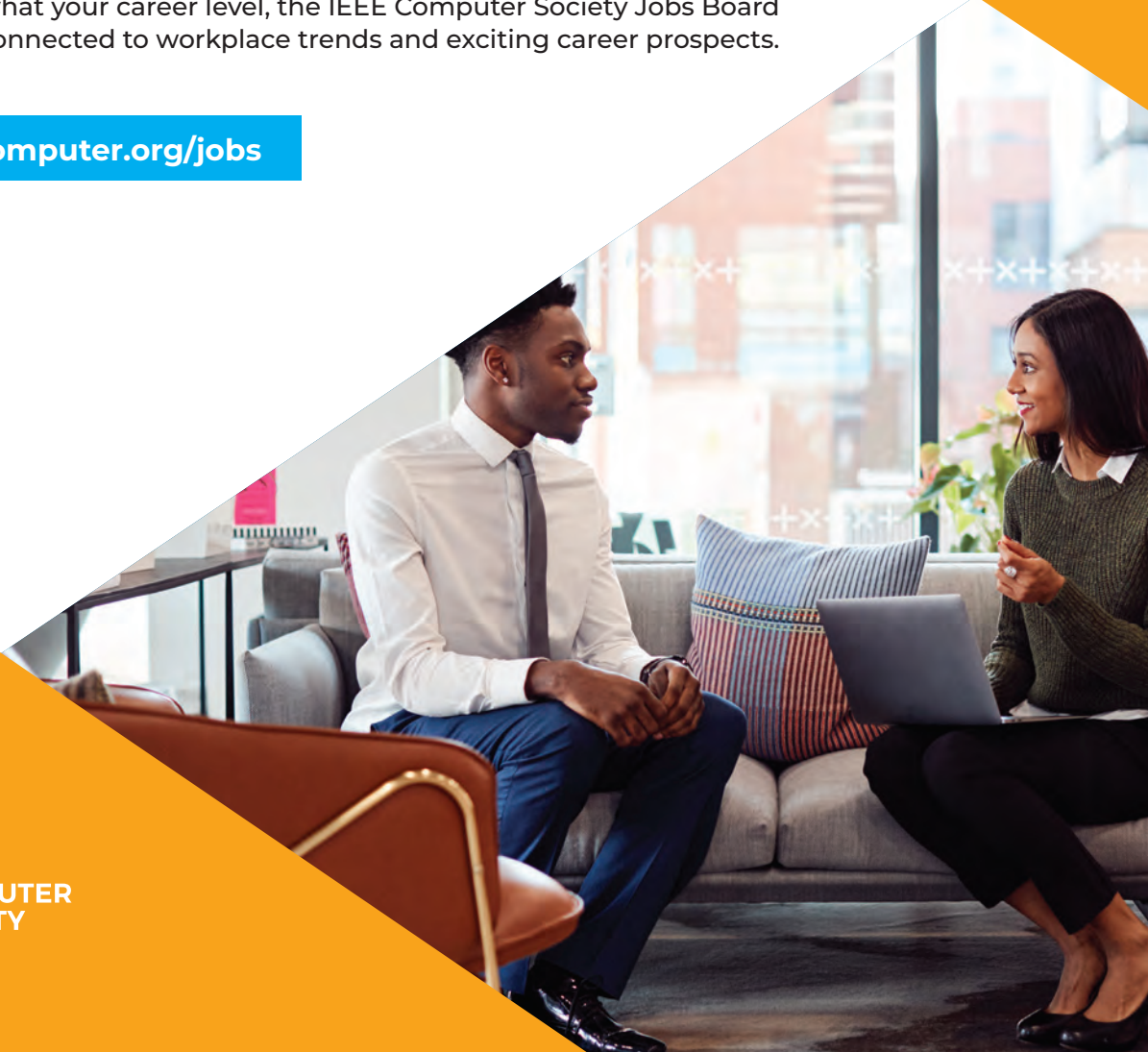
CAREER
ADVICE



RESUMES VIEWED
BY TOP EMPLOYERS

No matter what your career level, the IEEE Computer Society Jobs Board keeps you connected to workplace trends and exciting career prospects.

www.computer.org/jobs



IEEE TRANSACTIONS ON

COMPUTERS

Call for Papers: IEEE Transactions on Computers

Publish your work in the IEEE Computer Society's flagship journal, *IEEE Transactions on Computers (TC)*. *TC* is a monthly publication with a wide distribution to researchers, industry professionals, and educators in the computing field.

TC seeks original research contributions on areas of current computing interest, including the following topics:

- Computer architecture
- Software systems
- Mobile and embedded systems
- Security and reliability
- Machine learning
- Quantum computing

All accepted manuscripts are automatically considered for the monthly featured paper and annual Best Paper Award.

Learn about calls for papers and submission details at
www.computer.org/tc.



IEEE
COMPUTER
SOCIETY



IEEE Internet Computing

IEEE Internet Computing delivers novel content from academic and industry experts on the latest developments and key trends in Internet technologies and applications.

Written by and for both users and developers, the bimonthly magazine covers a wide range of topics, including:

- Applications
- Architectures
- Big data analytics
- Cloud and edge computing
- Information management
- Middleware
- Security and privacy
- Standards
- And much more

In addition to peer-reviewed articles, *IEEE Internet Computing* features industry reports, surveys, tutorials, columns, and news.

www.computer.org/internet

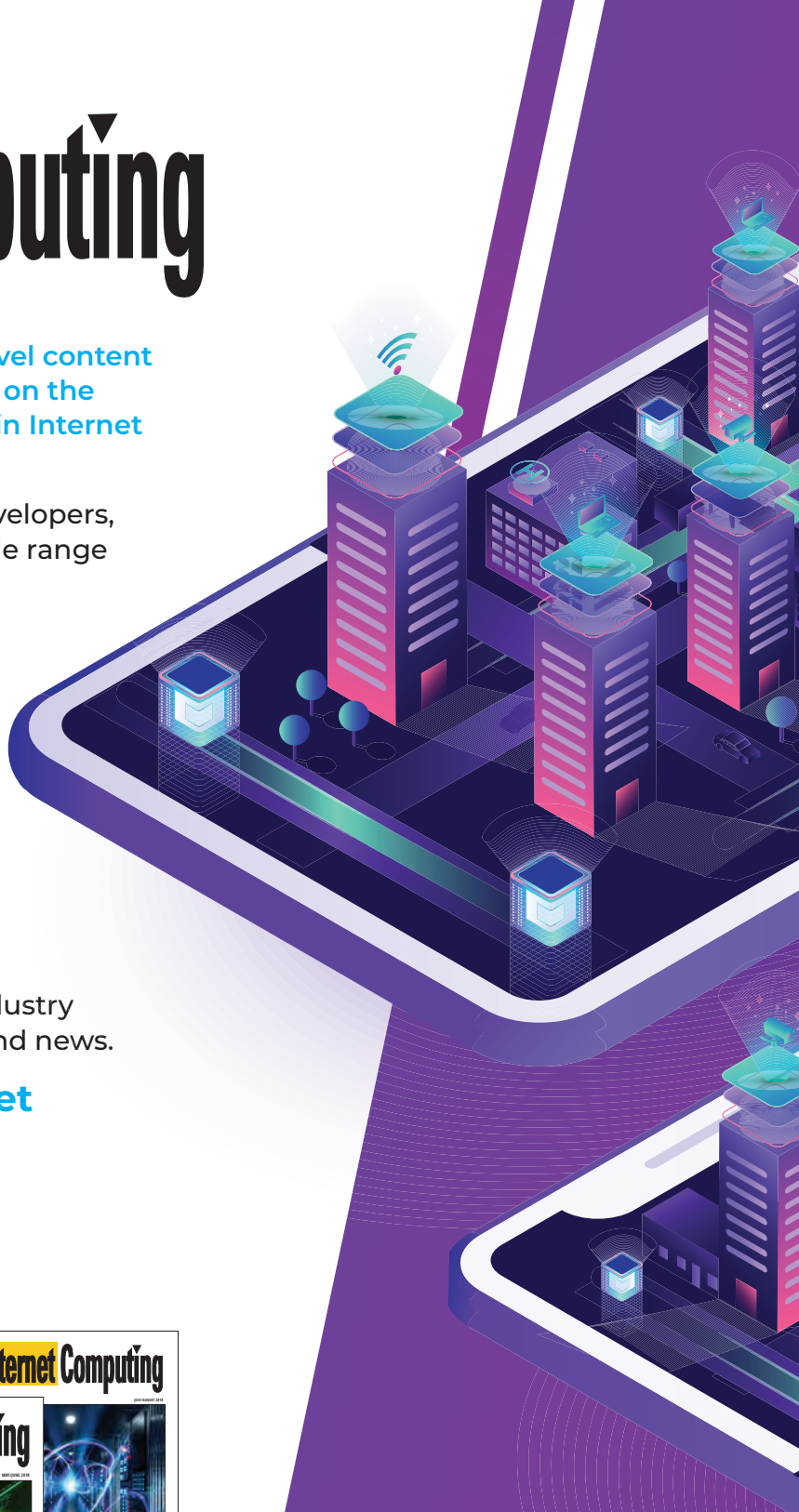


Join the IEEE Computer Society
for subscription discounts today!

www.computer.org/product/magazines/internet-computing



IEEE
COMPUTER
SOCIETY



IEEE

COMPUTER ARCHITECTURE

LETTERS

IEEE Computer Architecture Letters is a forum for fast publication of new, high-quality ideas in the form of short, critically refereed technical papers. Submissions are accepted on a continuing basis and letters will be published shortly after acceptance in IEEE Xplore and in the Computer Society Digital Library.

Submissions are welcomed on any topic in computer architecture, especially:

- Microprocessor and multiprocessor systems
- Microarchitecture and ILP processors
- Workload characterization
- Performance evaluation and simulation techniques
- Interactions with compilers and operating systems
- Interconnection network architectures
- Memory and cache systems
- Power and thermal issues at the architectural level
- I/O architectures and techniques
- Independent validation of previously published results
- Analysis of unsuccessful techniques
- Domain-specific processor architecture (embedded, graphics, network)
- High-availability architectures
- Reconfigurable computer architectures

www.computer.org/cal

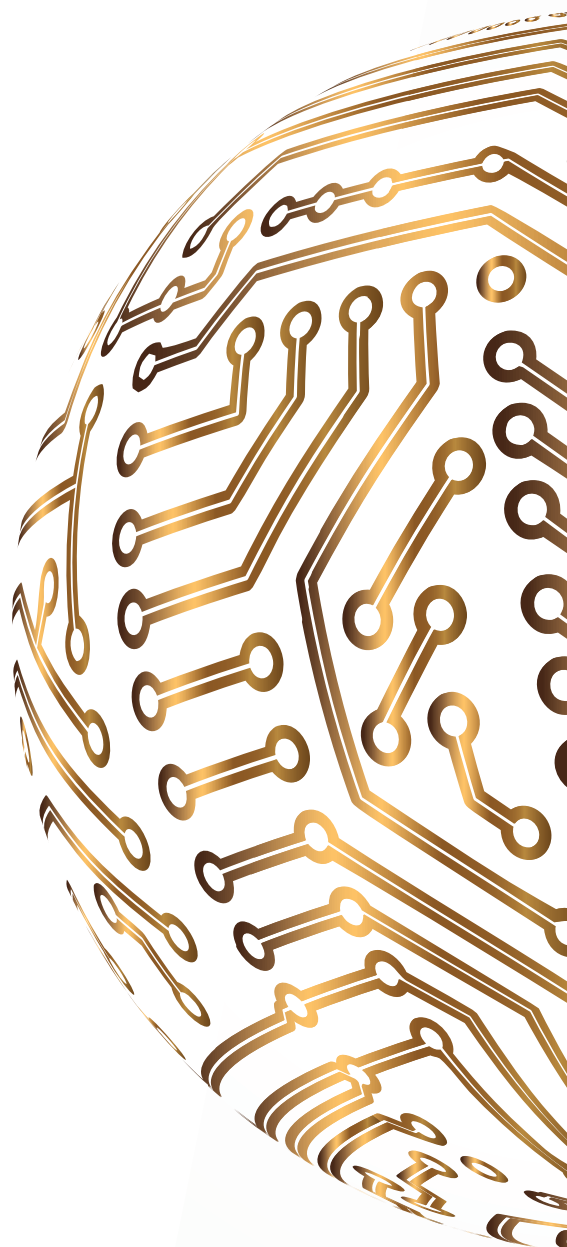


Join the IEEE Computer Society
for subscription discounts today!

www.computer.org/product/journals/cal



IEEE
COMPUTER
SOCIETY



IEEE

SECURITY & PRIVACY

IEEE Security & Privacy is a bimonthly magazine communicating advances in security, privacy, and dependability in a way that is useful to a broad section of the professional community.

The magazine provides articles with both a practical and research bent by the top thinkers in the field of security and privacy, along with case studies, surveys, tutorials, columns, and in-depth interviews. Topics include:

- Internet, software, hardware, and systems security
- Legal and ethical issues and privacy concerns
- Privacy-enhancing technologies
- Data analytics for security and privacy
- Usable security
- Integrated security design methods
- Security of critical infrastructures
- Pedagogical and curricular issues in security education
- Security issues in wireless and mobile networks
- Real-world cryptography
- Emerging technologies, operational resilience, and edge computing
- Cybercrime and forensics, and much more

www.computer.org/security

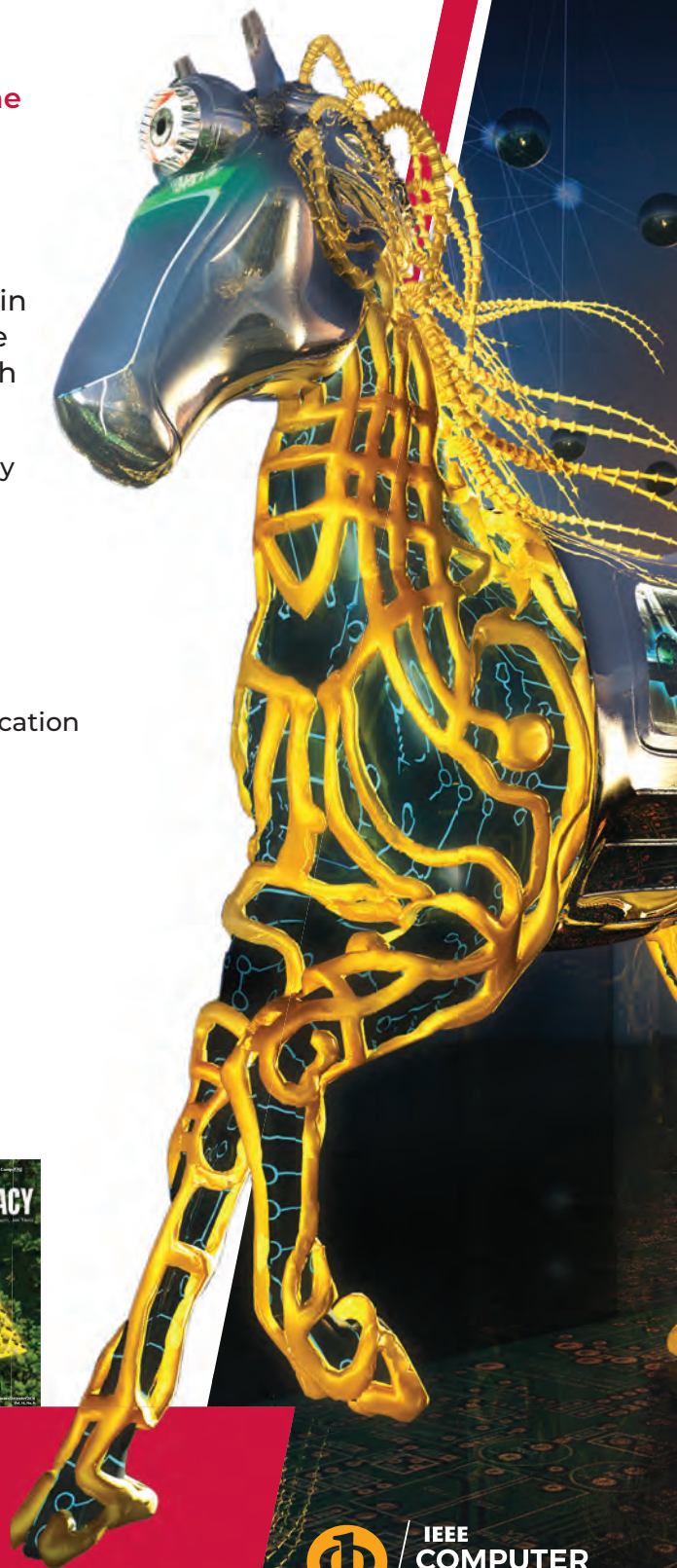


Join the IEEE Computer Society
for subscription discounts today!

www.computer.org/product/magazines/security-and-privacy



IEEE
COMPUTER
SOCIETY

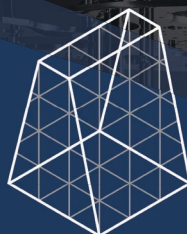


SHAPING THE FUTURE OF QUANTUM COMPUTING

With a history of delivering technology solutions, Honeywell has established itself as a leader in quantum computing

Learn more at: www.Honeywell.com/QuantumSolutions
Contact us at: QuantumSolutions@Honeywell.com

Honeywell | THE FUTURE IS WHAT WE MAKE IT



Azure Quantum

Experience quantum
impact today

Azure.com/quantum

Only Quantum Can Take On Quantum.

Most approaches to quantum ready cybersecurity fall down outside a laboratory environment. They require new dark fibre infrastructure, are too expensive, or unscalable—or otherwise unfit for the real world.

Meanwhile, Quantropi's unique, patented **QEEP™** quantum gate technology is not just enterprise grade and application ready. It works right here, right now, on ANY existing network. Even the WIRELESS Internet.

Industry leading performance.
Plug and play deployment.
Unbreakable confidence.

Today, tomorrow, forever.



*QEEP-KD 3500 application layer
solution for quantum key generation,
management & distribution.*

Bring it on.

quantropi®

quantropi.com
info@quantropi.com

**Join Zapata at the
forefront of quantum**



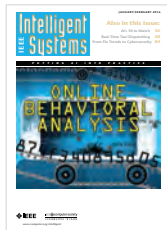
We are looking for creative, curious, and revolutionary mindsets. Is that you?

Then join our world-class team of resourceful problem solvers who are pioneering commercial quantum algorithms, product development and research. We are committed to creating an environment that accelerates your growth and positive impact in the world.

We are growing rapidly across our various teams. Search open jobs and fill out a profile with your resume at zapatacomputing.com/careers so we can contact you for future openings.

ZAPATACOMPUTING.COM/CAREERS

stay on the Cutting Edge of Artificial Intelligence



IEEE Intelligent Systems provides peer-reviewed, cutting-edge articles on the theory and applications of systems that perceive, reason, learn, and act intelligently.

The #1 AI Magazine
www.computer.org/intelligent

IEEE
Intelligent Systems

Let's build a quantum future together.

When we launched the first quantum computer accessible on the cloud in 2016, we were astonished when we reached 8,000 users.

Today, we've reached over 250,000 users. This is the world's largest and most engaged quantum developer community running over 1 billion circuits daily on a fleet of the world's most powerful quantum computers, all on the IBM Cloud.

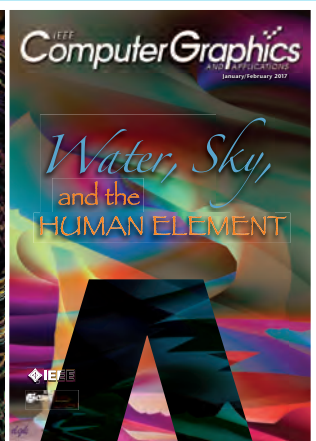
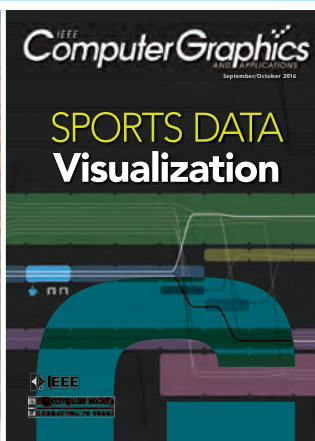
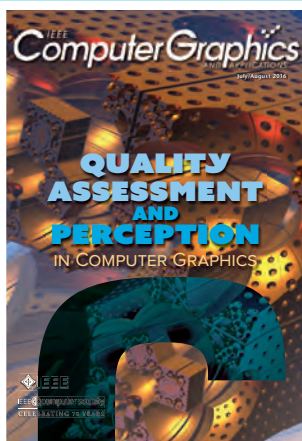
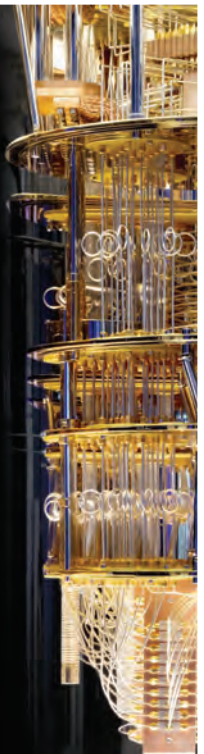
Our next goal is to reach the world's 25 million developers, to empower them to put quantum to use in solving today's intractable challenges.

Our strategy is to make quantum computing as seamless and as easy to use as today's classical programming frameworks.

If you'd like to learn more, please contact hmccortez@us.ibm.com

IBM Quantum

IBM



CG&A

www.computer.org/cga

IEEE Computer Graphics and Applications bridges the theory and practice of computer graphics. Subscribe to *CG&A* and

- stay current on the latest tools and applications and gain invaluable practical and research knowledge,
- discover cutting-edge applications and learn more about the latest techniques, and
- benefit from *CG&A*'s active and connected editorial board.

CVPR

VIRTUAL JUNE 14-19

Thank You to Our Sponsors!

CVPR 2020—the premier annual conference on computer vision and pattern recognition—virtually delivered more than 5,000 first-class papers, keynotes, sessions, workshops, and tutorials to an audience of 7,600 from all over the world!

We are so grateful to our wonderful volunteers and our amazing lineup of sponsors for their continuing support.

Champion



Supporter



cvpr20.com

**SUBMIT
TODAY**

IEEE TRANSACTIONS ON BIG DATA

► SCOPE

The *IEEE Transactions on Big Data (TBD)* publishes peer reviewed articles with big data as the main focus. The articles provide cross disciplinary innovative research ideas and applications results for big data including novel theory, algorithms and applications. Research areas for big data include, but are not restricted to, big data analytics, big data visualization, big data curation and management, big data semantics, big data infrastructure, big data standards, big data performance analyses, intelligence from big data, scientific discovery from big data security, privacy, and legal issues specific to big data. Applications of big data in the fields of endeavor where massive data is generated are of particular interest.

SUBSCRIBE AND SUBMIT

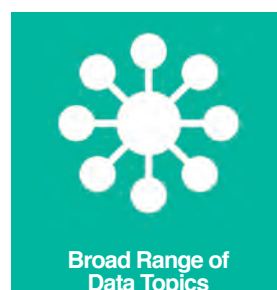
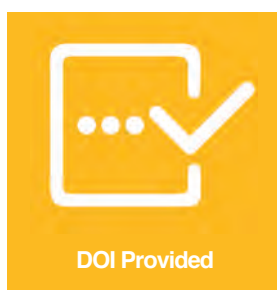
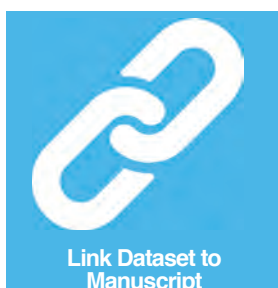
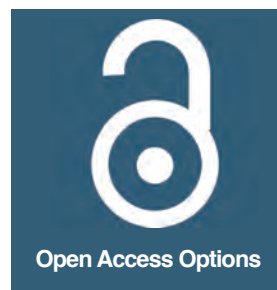
For more information on paper submission, featured articles, calls for papers, and subscription links visit:

www.computer.org/tbd



SHARE AND MANAGE YOUR RESEARCH DATA

IEEE DataPort is an accessible online platform that enables researchers to easily share, access, and manage datasets in one trusted location. The platform accepts all types of datasets, up to 2TB, and dataset uploads are currently free of charge.



IEEE*DataPort*[™]

UPLOAD DATASETS AT IEEE-DATAPORT.ORG



Keep up with the latest IEEE Computer Society publications and activities wherever you are.

Follow us:



| @ComputerSociety



| facebook.com/IEEEComputerSociety



| IEEE Computer Society



| youtube.com/ieeecomputersociety



| instagram.com/ieee_computer_society





Conference Calendar

IEEE Computer Society conferences are valuable forums for learning on broad and dynamically shifting topics from within the computing profession. With over 200 conferences featuring leading experts and thought leaders, we have an event that is right for you. Questions? Contact conferences@computer.org.

OCTOBER

3 October

- PACT (Int'l Conf. on Parallel Architectures and Compilation Techniques), virtual

5 October

- EDOC (IEEE Int'l Enterprise Distributed Object Computing Conf.), virtual

12 October

- ISSRE (IEEE Int'l Symposium on Software Reliability Eng.), virtual
- QCE (IEEE Int'l Conf. on Quantum Computing and Eng.), virtual

17 October

- MICRO (IEEE/ACM Int'l Symposium on Microarchitecture), virtual

18 October

- ICCD (IEEE Int'l Conf. on Computer Design), virtual
- MODELS (ACM/IEEE Int'l Conf. on Model-Driven Eng. Languages and Systems), virtual

19 October

- DFT (IEEE Int'l Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems), virtual
- EDGE (IEEE Int'l Conf. on Edge

Computing), virtual

- SERVICES (IEEE World Congress on Services), virtual

21 October

- FIE (IEEE Frontiers in Education Conf.), virtual

25 October

- LDAV (IEEE Symposium on Large Data Analysis and Visualization), virtual
- VIS (IEEE Visualization Conf.), virtual

NOVEMBER

6 November

- CCEM (IEEE Int'l Conf. on Cloud Computing in Emerging Markets), virtual
- SmartCloud (IEEE Int'l Conf. on Smart Cloud), virtual

9 November

- ICTAI (IEEE Int'l Conf. on Tools with Artificial Intelligence), virtual
- IRC (IEEE Int'l Conf. on Robotic Computing), virtual
- ISMVL (IEEE Int'l Symposium on Multiple-Valued Logic), virtual

11 November

- SEC (IEEE/ACM Symposium on Edge Computing), San Jose, USA

15 November

- SC, virtual

16 November

- FOCS (IEEE Symposium on Foundations of Computer Science), Durham, USA
- LCN (IEEE Conf. on Local Computer Networks), virtual

29 November

- ICDCS (IEEE Int'l Conf. on Distributed Computing Systems), Singapore

30 November

- ICHI (IEEE Int'l Conf. on Healthcare Informatics), Oldenburg, Germany

DECEMBER

2 December

- CSDE (IEEE Asia-Pacific Conf. on Computer Science and Data Eng.), Gold Coast, Australia
- ISM (IEEE Int'l Symposium on Multimedia), Naples, Italy

6 December

- HOST (IEEE Int'l Symposium on Hardware-Oriented Security and Trust), virtual

7 December

- BDCAT (IEEE/ACM Int'l Conf. on Big Data Computing, Applications and Technologies), virtual



- UCC (IEEE/ACM Int'l Conf. on Utility and Cloud Computing), virtual

9 December

- CC (IEEE Int'l Conf. on Conversational Computing), virtual
- AIKE (IEEE Int'l Conf. on Artificial Intelligence and Knowledge Eng.), virtual

10 December

- BigData (IEEE Int'l Conf. on Big Data), virtual

14 December

- CloudCom (IEEE Int'l Conf. on Cloud Computing Technology and Science), Bangkok, Thailand
- HPCCom (IEEE Int'l Conf. on High Performance Computing and Communications), Cuvu, Fiji

16 December

- BIBM (IEEE Int'l Conf. on Bioinformatics and Biomedicine), virtual
- HiPC (IEEE Int'l Conf. on High Performance Computing, Data, and Analytics), virtual

29 December

- BigDataSE (IEEE Int'l Conf. on Big Data Science and Eng.), Guangzhou, China
- EUC (IEEE Int'l Conf. on Embedded and Ubiquitous Computing), Guangzhou, China
- TrustCom (IEEE Int'l Conf. on Trust, Security and Privacy in Computing and Communications), Guangzhou, China

2021

JANUARY

5 January

- WACV (IEEE Winter Conf. on Applications of Computer Vision), Waikoloa, USA

10 January

- ICPR (Int'l Conf. on Pattern Recognition), Milan, Italy

17 January

- BigComp (IEEE Int'l Conf. on Big Data and Smart Computing), Bangkok, Thailand

27 January

- ICSC (IEEE Int'l Conf. on Semantic Computing), Laguna Hills, USA

MARCH

22 March

- PerCom (IEEE Int'l Conf. on Pervasive Computing and Communications), Kassel, Germany
- MIPR (IEEE Int'l Conf. on Multimedia Information Processing and Retrieval), Tokyo, Japan

APRIL

12 April

- ICST (IEEE Conf. on Software Testing, Verification and Validation), Porto de Galinhas, Brazil

19 April

- ICDE (IEEE Int'l Conf. on Data Engineering), Chania, Greece

21 April

- SELSE (IEEE Workshop on Silicon Errors in Logic - System Effects), Los Angeles, USA

MAY

3 May

- NAS (IEEE Int'l Conf. on Networking, Architecture and Storage), Riverside, USA

10 May

- CCGrid (IEEE/ACM Int'l Symposium on Cluster, Cloud and Internet Computing), Melbourne, Australia

17 May

- IPDPS (IEEE Int'l Parallel and Distributed Processing Symposium), Portland, Oregon, USA

23 May

- SP (IEEE Symposium on Security and Privacy), San Francisco, USA



Learn more
about IEEE
Computer
Society
conferences

computer.org/conferences

NEW EVENT

IEEE QUANTUM WEEK

12-16 OCTOBER 2020

REGISTRATION IS OPEN!
qce.quantum.ieee.org

The Future Directions Quantum Initiative invites you to IEEE Quantum Week 2020—the inaugural IEEE International Conference on Quantum Computing and Engineering (QCE).

IEEE
QUANTUM

IEEE
COMPUTER
SOCIETY

IEEE
ComSoc
IEEE Communications Society

IEEE CSC
Council on Superconductivity

IEEE
Photonics
Society

IEEE
ELECTRONICS
PACKAGING
SOCIETY

IEEE TEMS
Technology & Engineering
Management Society

IEEE