COMPUTING

edge

BIG DATA ANALYTICS

Also in this issue:

- **> Interview with Award-Winning *New York Times* Journalist on Cybersecurity**
- **> Should Architects Code?**

MARCH 2018
www.computer.org

IEEE
IEEE computer society

IEEE COMPUTER SOCIETY: Be at the Center of It All

IEEE Computer Society membership puts you at the heart of the technology profession—and helps you grow with it.

Here are 10 reasons why you need to belong.

1. **Training** that sharpens your edge in Cisco, IT security, MS Enterprise, Oracle and more.
2. **Certifications** and exam preparation that set you apart.
3. **Industry intelligence,** including *Computer,* myCS, Computing Now, and myComputer.
4. Over 200 annual **conferences** and technical events held worldwide.
5. 300-plus chapters and more than 30 technical committees **keep you connected.**
6. **Access** to hundreds of books, 13 technical magazines and 20 research journals.
7. **Deep discounts** on magazines, journals, conferences, symposia and workshops.
8. Opportunities to **get involved** through speaking, publishing and volunteering opportunities.
9. **Scholarships awarded** to computer science and engineering student members each year.
10. **A robust jobs board,** plus videos, articles and presentations to help you land that next opportunity.

IEEE Computer Society—keeping you ahead of the game. Get involved today.

www.computer.org/membership

IEEE computer society

COMPUTING edge

IEEE COMPUTER SOCIETY computer.org • +1 714 821 8380

Printed with inks containing soy and/or vegetable oils

IEEE

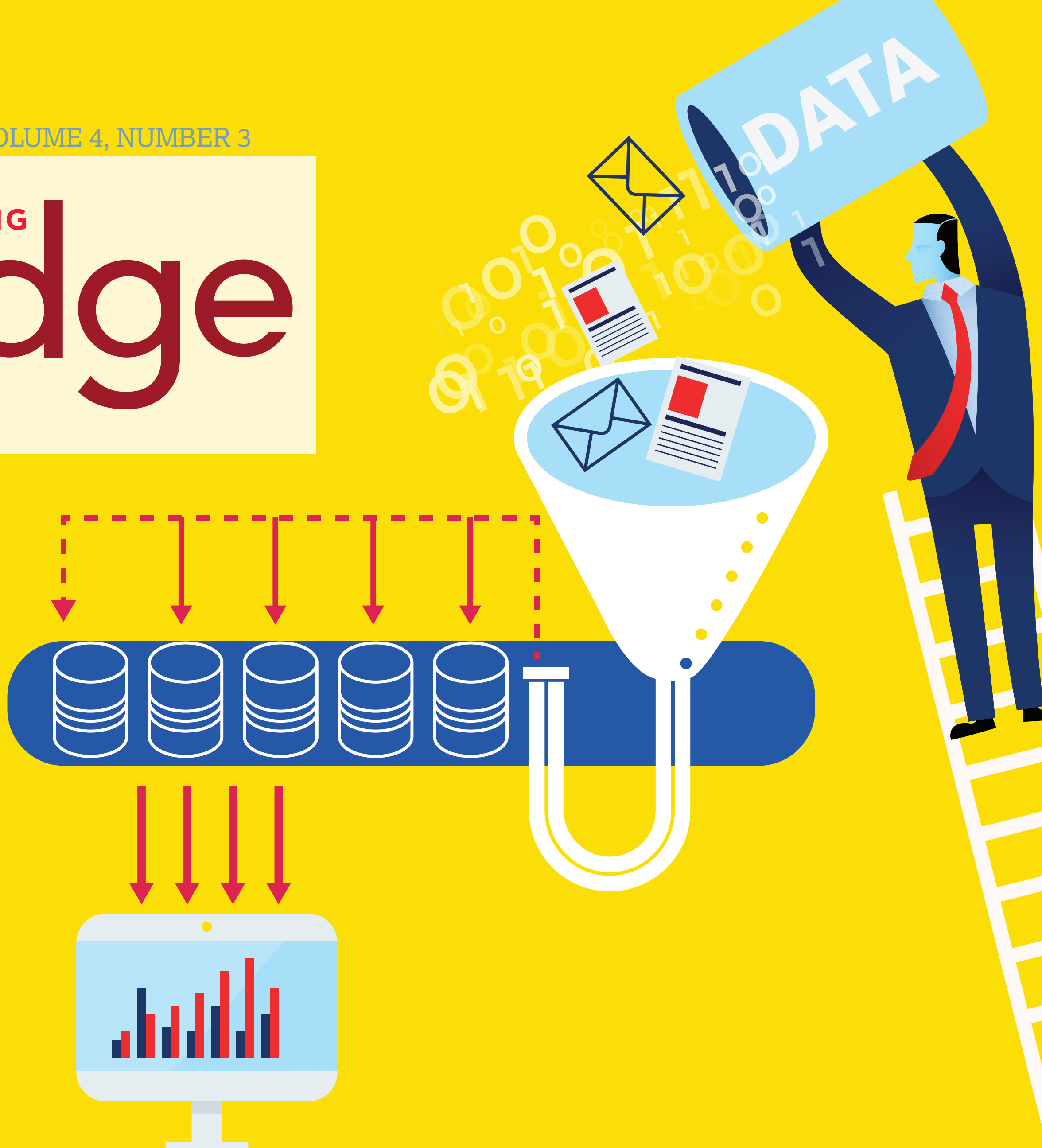
IEEE computer society

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.

MARCH 2018 • VOLUME 4, NUMBER 3

COMPUTING edge



12
The Problem of Personalization: AI-Driven Analytics at Scale

19
Big, Linked Geospatial Data and Its Applications in Earth Observation

38
In Defense of MapReduce

8 Editor's Note: Big Data Analytics

10 The Future of Data Management
DAVID B. LOMET

12 The Problem of Personalization: AI-Driven Analytics at Scale
SETH EARLEY

19 Big, Linked Geospatial Data and Its Applications in Earth Observation
MANOLIS KOUBARAKIS, KONSTANTINA BERETA, GEORGE PAPADAKIS, DIMITRIANOS SAVVA, AND GEORGE STAMOULIS

24 Orchestrating BigData Analysis Workflows
RAJIV RANJAN, SAURABH GARG, ALI REZA KHOSKBAR, ELLIS SOLAIMAN, PHILIP JAMES, AND DIMITRIOS GEORGAKOPOULOS

33 Population-Scale Pervasive Health
TIM ALTHOFF

38 In Defense of MapReduce
JIMMY LIN

44 Big Data and Big Money: The Role of Data in the Financial Sector
JENNIFER Q. TRELEWICZ

47 Silver Bullet Talks with Nicole Perlroth
GARY MCGRAW

50 Should Architects Code?
EOIN WOODS

Departments

4 Magazine Roundup

44
Big Data and Big Money: The Role of Data in the Financial Sector

Magazine Roundup

by Lori Cameron

The IEEE Computer Society's lineup of 13 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 cloud migration and microchip design. Here are highlights from recent issues.

Computer

Disinformatics: The Discipline behind Grand Deceptions

In light of the misinformation proliferated on social media, Out of Band columnist Hal Berghel believes a new discipline should be created in academia to further study and analyze deception—a discipline he has termed "disinformatics." Disinformatics reveals itself at the intersection of technology, propaganda, and miscreants. It's the glue that holds together modern faux news outlets, AM talk radio, Twitterstorms, and sundry other sorts of sociopolitical babble. "It's ideologically grounded in postmodern logic and epistemology (for example, truth is what makes the public strong in body and spirit) and rests upon a foundation of informal logical fallacies and falsehoods," claims Berghel. Read more in the January 2018 issue of *Computer*.

Computing in Science & Engineering

CoFlaVis: A Visualization System for Pulverized Coal Flames

One of the problems with which researchers of different domains, such as chemistry and fluid dynamics, are concerned is the optimization of coal combustion processes to increase the efficiency, safety, and cleanliness of such systems. The coal

Published by the IEEE Computer Society

combustion process is reproduced by using complex simulations that normally produce highly complex data comprising many characteristics. Such datasets are employed by scientists to validate their hypotheses or to present new hypotheses, and the data analysis is mostly restricted to time-consuming workflows only capable of a portion of the data's full spectrum. To support the experts, interactive visualization and analysis tools have been developed by different suppliers to manage and understand multivariate data. This article from the November/December 2017 issue of *Computing in Science & Engineering* demonstrates how one of these tools can improve data exploration of pulverized coal combustion.

IEEE Annals of the History of Computing

The "IBM Family": American Welfare Capitalism, Labor, and Gender in Postwar Germany

This article from the October–December 2017 issue of *IEEE Annals of the History of Computing* examines corporate labor and gender relations from a transatlantic perspective. It argues that the gendered communication of IBM's Thomas Watson Sr. shaped labor relations in his company's West German subsidiary. In the United States, Watson acted as a business progressive, expanding internationally, opening professional careers to young women, and implementing welfare capitalist measures. When IBM took tighter control of its foreign operations after World War II, Watson sought to implement welfare capitalist measures in the subsidiaries abroad. With his wife Jeanette by his side, he presented himself as the caring "pater familias." German IBM employees embraced Watson's conservative rhetoric of the IBM family but continued to join national unions and formed a works council, thwarting the major welfare capitalist goal of averting labor organization. Against such local labor practices, gendered communication undergirded a loyal workforce even in critical situations, an overlooked factor contributing to the company's success.

IEEE Cloud Computing

Intelligent Resource Management in Blockchain-Based Cloud Datacenters

Nowadays, more and more companies migrate business from their own servers to the cloud. With the influx of computational requests, datacenters consume tremendous energy every day, attracting great attention in the energy efficiency dilemma. In this article from the November/December 2017 issue of *IEEE Cloud Computing*, researchers investigate the energy-aware resource management problem in cloud datacenters, where green energy with unpredictable capacity is connected. Via proposing a robust blockchain-based decentralized resource management framework, they save the energy consumed by the request scheduler. Moreover, they propose a reinforcement learning method embedded in a smart contract to further minimize the energy cost. Because the reinforcement learning method is informed from the historical knowledge, it relies on no request arrival and energy supply. Experimental results on Google cluster traces and real-world electricity price show that their approach is able to reduce datacenters' costs significantly compared with other benchmark algorithms.

IEEE Computer Graphics and Applications

Visual Communication and Cognition in Everyday Decision-Making

The role of visual communication has quickly changed, and with the influence and evolution of new materials and technology, new commercial art and graphic design approaches have been created. From cuneiform (a writing system that dates back to Mesopotamia that used a stylus to imprint markings on clay tablets) to Johannes Gutenberg's development of metal movable type and the start of the printing revolution, materials and technology created opportunities for visual communication to reach more people and share more diverse messaging faster than ever before. Learn more in this article from the November/December 2017 issue of *IEEE Computer Graphics and Applications*.

IEEE Intelligent Systems

Robust Tracking of Soccer Robots Using Random Finite Sets

As in most multirobot systems applications, maintaining a good estimation of the other robots' positions is crucial in soccer robotics. Classical approaches use a vector representation of the robots' positions and Bayesian filters to propagate them over time. However, these approaches suffer from the data association problem. To tackle this issue, this article from the November/December 2017 issue of *IEEE Intelligent Systems* presents a new methodology for the robust tracking of robots based on the Random Finite Sets framework, which doesn't require any explicit data association. Moreover, the proposed methodology is able to integrate information shared by teammate robots, their positions, and their estimations of the other robots' positions. The proposed method is able to reduce the errors of the estimated robots' positions by about 35 percent.

IEEE Internet Computing

Nowcasting of Earthquake Consequences Using Big Social Data

Messages posted to social media in the aftermath of a natural disaster have value beyond detecting the event itself. Mining such deliberately dropped digital traces allows a precise situational awareness to help provide a timely estimate of the disaster's consequences on the population and infrastructures. Yet, to date, the automatic assessment of damage has received little attention. In this article from the November/December 2017 issue of *IEEE Internet Computing*, the authors explore feeding predictive models by tweets conveying on-the-ground social sensors' observations to nowcast the perceived intensity of earthquakes.

IEEE Micro

Ultra-Low-Power Processors

Society's increasing use of connected sensing and wearable computing has created robust demand for ultra-low-power (ULP) edge computing devices and associated system-on-chip (SoC) architectures. In fact, the ubiquity of ULP processing has already made such embedded devices the highest-volume processor part in production, with an even greater dominance expected in the near future. The Internet of Everything calls for an embedded processor in every object, necessitating billions or trillions of processors. At the same time, the explosion of data generated from these devices, in conjunction with the traditional model of using cloud-based services to process the data, will place tremendous demands on limited wireless spectrum and energy-hungry wireless networks. Smart, ULP edge devices are the only viable option that can meet these demands. Learn more in this article from the November/December 2017 issue of *IEEE Micro*.

IEEE MultiMedia

Deep Learning Triggers a New Era in Industrial Robotics

The pattern recognition capabilities of deep learning have pushed the limits in various fields—and industrial robotics is no exception. Deep learning alone will not solve all the problems encountered in industrial robotics, but it will certainly improve the perception capabilities of robotics systems, given its power to recognize complex real-world patterns robustly. In this article from the October–December 2017 issue of *IEEE MultiMedia*, the author examines robotics applications in deep learning.

IEEE Pervasive Computing

What Will We Wear After Smartphones?

With wearable computing research recently passing the 20-year mark, this survey from the October–December 2017 issue of *IEEE Pervasive Computing* looks back at how the field developed and explores where it's headed. According to the authors, wearable computing is entering its most exciting phase yet, as it transitions from demonstrations to the creation of sustained markets and industries, which in turn should drive future research and innovation.

IEEE Security & Privacy

The Future of Digital Forensics: Challenges and the Road Ahead

Today's huge volumes of data, heterogeneous information and communication technologies, and borderless cyberinfrastructures create new challenges for security experts and law enforcement agencies investigating cybercrimes. This article from the November/December 2017 issue of *IEEE Security & Privacy* explores the future of digital forensics, with an emphasis on these challenges and the advancements needed to effectively protect modern societies and pursue cybercriminals.

IEEE Software

Safe, Secure Executions at the Network Edge: Coordinating Cloud, Edge, and Fog Computing

System design where cyber-physical applications are securely coordinated from the cloud may simplify the development process. However, all private data are then pushed to these remote "swamps," and human users lose actual control as compared to when the applications are executed directly on their devices. At the same time, computing at the network edge is still lacking support for such straightforward multidevice development, which is essential for a wide range of dynamic cyber-physical services. This article from the January/February 2018 issue of *IEEE Software* proposes a novel programming model as well as contributes the associated secure-connectivity framework for leveraging safe coordinated device proximity as an additional degree of freedom between the remote cloud and the safety-critical network edge, especially under uncertain environment constraints.

IT Professional

The Economics of "Fake News"

False information has economic, political, and social consequences. The authors of this article from the November/December 2017 issue of *IT Professional* analyze the real and perceived costs and benefits to those that engage in the creation and platform support of false information. Special consideration is given here to digital advertising ecosystems that provide a supportive environment for "fake news" creation. Fake news is one type of false information. The authors discuss the context of fake-news consumption and suggest that fake-news creators, consumers, and various arbiters can reinforce each other and form a vicious circle. The article proposes mechanisms to break the circle and alter the cost-benefit structure of engaging in this activity.

Computing Now

The Computing Now website (computingnow.computer.org) features up-to-the-minute computing news and blogs, along with articles ranging from peer-reviewed research to opinion pieces by industry leaders. Read the latest Guest Editors' Introduction on advancing multimedia content distribution at tinyurl.com/y9yu5adj.

EDITOR'S NOTE

Big Data Analytics

Big data analytics is the use of advanced techniques to extract patterns, trends, and other insights from very large, diverse datasets that have one or more of the following characteristics: high volume, high velocity, or high variety. Vast amounts of data are generated from sensors, devices, video and audio, websites, and social media, and much of it requires close to real-time processing. Techniques such as text analytics, machine learning, data mining, natural language processing, and predictive analysis can help researchers and organizations access previously unavailable data sources and be able to gain new insights in applications ranging from healthcare and natural disaster response to fraud detection and traffic management. The March 2018 issue of *ComputingEdge* explores these topics and more.

In *Computer*'s "The Future of Data Management," Microsoft Research's David B. Lomet imagines a data-immersive world 50 years from now and what we can do to prepare for this new data-management landscape.

The author of *IT Professional*'s "The Problem of Personalization: AI-Driven Analytics at Scale" discusses how chief data officers can realize the personalization opportunities offered by massive amounts of available customer data and a variety of other big data sources.

In *IEEE Internet Computing*'s "Big, Linked Geospatial Data and Its Applications in Earth Observation," the authors posit that geospatial application development will become much easier if the terabytes of Earth observation data—currently stored in private archives—are freely available on the web.

The authors of *IEEE Cloud Computing*'s "Orchestrating BigData Analysis Workflows" argue that current big data analysis tools and workflow management orchestrators must evolve to a great degree before they can support the requirements of domain-specific big data workflow applications.

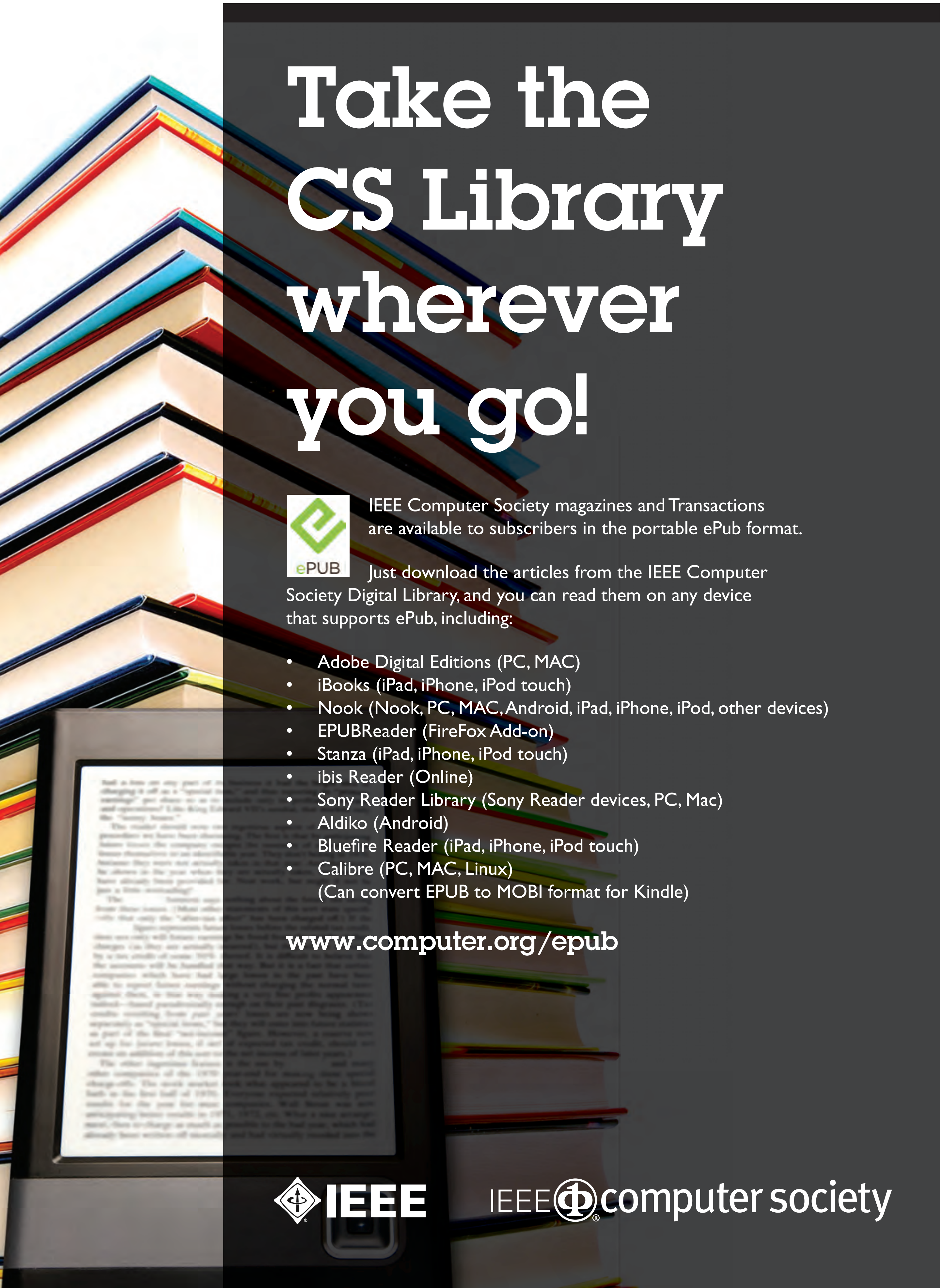
In *IEEE Pervasive Computing*'s "Population-Scale Pervasive Health," the author discusses research attempts to harness large-scale data that has already been collected through commercial devices and web applications to study human behaviors and the links between the data and health and well-being.

In *IEEE Internet Computing*'s "In Defense of MapReduce," the author presents a critical analysis of the dataflow operators provided by MapReduce and Spark, both of which are used in big data analytics.

Finally, the author of *IT Professional*'s "Big Data and Big Money: The Role of Data in the Financial Sector" discusses the relevance of big data approaches to the financial sector, outlining challenges to adoption and future opportunities for technology development.

This *ComputingEdge* issue also includes articles on topics other than big data analytics:

- In *IEEE Security & Privacy*'s "Silver Bullet Talks with Nicole Perlroth," Gary McGraw interviews a New York Times cybersecurity journalist who has covered stories on Russian election hacking efforts and more.
- *IEEE Software*'s "Should Architects Code?" argues that involving software architects in carefully selected implementation tasks can yield positive returns on investment for architects and their teams.


Take the CS Library wherever you go!
ePUB
IEEE Computer Society magazines and Transactions are available to subscribers in the portable ePub format.
Just download the articles from the IEEE Computer Society Digital Library, and you can read them on any device that supports ePub, including:
• Adobe Digital Editions (PC, MAC)
• iBooks (iPad, iPhone, iPod touch)
• Nook (Nook, PC, MAC, Android, iPad, iPhone, iPod, other devices)
• EPUBReader (FireFox Add-on)
• Stanza (iPad, iPhone, iPod touch)
• ibis Reader (Online)
• Sony Reader Library (Sony Reader devices, PC, Mac)
• Aldiko (Android)
• Bluefire Reader (iPad, iPhone, iPod touch)
• Calibre (PC, MAC, Linux)
(Can convert EPUB to MOBI format for Kindle)
www.computer.org/epub
IEEE
IEEE computer society

The Future of Data Management

David B. Lomet, Microsoft Research

Fifty years from today, we will live in a data-immersive world, doing things we have never done before.

Data management is driven by the ever-shrinking size and cost of a bit of storage, resulting in an increasing volume of data from which we can create value. We've grown accustomed to terabytes of data (with occasional petabytes), and exabytes are looming. Lower costs of data acquisition, storage, and management translate to increased data-driven opportunities. In turn, cloud vendors are growing their datacenter infrastructure at an astounding rate. Data needs to always be accessible and available, and must be easy to request. Online retailing with diverse shoppers, products, vendors, and means of delivery is an iconic example of this.

So, what does this mean for data management, and can we project what the world of data will look like in 50 years? As the famed saying goes, making predictions is difficult, especially about the future. Making a 50-year prediction is a fool's errand, but we can identify some long-term trends and possibilities. Let's first look back 50 years, both for inspiration and as a sanity check.

In the late 1960s, data management mainly meant business data processing. Online data management arrived around then (with some rounding), enabled by disk drives, and online transaction processing (OLTP) was born. Then, two great abstractions appeared that have served the data-management community well and should be relevant for the next 50 years: (1) transactions and (2) relational data models and algebra.

In a transaction, a user has the illusion that he is acting alone on the data. The result of the actions of all concurrent transactions is as if each transaction were executed serially without concurrency. If each transaction is correct in isolation, the overall result will be correct. Relational algebra transformed a collection of informal data-engineering operations into a mathematically sound closed system working on aggregated data (relations). We could then speak precisely of correct transformations and answers, equivalent but faster executions, and so on. Particularly important were ways to bring data together (joins) that permitted separating physical data formats from logical views and enabled diverse physical formats (column, row, hierarchical, tree, semistructured, and so on) to be treated within this algebra. Looking ahead, these two fundamental abstractions should continue to serve the data-management community well.

However, large portions of data management have needs that go beyond these abstractions. For example, dirty data and representation diversity are huge barriers to coherent access and manipulation of data from multiple sources. Web search solves this problem when human eyes are the target and "good enough" is good enough, but not when the target is another machine and when logical precision is a requirement. A new (well-paid) field called data science has evolved to cope with this.

Sensors produce astoundingly large dataflows that might need close to real-time processing and data reduction. Machine learning is being applied to ever-larger datasets to exploit the data for profit and for health, science, and the planet. Disasters happen, resulting in a need for data-disaster survival and for real-time, location-based information relevant to emergency responders. Social networks need data management at a global scale that is responsive to users everywhere, consistent, and available, with low latency and easy-to-use interfaces. New levels of well-being are possible when large parts of our environment can exploit data to serve us better—this is part of the promise of the Internet of Things (IoT).

We need to successfully grow the size and number of datacenters, which will transform the data-management landscape. Cloud vendors automate system operations, provide mechanisms for data security, assure data availability, and will shorten latency via edge servers, which will perhaps eventually be built into the Internet-switching infrastructure. Integrating geographically distributed data, applications, and disaster protection might well be enabled by this same infrastructure. However, the speed of light makes some latency unavoidable, and the CAP theorem (consistency, availability, and the resilience to network partitions) suggests that "having it all" will remain a challenge. But never underestimate clever engineering.

A revolution is beginning on how we query data. Computer cognoscenti might be able to cope with formal languages, but most users can't. For an end user, some queries can frequently be derived from context (for example, restaurants near your current location). But more sophisticated queries will increasingly be expressed at least partially verbally using increasingly conversational natural language. We are starting to see this with Cortana, Siri, and Alexa. The capability of this technology should steadily increase, and its use will eventually become second nature. The challenge is to turn conversational query into formal language query—but this will come.

Fifty years ago, we entered a data-assisted world. Today, the world is data-dependent—we can't check out at a store if their data systems are down. Fifty years from today, we will live in a data-immersive world, doing things we have never done before via data's ubiquitous integration into every facet of our lives. This has already begun. Enjoy the ride.

This article originally appeared in Computer, *vol. 50, no. 5, 2017.*

DAVID B. LOMET is a principal researcher at Microsoft Research and currently serves as the first vice president and treasurer of the IEEE Computer Society. Contact him at lomet@microsoft.com.

EDITOR: Seth Earley, Earley Information Science, seth@earley.com

The Problem of Personalization

AI-Driven Analytics at Scale

Seth Earley, *Earley Information Science*

During the past decade, a steady stream of arm-waving pronouncements has touted personalization as the holy grail of marketing programs. Organizations have been promised the ability to tap into billions of customer interactions and demographic data, and obtain new insights that yield personalization that presents the correct product, content, or solution for users no matter what they are trying to do or what role they are in.

This concept has been the topic of many events, papers, and research studies. It has spawned hundreds of technology solutions and services, creating entire product landscapes aimed at addressing this challenge. For the chief analytics officer and chief data officer (CDO), putting "analytics at scale" into production is the key to realizing the personalization opportunities that massive amounts of available customer data and a variety of other big data sources can offer.

No Magic Bullet

Despite all this investment and technology, organizations continue to struggle with personalizing customer and employee interactions. Many projects provide meaningful results in test environments, only to fail in production due to the high degree of human intervention required—whether during hypothesis development, data modeling, data preparation, or testing and fine-tuning. It is simply impractical and unsustainable to drive many analytic applications because it takes too long to produce usable results in the majority of use cases.

However, new and innovative approaches to using artificial intelligence (AI) and machine learning (ML) now enable accelerated personalization with fewer resources. The result is more practical and actionable customer insights that can be put to work without acts of heroics.

Too Much Data, Too Few Insights …

Today's applications, while powerful in some ways, lack maturity in certain areas. In particular, they cannot generate and automatically test the large number of hypotheses necessary to fully interpret the volume of data that is now being captured. The number of hypotheses and models is therefore limited to humans' capacity to repeat the process, leaving a majority of the possible analyses unexplored and on the virtual "cutting room floor."

Take the example of optimizing inventory for an airline. The goal is to maximize revenue while not degrading the traveler experience to the degree that there is an impact on customer loyalty. A personalization scenario might consider a traveler's recent experiences along with purchase history to offer an upgrade to specific ticket holders. Traditional analytic processes (articulating, testing, and tuning various scenarios) are

 Published by the IEEE Computer Society

Definition of Terms

Personalization describes the tailoring of a customer experience, or set of experiences, through the use of customer insights. These insights, collected from signals derived by analysis of customer data and a variety of other big data sources, must be available and actionable during a customer engagement to tailor the experience to individual tastes, needs, and behaviors.

Customer experience (CX) is the sum totality of how customers engage with your company and brand, not just in a snapshot in time, such as a website visit, but throughout the entire customer lifecycle. Karl Wirth, CEO of Evergage,[1] suggests that there are four core principles of CX: remember, understand, help, surprise/delight. Personalization should touch on each.

Analytics at scale describes the ability to perform analysis of massive datasets, referred to as big data. The analysis yields connections that enable personalization, relating a wide range of different customer data points—from digital marketing response, to website and search behavior, to service and support interactions, to reviews/sentiment data, as well as demographic, firmographic, and geographic data.

Big data refers to high-velocity, high-volume, highly variable data sources and combines structured and unstructured data.

Reference

1. K. Wirth, "Four Rules for Elevating Your Digital Customer Experience," Evergage blog, 25 Jan. 2016; bit.ly/2xLkh65.

not practical—disproportionate levels of resourcing and analysis are required to absorb and process the relevant data and construct models to achieve an optimized result.

… Not Enough Time and Attention

Customer spending on online channels is projected to continue on a growth trajectory (bit.ly/2sMlC9R). They are also transacting more business via mobile devices[1] and when on mobile devices, spending less time on e-commerce sites (bit.ly/2gfLmmZ). These trends all point to the need for increasing levels of personalization, where customers can be presented with the most relevant choices on a device with limited real estate. Online retailers have more data for analysis and personalization insight, but the cycle time in which it can be applied is rapidly diminishing. As consumer and business buyers continue to migrate their engagement to online experiences, the timeframes within which these experiences occur require more effective and efficient personalization. As shown in Figure 1, keeping pace with customer decisions while dealing with more and more customer data is becoming increasingly difficult.

Over US$2.1 trillion dollars will be spent via online transactions globally in 2017, a 26 percent increase in just two years. Depending on the type of device being used, the time to influence those decisions will shrink to just 5 to 7 minutes. In that amount of time, customer insights must be applied to personalize buying experiences.

Optimizing Personalization in Omni-Channel Marketing

An organization measuring the impact of various marketing channels, such as digital advertising response, social media, and outbound campaigns, might have the following inputs:

- customer satisfaction scores;
- likelihood to recommend;
- product or brand sentiment;
- performance measures, such as email click-through, website visits, conversions, abandonment rates, and sales transaction metrics across categories; and
- customer demographics, such as age, gender, ethnicity, wealth, education, and geolocation.

The organization might use tracking URLs for attribution and then determine the correlation of marketing campaigns with an increase in conversion events—whether an e-commerce transaction, a registration for content, or request for additional communications with a representative. Clickstream data provides additional signals to predict next actions or preferences linked to personas and use cases.

This mix will have a large number of inputs, coming from different systems. A conversion is the last step in a potentially long series of activities and interactions. Each step can strengthen or weaken the customer's perception of the brand and the relationship, ultimately predicting conversion potential.

Several important hypotheses can be developed and tested:

- What specific content is most effective in engaging across an omni-channel strategy?
- How does variation in tone (humorous versus factual, for example) impact the next behavior?
- How does that behavior change according to product line,

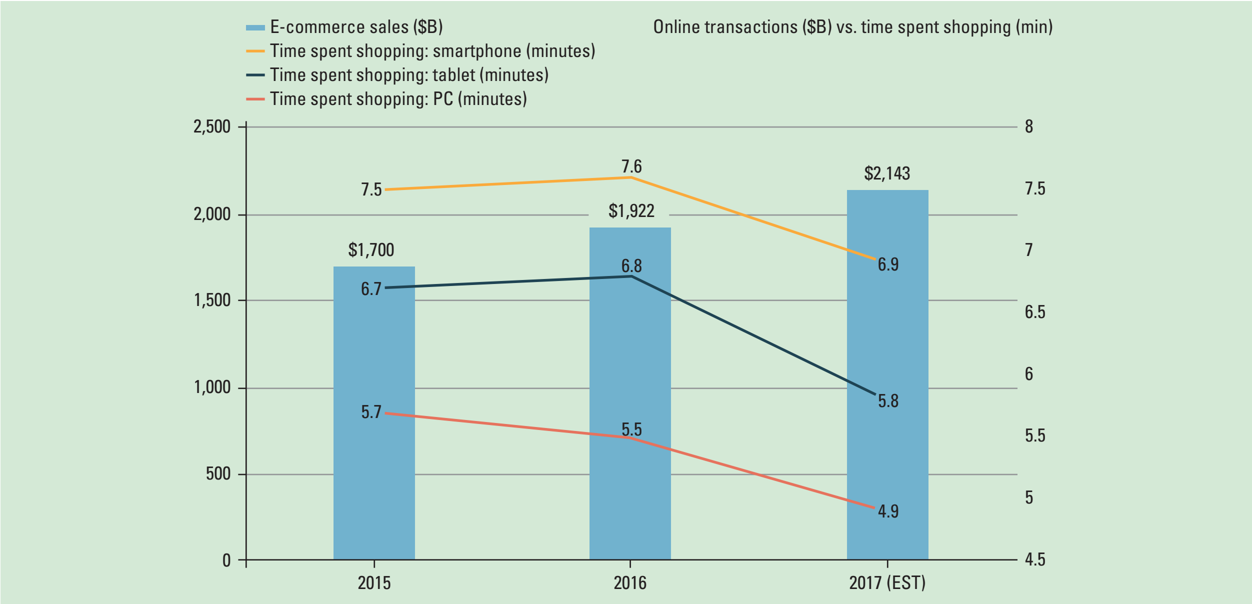


Figure 1. Global e-commerce sales ($billions) vs. time spent shopping (minutes) by device type. Keeping pace with customer decisions while dealing with more and more customer data is becoming increasingly difficult. (Source: Statista 2017)

channel, specific web property, demographic segment, or user intent?

- How do historical interactions impact the likeliness of future conversions?

The time and effort required to manually identify insights might not be justified by the uplift from applying those insights. Data modeling and hypothesis testing require development cycles that likely will not keep pace with the rate of change in the marketplace. Customer behaviors and trends change before results can be applied.

Even when modeling and testing is done efficiently, data collection, cleansing, integration, and structuring may still require significant effort prior to analysis. The preprocessing of data lengthens the timeline and increases the overall cost and level of effort for customer analytics and personalization programs. The clock speed of analysis, insight, and application cannot keep up with that of changing market dynamics and customer behaviors.

Nuances in insights can also be lost in translation from development to production or from data science teams to production analyst teams. Figure 2 summarizes these challenges. Scaling from research to production is challenging in every context and industry—from pharmaceuticals to manufacturing—and the answer is the same: standardization of processes and reuse of components and models to yield the efficiencies and benefits from scale. In other words, a factory model for analytics needs to be designed and developed to achieve benefits from scale.

Craft Data Models and Artisan Data Scientists

Craft beer and artisan breads are lovingly created by skilled individuals who slowly and carefully produce their goods. Many of the personalization models at first-generation retail sites have been similarly hand crafted by the equivalent of data science artisans. If your budget is big enough, it is also possible to purchase individually manufactured super cars for millions of dollars. Most of us are content with high-quality manufactured products that are exceedingly well designed and produced. Data science, predictive analytics, and machine learning applications are similarly becoming more practical and affordable, with capabilities that meet those of prior generations of applications costing orders of magnitude more.

What are the elements of modern personalization factories? A key element of achieving scalable personalization is the quality and provenance of data sources. Without harmonized data and consistent metadata, a great deal of work needs to go into cleaning up, performing extraction, translation, and load (ETL) functions, and making data usable for integration into analytic models. This is where the trouble usually starts. Data scientists end up spending a significant amount of their

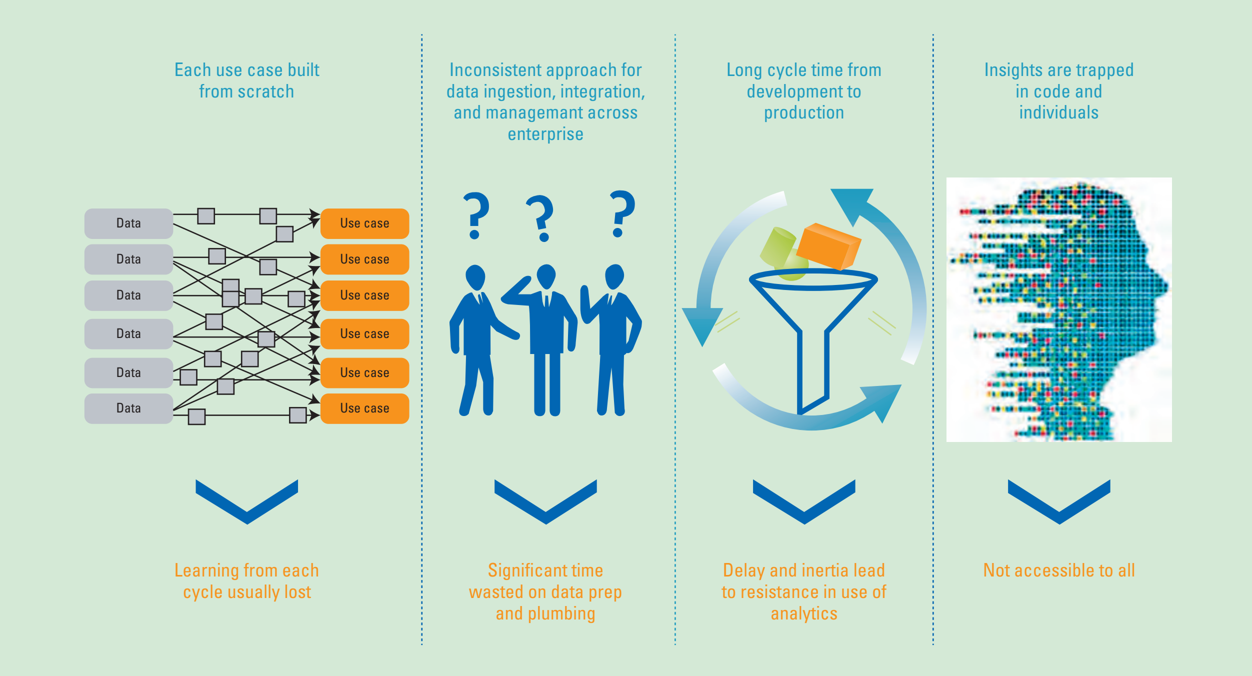


Figure 2. The problem with personalization projects. (Source: Opera Solutions, 2017)

time performing highly manual information architecture (IA) work—cleaning, structuring, and integration tasks—instead of accomplishing the higher-value analytical work that they were hired to do in the first place (Figure 3).

This data hygiene must be carried out for data to be clean and trusted enough to be used to solve focused business problems, such as the digital marketing scenario described earlier.

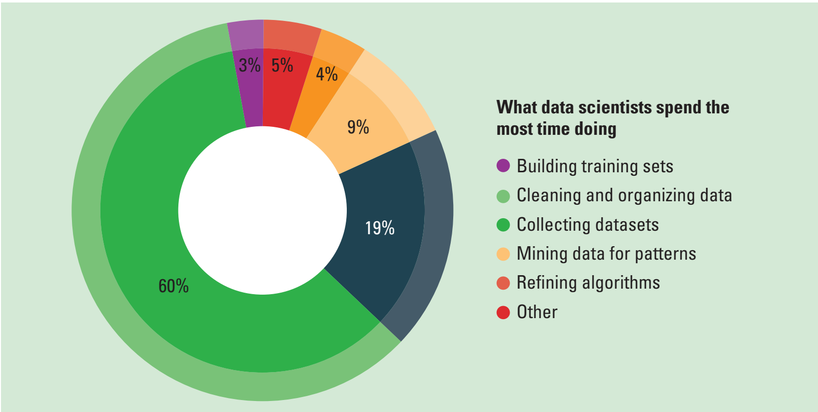


Figure 3. How data scientists spend their time. Data scientists end up spending considerable time on manual information architecture work rather than higher-value analytical work. (Source: CrowdFlower Research, 2016)

Making Data Scientists' Lives Easier

Sometimes, companies do not see an alternative to using data scientists for this task. The foundational requirements include a comprehensive view of data accuracy, validation, quality, and consistency to provide confidence in the output of personalization programs. These necessary elements should be part of the starting point for data scientists, not elements that they should be responsible for. Instead, they should be the responsibility of the business functions producing the data and the IT organization that provides the necessary governance infrastructure (in partnership with the business). In other words, fix the problems upstream through data stewardship and enterprise data standards rather than apply more costly remediation by scarce data science resources.

Development of consistent naming conventions and data definitions to make information more findable and improve access and data availability is another of the essential infrastructure elements. Metadata must be harmonized and optimized, with the removal of inconsistent and redundant terms and fields a part of data quality remediation. A formal business glossary and set of reference metadata (available in a catalogue, dictionary, or repository) is also part of these foundational requirements.

Many of the challenges related to creating clean data are assumed to be trivial because they are well understood. Although not mysterious, the challenges are by no means trivial and require changes to ownership, responsibility, process, and ultimately the culture of the organization, with a shift in responsibility from IT to the business (or at least partnership with the business). A great deal of work, including management of both process and content, must occur to cross the gulf between a hypothesis and that hypothesis tested, confirmed, and applied as a business innovation. In a production analytics environment, data and content preparation, application tuning, and configuration will constitute the bulk of the work and require the greatest effort. This is IA work, without which personalization is not possible.

A great deal of work must occur to cross the gulf between a hypothesis and that hypothesis tested, confirmed, and applied as a business innovation.

Rethinking Applied Analytics at Scale

Demand is exploding for scalable production solutions to personalization, while customer decision windows continue to narrow and choices expand. Thus, data scientists are increasingly looking for mechanisms to automate upstream data preparation and production. Several issues need to be addressed:

- Multiple projects in the organization, conducted by multiple teams or departments, begin with the same data inputs, which, unless consolidated or centralized, might require the same preparation work, resulting in duplication of effort. The wasted effort slows responsiveness across the functions that support different stages of the customer journey, multiplying the impact of the inefficiency on the customer experience.
- After data is prepared and models tuned (for example, optimizing offers for a particular segment), a lot of time and effort are required to go from the data science sandbox to operationalizing the solution. Integrating the processes and technologies in the customer experience environment is a time-consuming step. One example is defining the correct offer components across segments and serving those up through content and commerce applications.

This disconnect occurs because the development environment and the production environment are very different. Data scientists are probably not defining the needs of personas in market segments and are certainly not creating the content and merchandizing offers. The technology environments are also different. Data scientists might build a model using an analytic tool in the sandbox, and the IT department needs to recode variables and models using a more scalable toolset in Hadoop when the application goes into production.

Production data is more variable than test data because it goes beyond the test datasets, and data scientists have to retest against the model. When the projects are in final production, many of the insights and know-how are fragmented between documents in different repositories, code in various application iterations, and in the tacit experience of the people doing the work. As staff turns over, a great deal of this tacit knowledge is lost.

Platforms for AI-Driven Personalization

A platform approach to AI and ML techniques brings together many new, innovative personalization capabilities, along with data and content processing. The platform enables greater scale for building and optimizing analytical models. Using these techniques, an emerging approach combines key components of data preparation, data operationalization, and translation between business challenges and analytical models.

Multiple steps are required for raw data to travel the entire route from origin to insight, some of which can be aided by the platform. First, it must be preprocessed to normalize it, account for missing data, and review the data for accuracy. Second, a step called “feature engineering” is carried out to identify or generate variables. These steps need significant human input because they are part of IA. The remaining steps are feature selection (choosing a subset of all available features to simplify the model and produce results more quickly), model building, and operationalizing the insights.

Although not all aspects of feature engineering can be automated, various techniques can be used to generate many features systematically. Schema-driven feature generation captures the overall structure of various entities in a business, such as the customer, products, vendors, campaign constructs, and other objects. Patterns and behaviors can then be compared over time, with reference groups, industry peers, or other datasets. Deep learning can also be used to create features without domain-specific knowledge, with human input based on domain knowledge used to expand the model by generating additional features.

With a combination of systematic feature engineering and automated and system-assisted model building, hundreds of predictive models can be created faster and more efficiently. The development-to-production stage can also be increasingly automated through a platform approach. A semantic layer around models, data sources, and insights allows users to search and re-use inputs and results.

The end result is the "democratization" of data science capabilities across the enterprise, rather than those capabilities residing in a few well-staffed and well-funded departments. This capability frees data scientists to build more models and allows business intelligence staff and business users to benefit from the power of data science insights on a day-to-day basis.

Through this process, data becomes a service that fuels algorithms. The platform becomes

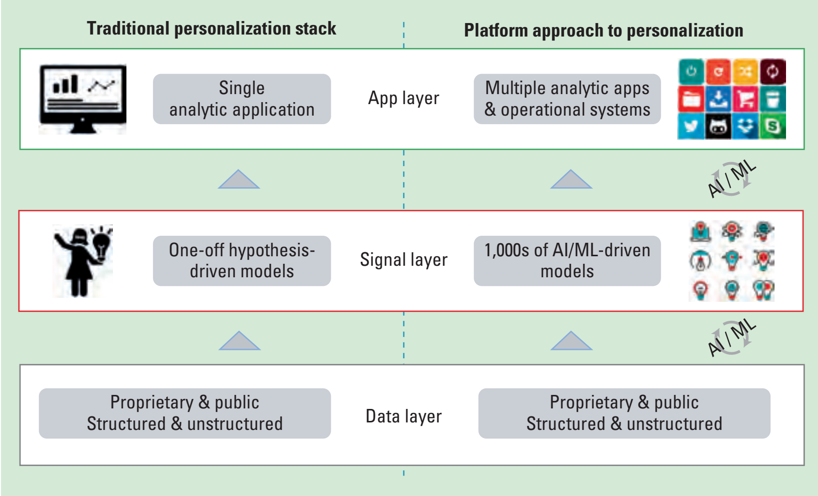


Figure 4. Traditional vs. platform approach to personalization.

an orchestration layer between the data and the technologies that solve business problems. Signals from the data are processed and interpreted in the orchestration layer, and algorithms operate on those signals to produce an output to one or more technologies that act on the output. This signal processing/orchestration layer has sometimes been described in shorthand as a "signal layer."

The signal layer brings together the processing of large volumes of data, including real-time behaviors from websites, social media, or sensors as well as operational and transactional information. This approach helps to standardize and consolidate sources, making them more readily available to an analytics workbench in which algorithms are tuned and tested. The approach can also be used to package algorithms for export to production systems without loss of fidelity. Figure 4 compares a signal-layer approach to the traditional stack for creating personalization models and insights.

Netflix recently enhanced its personalization functionality, which allows individual users to receive tailored programming suggestions. This eliminates the need for multiple users in a household to share one set of recommendations. By knowing something about the preferences of individual users, Netflix can now correlate this information across millions of user "signals" and refine its recommendation engine based on the insights it gleans every minute. Establishing this capability using a signal layer allows Netflix data scientists to continuously receive new customer insights and test hypotheses rather than have to rebuild analytical models for each hypothesis.

Getting more value from data and advanced approaches to applying data is a top challenge of chief marketing officers (CMOs; bit.ly/2ya3B78). However, many organizations lack the expertise in their marketing departments to exploit data assets using AI and ML. Leveraging a platform- and signal-layer approach can help CMOs make better use of scarce talent and reuse solutions that are developed across the enterprise. CDOs are tasked with improving and enriching data and increasing the value from data assets (bit.ly/2yabUzB). To do so, the CDO needs a mechanism to make data readily available and in a format that can be consumed and applied by business users. Tackling data preprocessing using a platform approach helps to achieve this objective.

The objective of a preprocessing approach is to process the data independent of its future use, and then apply AI and ML in the signal layer to drive business application functionality. The approach impacts data scientists' time efficiency by making inputs independent of use cases and operationalizing analyses using standardized algorithms that can be developed as libraries of AI functionality. The system captures learning for reuse and avoids one-off data modeling and algorithm-developing projects. As new insights are discovered and feedback from operations is incorporated, the signal-layer/platform approach becomes a continuous learning system for the enterprise, driven by the data variables as their impact on business problems is discovered. IT

Reference

1. M. Lazar, "Click. Buy. Done. These 'M Commerce' Statistics Show that Smartphone Shopping Is Viral," *Huffington Post* blog, 22 Aug. 2016; bit.ly/2gKpAIm.

This article originally appeared in IT Professional, *vol. 19, no. 6, 2017.*

***Seth Earley** is CEO of Earley Information Science (www.earley.com). He's an expert in knowledge processes, enterprise data architecture, and customer experience management strategies. His interests include customer experience analytics, knowledge management, structured and unstructured data systems and strategy, and machine learning. Contact him at seth@earley.com.*

Editor: Carole Goble • cag@cs.man.ac.uk

Big, Linked Geospatial Data and Its Applications in Earth Observation

Manolis Koubarakis, Konstantina Bereta, George Papadakis, Dimitrianos Savva, and George Stamoulis • *National and Kapodistrian University of Athens*

If the terabytes of Earth observation data currently stored in archives are published on the web using the linked data paradigm, data discovery, integration with other data sources, and the development of applications will become much easier.

Terabytes of geospatial data have been made freely available recently on the web. For example, data are available from gazetteers such as GeoNames, maps from geospatial search engines like Google Maps and OpenStreetMap, and user-contributed content from social networks such as Foursquare.

Some particularly important rich sources of open and free geospatial data are the satellite programs of various countries, such as the Landsat program of the US and the Copernicus program of the European Union (EU). Satellite images can be used in many applications with financial and environmental impact in areas such as emergency management, climate change, agriculture, and security. This potential has not been fully realized because satellite data are hidden in various archives operated by NASA, the European Space Agency, and other national space agencies. Therefore, application developers need to search in these archives to discover the needed data and integrate them into their applications. In this article, we show how to break these barriers by publishing this data in the Resource Description Framework (RDF), interlinking it with other relevant data, and making it freely available on the web to enable easy development of geospatial applications.

Big, Linked, and Open EO Data Lifecycle

The life of Earth observation (EO) data starts with the data's generation in the ground segment of a satellite mission, where the management of this so-called payload data is an important activity. Figure 1 gives a high-level view of the lifecycle of big, linked EO data as we envisioned it in our work. Each phase of the lifecycle and its associated software tools is discussed in more detail in the following.

Ingestion, Processing, Cataloguing, and Archiving

Raw data, often from multiple satellite missions, are first ingested, processed, catalogued, and archived. This phase involves processing results in the creation of various standard products (Levels 1, 2, and so on, in EO jargon; raw data are Level 0), together with extensive metadata describing them.

Satellite Image Descriptor Extraction, KDD, and Semantic Annotation

We extended traditional image-processing methods to deal with the specificities of satellite images and extract image descriptors — for example, texture features or spectral characteristics of an image. Knowledge discovery and data mining (KDD) techniques combine image

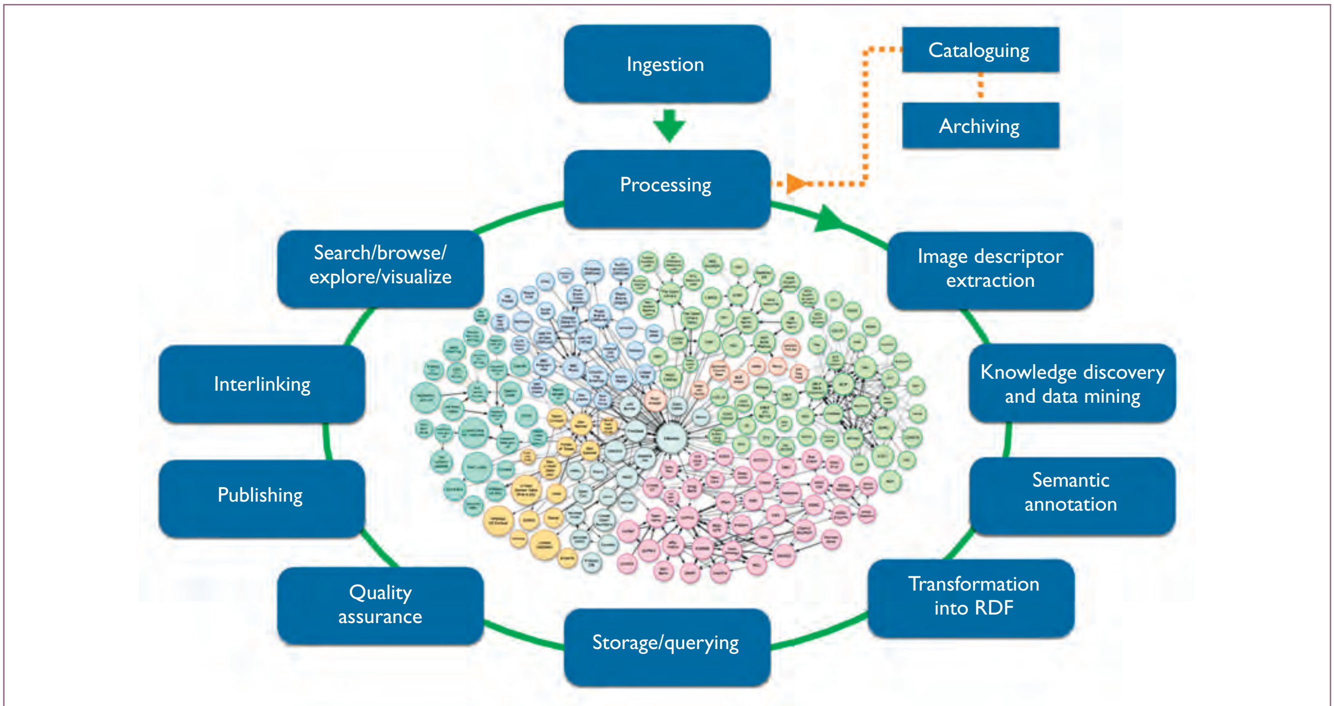


Figure 1. The lifecycle of big, linked, open Earth observation (EO) data. The yellow dashed line indicates traditional processing chains in EO datacenters. The green circle captures our additions.

descriptors, image metadata, and auxiliary data (such as GIS data) to determine concepts from a domain ontology (for example, a forest, lake, fire, or burned area) that characterize an image's content. Hierarchies of domain concepts are formalized using ontologies encoded in the Web Ontology Language, OWL-2, and are used to annotate standard products. Annotations are expressed in RDF and are made available as linked data so that they can be combined easily with other publicly available linked data sources (such as GeoNames, OpenStreetMap, and DBpedia) to allow for the expression of rich user queries.

Semantic Annotation

For encoding semantic annotations and publishing geospatial and temporal linked data, we developed the data model stRDF and the query language stSPARQL. The model stRDF is an extension to the W3C standard RDF that allows the representation of geospatial data that changes over time. It is accompanied by stSPARQL, an extension of the query language SPARQL 1.1 for querying and updating stRDF data. Both stRDF and stSPARQL use the Open Geospatial Consortium (OGC) standards Well-Known Text (WKT) and Geography Markup Language (GML) for the representation of temporal and geospatial data. Both stRDF and stSPARQL have been implemented in the system Strabon (see http://strabon.di.uoa.gr), which extends the well-known RDF store Sesame and uses PostgreSQL or MonetDB as the backend spatially- and temporally-enabled database management system. As shown by our experiments, Strabon is currently the most functional and efficiently performing geospatial and temporal RDF store available.

In our work, we use stRDF to represent satellite image metadata (for example, time of acquisition or geographical coverage), knowledge extracted from satellite images (for example, a certain image pixel is a fire hotspot), and auxiliary geospatial datasets encoded as linked data. We can then use stSPARQL to express in a single query an information request such as the following: find an image taken by a Meteosat second-generation satellite on 25 August 2007 that covers the area of Peloponnese and contains hotspots corresponding to forest fires located within 2 km of a major archaeological site. Encoding this information request today in a typical interface of an EO data archive such as the Copernicus Open Access Hub (see https://scihub.copernicus.eu) is impossible, because domain-specific concepts such as "forest fires" aren't included in the archive metadata, and thus they can't be used as search criteria. In Copernicus Open Access Hub and other similar web interfaces, search criteria include a hierarchical organization of avail able products (for example, high-resolution optical or synthetic aperture radar data), together with a temporal and geographic selection menu.

Semantic Catalogue for the TerraSAR-X Archive

The workflow for constructing a semantic catalogue for the TerraSAR-X archive of Germany's Aerospace Center (DLR) can be summarized as follows. First, the TerraSAR-X products are obtained from the archive and stored separately into an image and metadata database. Then, each image is tiled into patches based on the resolution and pixel spacing extracted from the metadata database. For each patch, its quick-look is generated and stored into a quick-look database. Then, the primitive features from each tiled patch are extracted and stored into a primitive feature database. The three databases are implemented using MonetDB. Then, the features are grouped into categories from a predefined hierarchy (the DLR ontology) using an interactive learning algorithm. These categories are used to populate the semantic catalogue.

As a proof of concept, this workflow has been applied to a big dataset containing 300 scenes from the DLR TerraSAR-X archive (around 3 Tbytes of data). Applying the knowledge discovery and data mining (KDD) framework to this dataset resulted in detecting 850 semantic classes with high precision and recall.[1] As an example, Figure A shows the visualization of the information regarding the area of Venice that was introduced in the semantic catalogue using Sextant.

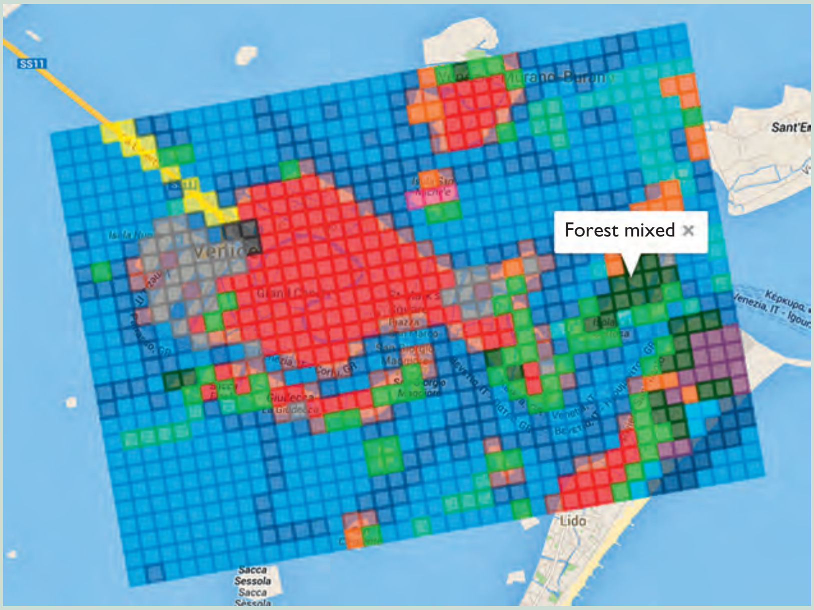


Figure A. Visualization of the information regarding the area of Venice that was introduced in the semantic catalogue using Sextant.

Reference

1. D. Espinoza-Molina et al., "Very-High-Resolution SAR Images and Linked Open Data Analytics Based on Ontologies," *IEEE J. Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 4, 2014, pp. 1696–1708.

With the KDD techniques, we can characterize satellite image regions with concepts from appropriate ontologies (for example, land-cover ontologies with concepts such as water-body, lake, and forest; or environmental monitoring ontologies with concepts such as forest fire and flood). These concepts are encoded in OWL ontologies and are used to annotate EO products. In this way, we attempt to close the semantic gap that exists between user requests and searchable information available explicitly in the archive.

But even if semantic information was included in the archived annotations, we would need to join it with information obtained from auxiliary data sources to answer the previous query. Although such open sources of data are available to EO datacenters, they aren't currently used to support sophisticated ways of end-user querying in web interfaces such as the Copernicus Open Access Hub. In our work, we assume that auxiliary data sources, especially geospatial ones, are encoded in stRDF and are available as linked geospatial data; thus, stSPARQL easily can be used to express information requests as in the previous example.

Transformation into RDF

This phase transforms vector or raster EO data from their standard formats (for example, ESRI shapefile or NetCDF) into RDF. We advanced the state of the art in transforming EO data and geospatial data into RDF by developing GeoTriples (see https://github.com/LinkedEOData/GeoTriples). GeoTriples is a tool that transforms vector data and their metadata into RDF and that natively supports many popular geospatial data formats, including shapefiles, spatially enabled DBMS, Keyhole Markup Language (KML), and GeoJSON.

Storage/Querying

This phase deals with storing all relevant EO data and metadata on persistent storage so that they're readily available for querying in subsequent phases. In our work, we use MonetDB (see www.monetdb.org) for the storage of raw image data and metadata; while we use the spatiotemporal RDF store system Strabon and the query language stSPARQL for storing/querying semantic annotations and other types of linked, geospatial data possibly originating from transforming EO products into RDF.

Often, relevant geospatial data are stored in geospatial relational databases (for example, PostGIS) and aren't available as linked data. When these databases are frequently updated and/or are large, domain experts are discouraged from transforming the data into RDF and then storing it in a triple store such as Strabon. For this reason, we developed the system Ontop-Spatial, which is a geospatial extension of the Ontology-Based Data Access system Ontop (see https://github.com/ConstantB/ontop-spatial). Ontop performs on-the-fly SPARQL-to-SQL translation on top of relational databases using ontologies and mappings. Ontop-Spatial extends Ontop by enabling on-the-fly GeoSPARQL-to-SQL translation on top of geospatial databases. Our experimental evaluation showed that this approach is not only simpler for users (because it doesn't require materialization of data), but is also more efficient in terms of query response time.

Quality Assurance

Before linked EO data are ready for publication, this step cleans the data by, for example, removing duplicates and so on. An important issue in this phase is entity resolution, which can also be viewed as part of the linking phase.

Publishing

This phase makes linked EO data publicly available in the linked open data (LOD) cloud using well-known data repository technologies such as the Comprehensive Knowledge Archive Network (CKAN). In this way, others can discover and share this data, avoiding duplication of effort.

Interlinking

This is an important phase in the linked EO data lifecycle, because much of linked data's value comes through connecting seemingly disparate data sources to each other. Until now, there hasn't been much research or tools for interlinking linked EO data. If we consider other published linked datasets that aren't from the EO domain, but have similar temporal and geospatial characteristics, the situation is the same. These datasets are typically linked only with `owl:sameAs` links and only to core datasets such as DBpedia or GeoNames. In addition, links are often created manually.

With our work, we advance the area of interlinking of linked open data by concentrating on the geospatial, temporal, and measurement characteristics of EO data. Specifically, we address the problem of discovering other types of geospatial or temporal semantic links. In linked EO datasets, it's often useful to discover links involving topological relationships, for example, `A geo:sfContains F` where `A` is the area covered by a remotely sensed multispectral image `I`, `F` is a geographical feature of interest (field, lake, city, and so on), and `geo:sfContains` is a topological relationship from the topology vocabulary extension of GeoSPARQL. The existence of this link might indicate that `I` is an appropriate image for studying certain properties of `F`.

We dealt with these issues by extending the well-known link discovery tool Silk to discover precise geospatial and temporal links among spatiotemporal RDF data (see http://silkframework.org).

Search/Browse/Explore/Visualize

This phase enables users to find, explore, browse, and visualize the data they need, and start developing interesting applications.

For this phase of the lifecycle, we developed the tool Sextant (see http: //sextant.di.uoa.gr). Sextant is a Web-GIS tool that produces maps by combining geospatial data from SPARQL endpoints and well-known GIS file formats. To achieve interoperability with other well-known GIS tools, Sextant is based on OGC standards for vector and raster data such as WKT, GML, KML, and GeoJSON. Sextant supports the creation of layers using the OpenGIS Web Map Service Interface Standard that's a standard protocol for serving georeferenced map images over the web, and the OGC Web Feature Service 2.0 Interface Standard that defines interfaces for describing data manipulation operations of geographic features.

Application Examples

The sidebar "A Semantic Catalogue for the TerraSAR-X Archive" showcases the lifecycle of big linked open EO data in a working application. In a related article, the lifecycle is presented with more details, and is applied to the case of wildfire monitoring using satellite images and related GIS data.[1]

More recently, the tools presented here were used in the Big Data Europe project (see www.big-data-europe.eu) to develop a pilot application in the area of space and security, The pilot aims to enhance the process of detecting changes in land cover or land use from satellite images (for example, the construction or destruction of settlements) and correlating them with the detection of geolocated events in news sites and social media. Interweaving remote sensing with social sensing constitutes a key advancement in the space and security domain, where useful information can be derived not only from EO products, but also from their combination with news articles and the user-generated content from social media. In the pilot, the tool GeoTriples is used to transform geospatial data into RDF, Strabon is used to store linked geospatial data, and Sextant has been extended to function as a user-friendly graphical interface for the whole application.

Big, linked, and open EO data can be managed using the technologies developed in the TELEIOS and Linked Open Earth Observation Data (LEO) projects. Our group's work presented here concentrates more on linked open data and has only partially addressed big data issues. The area of big data is where our current work concentrates. We're reengineering GeoTriples, Strabon, and Ontop-Spatial to take advantage of big data technologies Apache Hadoop and Spark and their recent extensions for big, geospatial data. All the tools presented here (Strabon, Ontop-Spatial, GeoTriples, Silk and Sextant) are available as open source.

Acknowledgments

This work has been funded by European Union Seventh Framework Programme (FP7) projects TELEIOS (257662), LEO (611141), Melodies (603525), and Horizon 2020 projects BigDataEurope (730124) and Copernicus App Lab (730124).

Reference

1. M. Koubarakis et al., "Managing Big, Linked, and Open Earth-Observation Data: Using the Teleios/LEO Software Stack," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 3, 2016 pp. 23–37.

Manolis Koubarakis is a professor at the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens. His research interests include artificial intelligence, the Semantic Web, and linked data. Koubarakis has a PhD in computer science from the National Technical University of Athens. Contact him at koubarak@di.uoa.gr.

Konstantina Bereta is a PhD student at the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens. Her research interests include the Semantic Web and linked data. Bereta has an MSc in informatics and telecommunications from the National and Kapodistrian University of Athens. Contact her at konstantina.bereta@di.uoa.gr.

George Papadakis is a postdoctoral fellow at the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens. His research interests include artificial intelligence, databases, and the web. Papadakis has a PhD in computer science from the Leibniz University of Hanover. Contact him at gpapadis@di.uoa.gr.

Dimitrianos Savva is a research assistant at the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens. His research interests include the Semantic Web and linked data. Savva has an MSc in informatics and telecommunications from the National and Kapodistrian University of Athens. Contact him at dimis@di.uoa.gr.

George Stamoulis is a PhD student at the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens. His research interests include the Semantic Web and linked data. Stamoulis has an MSc in informatics and telecommunications from the National and Kapodistrian University of Athens. Contact him at gstam@di.uoa.gr.

This article originally appeared in IEEE Internet Computing, *vol. 21, no. 4, 2017.*

Orchestrating BigData Analysis Workflows

Rajiv Ranjan,
Newcastle University

Saurabh Garg
University of Tasmania

Ali Reza Khoskbar
Australian National University

Ellis Solaiman, Philip James
Newcastle University

Dimitrios Georgakopoulos
Swinburne University of Technology

Data analytics has become not only an essential part of day-to-day decision making, but also reinforces long-term strategic decisions. Whether it is real-time fraud detection, resource management, tracking and prevention of disease outbreak, natural disaster management or intelligent traffic management, the extraction and exploitation of insightful information from unparalleled quantities of data (BigData) is now a fundamental part of all decision making processes. Success in making smart decisions by analyzing BigData is possible due to the availability of improved analytical capabilities, increased access to different data sources, and cheaper and improved computing power in the form of cloud computing. However, BigData analysis is far more complicated than the perception created by the recent publicity. For example, one of the myths is that BigData analysis is driven purely by the innovation of new data mining and machine learning algorithms.

While innovation of new data mining and machine learning algorithms is critical, this is only one aspect of producing BigData analysis solutions. Just like many other software solutions, BigData analysis solutions are not monolithic pieces of software that are developed specifically for every application. Instead, they often combine and reuse existing trusted software components that perform necessary data analysis steps. Furthermore, in order to deal with the large variety, volume and velocity of BigData, they need to take advantage of the elasticity of cloud and edge datacentrer computation and storage resources as needed to meet the requirements of their owners. More specifically, many BigData analysis solutions today are organised as data-driven workflows that combine existing and new data analysis steps (which we often refer to as workflow activities).

The flow of information between the analysis activities in a BigData analysis workflow is *dynamic*, meaning it is either determined by the data produced in earlier steps in the workflow (we refer to these as data flow dependencies) or by the structure of the BigData analysis solution that orchestrates the data analysis activities in the workflow (we refer to such structural orchestrations as control flow dependencies). Another dynamic aspect of BigData analytics workflows is mapping data analysis steps/activities to the variety of computing and storage resources of the cloud and edge data center(s) with changing performance. Dealing with these dynamic aspects become more challenging in BigData analysis applications which need to support owner's de-

 Published by the IEEE Computer Society 2469-7087/18/$33.00

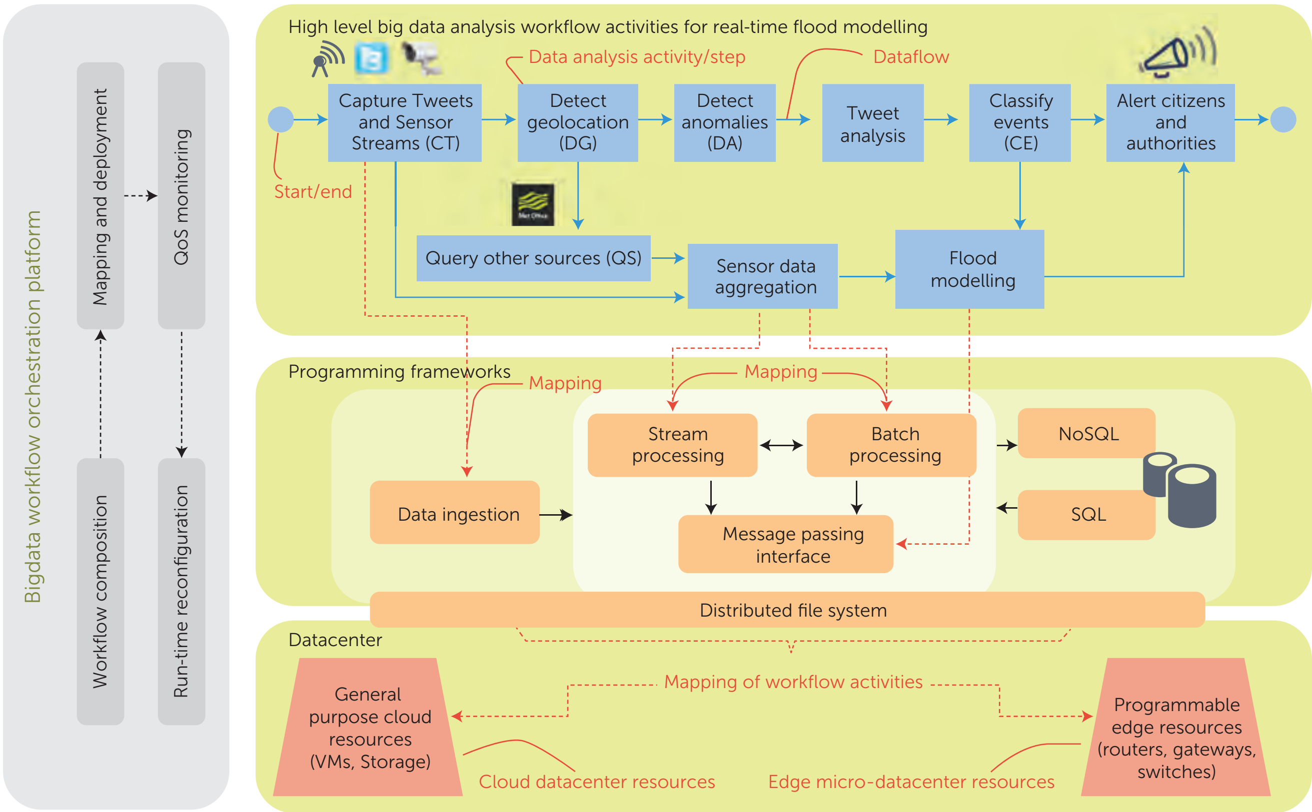


FIGURE 1. Mapping of high level workflow activities of Real-Time Flood Modelling application to programming frameworks and cloud datacenter and/or Edge resources. The workflow orchestration is a cross-cutting issue as it spans across all the layers (analysis activities, programming framework, and datacenters).

cision making requirements (specified in form of Service Level Agreements (SLA)) in real-time. Any delay in meeting their requirements can cause loss of life (such as in disaster prediction and response situations), money (for example, in banking security and fraud situations), or the environment (for instance, in resource exploration). These are some of the real penalties for failing to meet the real-time data analysis requirements in such decision support applications. Computing infrastructures supported by cloud and edge resources can help in solving such problems to some degree by providing elastic and on-demand computing infrastructure. They can also create additional challenges due to the heterogeneous nature of different cloud and edge resources and the dynamically changing performance of their computing infrastructure.

In this Blue Skies installment, we point out the requirement of orchestration systems that can assist in management and execution of such BigData analysis workflows on a cloud and edge infrastructure. We also discuss current state of art and point out open issues in a later section before concluding the article.

An example BigData analysis Workflow

As an example of a BigData analysis workflow, consider Real-Time Flood Modelling (RTFM) for detecting and predicting a flooding event by analyzing tweets and sensor data, as depicted in Figure 1. The RTFM workflow is triggered from long-range forecasting (for example from UK Met Office DataPoint) and radar scans at multiple scales are initiated and passed to statistical processing models, updating probability based forecasts.

As an event progresses, streaming data sources (such as Twitter and ancillary data comparable to traffic flows) can be processed to improve modelling forecasts of a rainfall event's path and intensity. Flood modelling ensembles must then be triggered and matched to known observations (for example from CCTV analysis or rain gauges) in a dynamic system. Flood model outputs are only part of the modelling process providing input into risk and impact models. All of this is happening within a fluid, dynamic, evolving ecosystem where models are refined, re-run or abandoned as new information becomes available. In other words, the workflow includes several top-level data analytics activities. These include long-range forecasting, sensor data aggregation, Tweet analysis, flood modelling, CCTV image processing, and so on. Moreover, the execution of these activities need to be seamlessly coordinated such that real-time decision-making performance objectives (for instance, minimise event detection delay) are constantly achieved under various types of uncertainties (for example, changing data volume and velocity). Hence, the key to seamless execution of this new class of workflows is the issue of resource and data orchestration, which is quite complex due to complex BigData flow pattern and the plethora of BigData programming frameworks, computational models, and infrastructure types (such as cloud datacenters and edge resources) involved:

1. The latency sensitive CCTV image processing activity can benefit by performing "edge analytics" on the video frames by exploiting the on-board processor (edge resources) supported by current generation of CCTV cameras (such as Waggle platform). Using edge analytics techniques has multiple benefits: (i) reduced network congestion achieved by filtering non-relevant events at the edge; and (iii) reduction in event detection latency (for example, detecting dangerous water flow level by analysing real-time images on-board processors available within CCTV cameras) as sensors no longer need to send data to far off cloud datacenters.
2. The flood modelling activity, which does risk analysis by executing a complex hydrodynamic computational model in a message passing interface data programming framework (OpenMPI), should be mapped to the cloud resources, because it is demanding of both storage (due to large historical rainfall records and ensemble city models) and computation (for simulating floods along large river reaches).
3. Workflow activities are inter-dependent and changes in execution characteristics of one activity (at run-time) will influence others. For example, the step handling the flood modelling is dependent on input (on rain and water level thresholds) from the sensor aggregation activity (analysing data from diverse real-time sensors).
4. Tweet analysis activity requires distinct computational models for anomaly detection (flood disasters are anomalous tweets), clustering to combine all the information from different tweets reporting flooded properties in a specific location, and classification to identify major events such as a flood. Moreover, these computational models require either a batch processing or stream processing data programming framework, depending on data characteristics (historical vs. real-time tweets). The activity needs to utilise specialised main memory NoSQL BigData framework and solid state storage resources available in the cloud datacenter to deal with Twitter's data velocity and volume.

To handle these complexities, the underlying Orchestration[1] platform and techniques should be able to dynamically manage a workflow of activities (initially composed based on Domain expert inputs) on the resources available in the cloud datacenter (for example, Amazon Web Services) and on the edge (such as the Waggle platform) driven by processing needs (for instance latency sensitive vs. non-latency sensitive), performance objectives (for example, minimise sensor stream processing latency vs. minimise flood model execution delay) and type of analytic tasks (CCTV image processing vs. flood modelling) relevant to activities. Current BigData workflow orchestration platforms (such as, Apache YARN, Apache Mesos, AWS Lambda, AWS IoT, Google Cloud Dataflow, Google TensorFlow) and research assume either monolithic and purpose-built data analysis solutions that do not need to meet real-time decision support requirements (that is, no workflows, no dynamic orchestration of existing and new data analysis activities, no implementation that can exploit both cloud datacenter and edge resources, and no dynamic tuning of such implementations to meet the users' real-time decision making requirements), or considers only solutions consisting of data analysis workflows that have predictable performance (that is existing orchestration research ignores the complexities of resource and BigData management across cloud datacenter and edge resources for data analytics workflows and does not deal with meeting real-time performance objectives as determined by owner's SLA requirements).

Last but not the least, the existing workflow composition frameworks such as OASIS TOSCA,[2] was developed for web services based workflows and allows workflow modelling and deployment specification up to two levels, software components and cloud services (that is, infrastructure). They do not allow composition of workflows at three different layers (see Figure 1) first at analytical activities, then at programming framework, and finally at datacenter layer, nor do they allow integration of dynamic QoS requirements of decision makers.

Hence, the key research challenges that we perceive are the development of orchestration platforms and techniques that can aid in dynamically composing workflows through an analytical workflow composition framework and developing a robust run-time algorithms that can automatically manage the allocation of the datacenter and edge resources to the analytic activities in response to unexpected changes in data volume, data velocity or other infrastructure level issues (for example, congestion, availability, load-balancing, or anomalies, and so on.).

Understanding the BigData workflow Orchestration Challeneges

To support such complicated and dynamically configurable BigData workflow ecosystems, we need a new orchestration platforms and techniques for managing three layers: (i) sequence of data analysis activities (the workflow) that needs to deal with real-time and historical datasets produced by different sources; (ii) heterogeneous BigData programming frameworks; and (iii) the heterogeneous cloud and/or edge resources. The BigData workflow orchestration is a multi-level resource management and coordination process that spans across workflow activities, BigData programming frameworks and cloud/edge resources. It includes a range of programming operations, from workflow composition, mapping of workflow activities to BigData programming frameworks and cloud/edge resources, to monitoring their end-to-end run-time QoS and SLA statistics (for example, event detection delay, alert delay, load, availability, throughput, utilization, latency, etc.) for ensuring consistency and adaptive management. Briefly stated, major research challenges involved with developing orchestration platforms and techniques for BigData workflow applications include:

Workflow composition: In a BigData analysis workflow (such as RTFM in Figure 1), workloads (data volume and velocity) pertaining to different activities are dependent on each other and changes in execution and data flow of one activity will influence others. For example, the flood modelling activity is dependent on the real-time input on rain and water level thresholds from the sensor data aggregation and CCTV image processing activities. Hence, the hard challenges exist in developing workflow composition framework that can guide the domain experts (for example, flood modeller in a city council office) in specifying, understanding and managing the whole pipeline of activities, data and control flow inter-dependencies and their QoS and/or SLA objectives and measures. For example, suppose we have two owners and/or decision makers for the workflow in Figure 1. The first owner is from a national disaster centre who is interested in information about any infrastructure damage, while another owner from the emergency management services (EMS) may be interested in information about human fatalities and injuries. In this case, the workflow in Figure 1 will dynamically need to compose different clustering activities (infrastructure damages vs human fatalities) that will both utilise the data flow from the anomaly detection activity. Hence, based on decision maker goal workflow composition pattern changes. Moreover, the problem is further complicated by the fact that type and mix of workflow activities, data and control flow inter-dependencies and their QoS and/or SLA measures varies significantly across different application domains (such as, real-time air pollution monitoring, real-time traffic congestion monitoring, remote patient monitoring, etc.).

Workflow mapping: Mapping BigData workflow (graph of data analysis activities) to BigData programming frameworks and cloud/edge resources demands selecting bespoke configurations from abundance of possibilities. Therefore, the mapping process for has to take into account diverse configuration selection decision. For example,

- *BigData programming frameworks:* Select optimal configurations for each framework (for example, in context of stream processing engine such as Apache Storm one needs to determine optimal mix and number of spouts, bolts, and worker instances to minimize data processing latency of stream processing activities)
- *Cloud resources:* Consider configurations such as datacenter location, pricing policy, server hardware features, virtualization features, upstream/downstream network latency, a
- *Edge resources:* Consider configurations such as Edge device (Raspberry Pi 3, UDOO board, esp8266) hardware features (for example, CPU power, main memory size, storage size) , upstream/downstream network latency, supported virtualization features, and so on. Above diverse configuration space coupled with conflicting (trade-off) QoS and SLA requirements leads to exponential growth of potential search space. At the mapping stage, orchestration platform needs to utilise scheduling resource allocation techniques that can allow selection of optimal platform (BigData frameworks) and infrastructure (cloud or edge) configurations for given different workflow components. These techniques also need to consider QoS or SLA requirements such as deployment costs, response time, data processing speed, security level specified by decision makers depending on the application context. These constraints make the mapping problem of each workflow activity to BigData programming framework and datacenter layers NP-Complete. The mapping problem can be easily deducted toto a 0-1 Knapsack or bin-packing problem depending on the constraints given by the decision maker and/or owner.

Workflow QoS monitoring: After the deployment of BigData workflow applications it is important to monitor the run-time QoS and data flow across each activity in the graph, so that administrators and developers can track how application is performing. Much of the difficulty in QoS monitoring from the inherent scale and complexity of BigData workflow application. The problem is complicated because QoS metrics for workflow activities, BigData frameworks, and cloud/edge resources, are not necessarily the same. For example, key QoS metrics are i) event detection and decision making delay for sensor data analysis activity; ii) tweet classification delay and accuracy for Tweet Analysis activity; iii) throughput and latency in distributed data ingestion frameworks (Apache Kafka), iii) response time in batch processing frameworks (Apache Hadoop), (iv) read/write latency and throughput for distributed file system frameworks (for instance, Hadoop Distributed File system); v) server utilization, throughput, and energy-efficiency for cloud resources; and (vi) network stability, throughput optimality, routing delays, fairness in resource sharing, available bandwidth, etc. for the Edge resources.

Therefore it is not clear how i) these QoS metrics could be defined and formulated coherently across workflow activities, BigData programming frameworks, and/or cloud/edge resources and ii) the various QoS metrics should be combined to give a holistic view of data analysis flows. Moreover, to ensure workflow-level performance SLAs we must also monitor workload input metrics (data volume, data velocity, data variety and sources, types and mix of analytics queries) across diverse workflow activities.

Workflow dynamic reconfiguration: The dynamic reconfiguration of BigData workflows in the complex computing infrastructure (Cloud + Edge + multiple BigData frameworks) is complex research problem due to following run-time QoS prediction modelling uncertainties: 1) it is difficult to estimate activity-specific data flow behaviours in terms of data volume to be analysed, data velocity, data processing time distributions, and I/O system behaviour and 2) without knowing the run-time changes to the flow it is difficult to make decisions about the configuration of BigData programming frameworks, cloud and edge resources to be orchestrated so that QoS targets across activities and workflow as whole are constantly achieved; 3) it is difficult to detect causes of QoS anomalies across the complex computing infrastructure due to heterogeneous data flow and QoS measures across multiple workflow activities and the availability, load, and throughput of cloud and/or edge resources can vary unpredictably due to failure or congestion of network links. For example, in Figure 1, velocity of flooding related tweets can increase or decrease based on extent severity of the monsoon. Similarly, during rain gauge sensors can be instrumented to transmit information at much higher velocity and volume during monsoon.

Current State of the art

In this section, we will discuss the current state of the art with respect to the four orchestration challenges in terms of workflow composition, mapping, QoS monitoring, and dynamic reconfigurationto understand to what degree they are able to meet the new end-to-end QoS and SLA requirements of BigData workflow applications.

Workflow composition: Existing orchestation platform such as Apache Oozie and Linkedin Azkaban supports composition of workflows, which can include multiple batch processing activities hence, does not suit the composition needs of complex workflows such as RTFM (see Figure 1) and others. On the other hand, platforms such as Apache YARN, Apache Mesos, Amazon IoT and Google Cloud Dataflow can support script-based composition of heterogeneous analytic activities on cloud datacenter resources cannot deal with Edge resources. Another example of applying analytical techniques for composing BigData applications is the performance analysis of QoS models based on queuing networks and stochastic Petri nets as mentioned by Ardagna and colleagues.[3] Other works aimed at analysing the Map Reduce paradigm using stochastic Petri nets as well as process algebras and Markov chains are.[4,5] Development like these tend to be greatly focused on a single programming paradigm, in this case Map Reduce (batch processing), and are therefore cannot be easily extended to multiple BigData programming frameworks and heterogeneous computing environments (Cloud + Edge). Workflow modelling and deployment specification frameworks and languages such as TOSCA,[2] OPENSTACK Heat, AWS Cloud Formation template and WS-CDL[6] can assist in web services based workflows for software components and Cloud service. However, BigData workflows are quite complex as each analytical activity itself is a workflow in itself. Moreover, to support decision making process, workflow specification should integrate contextual information, which can be dynamically edited by decision maker.

Workflow mapping: Existing BigData workflow orchestration platforms (Apache YARN, Mesos, and Apache Spark) are designed for homogeneous clusters of cloud resources (agnostic to Edge resources). These orchestrators expect workflow administrators to determine the number and configuration of allocated cloud resource types and provide appropriate software-level configuration parameters for each BigData programing frameworks to which one or more analytic activities are mapped to. Branded price calculators are available from public cloud providers (Amazon, Azure) and academic projects (Cloudrado), which allow comparison of cloud resource leasing costs. However, these calculators cannot recommend or compare configurations across BigData processing frameworks driven diverse QoS measures across workflow activities. In a narrow domain, recent efforts[7-10] have attempted to automate the configuration selection of Hadoop frameworks (batch processing) over heterogeneous cloud-based virtualized hardware resources. Multiple approaches[11] have applied optimization and performance measurement techniques for mapping web applications to cloud by selecting optimal virtual machine configuration

> After the deployment of BigData workflow applications it is important to monitor the run-time QoS and data flow across each activity in the graph.

(CPU Speed, RAM Size, cloud location, etc.) based on diverse QoS requirements (throughput, availability, cost, reputation, etc.). However, the configuration space, QoS, and SLA requirements for mapping workflow activities to BigData programming frameworks and cloud/edge resources is fundamentally different from selecting virtual machine configuration for web applications.

Workflow QoS monitoring: BigData Cluster-wide monitoring frameworks (Nagios, Ganglia, Apache Chukwa, Sematex, DMon, SequenceIQ) provide information about QoS metrics (cluster utilization, CPU utilization, memory utilization and nature of application: disk-, network-, or CPU-bound) of virtualized resources that may belong to public or private cloud. These monitoring frameworks[12] do not support workflow activity-level QoS metrics and/or SLAs, which is essential for BigData workflows where change in processing capability of one analytical activity can affect all the activities in the downstream. In the public cloud computing space, monitoring frameworks (Amazon CloudWatch used by Amazon Elastic Map Reduce) typically monitor cloud (agnostic to Edge) VM resource as a black box, and so cannot monitor activity-level QoS metrics and/or data flow. Techniques presented by Alhamazani and colleagues[13] and frameworks such as Monitis[14] and Nimsoft[15] can monitor QoS metrics of web applications hosted on the cloud. Complex event processing and content-based routing applications hosted on clouds. In summary, none of the existing QoS monitoring frameworks and techniques can (i) monitor and integrate data (workload input and performance metrics, disruptive events, SLAs at the platform level, SLAs at the infrastructure) across each activity of the workflow running on multiple BigData processing frameworks and underlying hardware (Cloud + Edge) resources or (ii) detect root causes of workflow activity-level SLA violations and failures across the multiple BigData processing frameworks and hardware resources based on data flow and QoS metrics logs.

Workflow dynamic reconfiguration: Current generation BigData orchestration platforms (YARN, Mesos, Amazon EMR) offer no guarantees about handling failures at workflow-level and/or resource level, nor can they automatically scale or de-scale the platform in response to changes in data volume, velocity or variety, or query types, which can affect the resource requirements of activities within a BigData workflow. There are very few current research works that are trying to address the automatic scaling of single BigData processing framework, batch processing[16] and stream processing.[17] Database community have mostly worked on optimising the query execution performance considering both interleaved18,19 and parallel executions[20,21] via both black-box approaches such online and offline machine learning and white-box approaches for analytical modelling of SQL and/or NoSQL BigData processing frameworks. Existing orchestrators in cloud community that can do online or dynamic reconfiguration have been built specifically for interactive multi-tier web applications.[4,5] However, most of the techniques utilised by them cannot be directly applied to predict data flow metrics (data volume, data velocity, stream operator processing time distributions, query types) or workflow activity-specific QoS metrics (batch processing response time, stream processing latency, data ingestion latency, Tweet analysis accuracy) as BigData workflows are fundamentally different from multi-tier web applications. To make dynamic reconfiguration in the execution of BigData workflow applications, their run-time resource requirements and data flow changes needs to be predicted including any possible failure occurrence. These requirements need to be computed based on inter and intra dataflow of the workflows but also on the user's contextual requirements.

As the concluding remark, current BigData analysis tools and workflow management orchestrators have to evolve to great degree before they can support the requirements of domain-specific BigData workflow applications. Most of these workflows applications are not just monolithic solution but a complex interaction of several BigData programming frameworks, multiple data sources, and heterogeneous Cloud/Edge resources. Each of these applications need to orchestrated to support real time requirements of decision makers expressed in terms of Service Level Agreements.

No prior work has developed workload and resource performance models to enable contention-free scaling and de-scaling of BigData processing frameworks and hardware (Cloud+Edge) resources. In other words, there is no support for new generationBigData workflows' requirements particularly for time-sensitive ones (that is, no workflows, no dynamic orchestration of existing and new data analysis steps, no (Cloud+Edge)-based implementation, and no dynamic tuning of such implementations to meet the owner's decision making requirements), or considers only solutions consisting of data analysis workflows that have predictable performance, which is assumed to be sufficient for its owners (that is, existing research ignores the complexities of cloud and edge resource management for data analysis workflows and does not deal with meeting performance targets as determined by owner's requirements).

Therefore, it is essential that future research consider (1) BigData workflow analysis solutions based on data-driven workflows, (2) mapping such workflows to BigData programming frameworks and Cloud/Edge resources, and (3) manage such mappings and resources to meet specific owner's requirements (or contexts). More specifically, the research community must aim to design new frameworks and novel platforms and techniques that enable decision making by allowing the orchestration of their execution in a seamless manner allowing dynamic resource reconfiguration at runtime.

Acknowledgement

The authors would like thank Mazin Yousif (EIC, *IEEE Cloud Computing*) and Ivona Brandic (TU Wien, Austria) for their valuable feedback, which significantly contributed to improving quality of this column.

References

1. R. Ranjan, B. Benatallah, S. Dustdar and M. P. Papazoglou, "Cloud Resource Orchestration Programming: Overview, Issues, and Directions," *IEEE Internet Computing*, vol. 19, no. 5, 2015, pp. 46–56; doi: 10.1109/MIC.2015.20.
2. Mendling, J., & Hafner, M. (2005). From Inter-organizational Workflows to Process Execution: Generating BPEL from WS-CDL. *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshop*, Springer, 2005.
3. D. Ardagna et al., "Modeling Performance of Hadoop Applications: A Journey from Queueing Networks to Stochastic Well Formed Nets," *Algorithms and Architectures for Parallel Processing - 16th International Conf.* (ICA3PP 16), 2016, pp. 599–613.
4. A. Castiglione et al., "Exploiting mean field analysis to model performances of big data architectures," *Future Generation Comp. Syst.*, vol. 37, 2014, pp.203–211.
5. D Bruneo et al., Analytical Modeling of Reactive Autonomic Management Techniques in IaaS Clouds. *Proc. 2015 IEEE 8th International Conf. on Cloud Computing* (CLOUD 15), 2015, pp. 797–804.
6. T. Binz et al., "Portable Cloud Services Using Tosca," IEEE Internet Computing, vol. 16, no. 3, pp. 80–85.
7. H. Herodotou et al., "A What-if Engine for Cost-based Mapreduce Optimisation," *IEEE Data Engineering Bulletin*, vol. 36, no. 1, 2013, pp. 5–14.
8. G. Lee, B. Chun, and H. K. Randy, "Heterogeneity-Aware Resource Allocation and Scheduling in the Cloud," *Proc. 3rd USENIX Conf. on Hot topics in cloud computing* (HotCloud 11), 2011, p. 4.
9. P. Lama and X. Zhou, "AROMA: Automated Resource Allocation and Configuration of MapReduce Environment in the Cloud," *Proc. 9th International Conference on Autonomic Computing* (ICAC 12), 2012, pp. 63–72.
10. K. Kambatla et al., "Towards Optimising Hadoop Provisioning in the Cloud," *Proc. 2009 Conf. on Hot topics in cloud computing* (HotCloud 09), 2019, Article 22.
11. R. Ranjan, J. Kolodziej, L. Wang and A. Y. Zomaya, "Cross-Layer Cloud Resource Configuration Selection in the Big Data Era," *IEEE Cloud Computing*, vol. 2, no. 3, 2015, pp. 16–22;. doi: 10.1109/MCC.2015.64
12. K. Alhamazani, R. Ranjan et al., "An Overview of the Commercial Cloud Monitoring Tools: research dimensions, design issues, and state-of-the-art," *Springer Journal of Computing*; doi: 10.1007/s00607-014-0398-5.
13. K. Alhamazani, R. Ranjan et al., "CLAMS: Cross-Layer Multi-Cloud Application Monitoring-as-a-Service Framework," *Proc. of the 11th IEEE Int. Conf. on Services Computing* (SCC 14), 2014 pp. 283–290.

This article originally appeared in IEEE Cloud Computing, *vol. 4, no. 3, 2017.*

14. TeamViewer, "Monitis," Feb. 2015; http://portal.monitis.com/.
15. CA Technologies "Proactively enure experience of cloud and hybrid IT infrastructures," Feb. 2015; https://www.ca.com/us/products/ca-unified-infrastructure-management.html
16. A. Abouzeid et al., "Hadoopdb: An Architectural Hybrid of Mapreduce and DBMS technologies for Analytical Workloads," *Proc. of the VLDB Endowment*, (PVLDB 09), 2009, pp. 925–933, 2009.
17. R.CastroFernandez, et al., "Integrating Scale Out and Fault Tolerance in Stream Processing using Operator State Management," *Proc. of the International Conf. on Management of Data* (SIGMOD 13), 2013, pp. 725–736.
18. A. Ganapathi et al., "Predicting Multiple Metrics for Queries: Better Decisions Enabled by Machine Learning", *Proc. of the IEEE International Conference on Data Engineering* (ICDE 2009), 2009, pp. 592-603.
19. M. Akdere et al., "Learning-based Query Performance Modeling and Prediction," *Proc. of the 2012 IEEE International Conf. on Date Engineering* ICDE 2012, 2012, pp. 395–401.
20. J. Duggan et al.,"Performance Prediction for Concurrent Database Workloads," *Proc. Of the International Conf. on Management of Data* (SIGMOD 11), 2011, pp. 337–348.
21. W. Wu et al., "Towards Predicting Query Execution Time for Concurrent and Dynamic Database Workloads," *Proc. of the VLDB Endowment* (PVLDB 12), 2012, pp. 925–936.

ADVERTISER INFORMATION

Advertising Personnel

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

Advertising Sales Representatives (display)

Central, Northwest, Southeast, Far East:
Eric Kincaid
Email: e.kincaid@computer.org
Phone: +1 214 673 3742
Fax: +1 888 886 8599

Northeast, Midwest, Europe, Middle East:
David Schissler
Email: d.schissler@computer.org
Phone: +1 508 394 4026
Fax: +1 508 394 1707

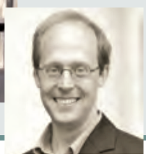
Southwest, California:
Mike Hughes
Email: mikehughes@computer.org
Phone: +1 805 529 6790

Advertising Sales Representative (Classifieds & Jobs Board)

Heather Buonadies
Email: h.buonadies@computer.org
Phone: +1 201 887 1703

Advertising Sales Representative (Jobs Board)

Marie Thompson
Email: marie@4caradio.org
Phone: 714-813-5094

Editors: Jesús Favela ■ CICESE ■ favela@cicese.mx
Stephen Intille ■ Northeastern University ■ s.intille@neu.edu

Population-Scale Pervasive Health

Tim Althoff, *Stanford University*

Our everyday behavior is critical to our health. An estimated 60 percent of the human health condition is determined by behavioral factors—including exercise, sleep, and diet—as well as social and environmental factors.[1] Historically, these behavioral, social, and environmental factors have been difficult to measure and quantify. Scientists and clinicians typically relied on (guided) self-reports, which are subjective and often biased. Additionally, the assessments have been typically limited to participants recalling static health information (such as their weight or health conditions) and summarizing health-related behaviors and activities over limited time periods. These measurement limitations have led to reduced ecological validity and often a sparseness of data about highly dynamic health-related behaviors, such as physical activity.

Pervasive computing and pervasive health research could transform this landscape and fill in the measurement "gaps." Ubiquitous sensors, both in the environment and in our personal devices, clothing, and bodies, can continuously collect data, allowing for dynamic measurement and more nuanced and robust investigation of health-related behaviors, activities, and physiological signals from our bodies. Connecting this data to health outcomes could unveil a great deal of which behaviors are predictive of, or even causally responsible for, our well-being.

In particular, *population-scale pervasive health* research attempts to harness such data, which has already been collected through commercial devices and web applications (see the "Ubiquitous Sensing" sidebar), to study human behaviors and the links between that data and health and well-being. Leveraging these existing datasets enables studies of behaviors and health at an unprecedented scale (in terms of number of subjects), resolution (regarding the number and granularity of activities tracked), and duration (length of observation period) relatively inexpensively and quickly.

Population-scale pervasive health research can complement more

UBIQUITOUS SENSING

Current commercial mobile and wearable devices include many of the sensors and techniques used in health research (including accelerometer, gyroscope, location, heart-rate, and skin-conductance sensors and activity classification algorithms, smart notifications, and report capabilities). Furthermore, many smartphone apps let the user self-report activities and conditions that are challenging to capture automatically, such as the consumption of food, alcohol, and caffeine or the user's emotional status. Today, smartphones are used by 69 percent of the adult population in developed countries and 46 percent of the adult population in developing economies, with adoption rates growing rapidly.[1] Social media posts, as well as web search queries, can also reveal a great amount about individuals' behaviors, health, and well-being.[2] For example, individuals share and discuss goals, behaviors, sicknesses, diagnoses, and mental health challenges on platforms including Facebook, Reddit, Twitter, and Instagram.[2–5]

REFERENCES

1. E. Anthes, "Mental Health: There's an App for That," *Nature*, vol. 532, no. 7597, 2016, pp. 20–23.
2. E. Yom-Tov, *Crowdsourced Health: How What You Do on the Internet Will Improve Medicine*, MIT Press, 2016.
3. T. Althoff, K. Clark, and J. Leskovec, "Large-Scale Analysis of Counseling Conversations: An Application of Natural Language Processing to Mental Health," *Trans. Assoc. Computational Linguistics*, 2016, pp. 463–476.
4. M. De Choudhury et al., "Discovering Shifts to Suicidal Ideation from Mental Health Content in Social Media," *Proc. 2016 CHI Conf. Human Factors in Computing Systems*, 2016, pp. 2098–2110.
5. F. Ofli et al., "Is Saki #Delicious? The Food Perception Gap on Instagram and Its Relation to Health," *Proc. 26th Int'l Conf. World Wide Web*, 2017, pp. 509–518.

2469-7087/18/$33.00

PERVASIVE HEALTH

traditional pervasive computing and pervasive health research by highlighting the possibilities, opportunities, and challenges that arise from analyzing behavior and health data at scale. Here, I identify lessons learned from my own work and from other excellent contributions to the field and current challenges in this research area.

LESSONS LEARNED

Although there are great advantages in leveraging large-scale datasets for individual and population health, there are limitations to this approach as well. Unlike typical "in-person" experimental studies or cohort studies, researchers often have limited or no control over what data is collected or how it has been collected. Furthermore, the data is typically observational and thus without any randomization into different conditions or treatments. This makes identification of causal relationships fundamentally challenging.

To overcome these challenges, interdisciplinary teams of researchers are developing specialized computational methods and tools, often drawing on data mining, social network analysis, and causal inference research, with the aim of obtaining unconfounded and actionable conclusions that are necessary to make any impact on healthcare and public policy. Following are lessons learned from leveraging existing devices and data for population-scale pervasive health research.

The Power of Scale

Leveraging widely used devices and applications allows for studies at unprecedented scale, much beyond what is feasible in more traditional laboratory or cohort settings. For example, leveraging data collected through a popular smartphone activity-tracking application (Argus by Azumio), I worked with Rok Sosič, Jennifer Hicks, Abby King, Scott Delp, and Jure Leskovec to study and compare physical activity patterns at a planetary scale across 717,000 people from over 110 countries.[2] The scale of such datasets enables new insights based on comparisons across subpopulations of various kinds—for example, based on demographic attributes including gender, age, weight status, and country of origin.

In the past, many of these comparisons were extremely expensive if not infeasible. For example, until recently, we had no large-scale database of objective physical activity measures spanning multiple countries. However, using the data from Argus, we were able to estimate the distribution of physical activity levels within countries, on a global scale.[2] This analysis, for the first time, revealed patterns of worldwide activity inequality. In some countries the gap between "activity rich" and "activity poor" people was much larger than in other countries, and the size of this gap was found to be a strong predictor of obesity incidence in the respective countries. Furthermore, it had been well established that men tend to be more physically active than women, on average. However, our analyses revealed that in countries such as Sweden and Ukraine the gender gap was almost negligible, whereas in countries such as Saudi Arabia and the US the gap was substantial. Such gaps could have detrimental consequences for women's health.

The large number of people tracking their exercise through a wearable device further enables us to study population-scale phenomena, such as the viral spread of Pokémon Go. Because many wearable users of devices such as the Microsoft Band have agreed to share their data for research, Ryen White, Eric Horvitz, and I were able to conduct a study of Pokémon Go's impact on 83,000 people's physical activity (without this data, such a study could have required a massive participant recruiting and data collection enterprise).[3] We found that playing Pokémon Go led to significant increases in physical activity over a period of 30 days, with particularly engaged users increasing their activity by 1,473 steps a day on average, a more than 25 percent increase compared with their prior activity level. Although these activity increases were often short-lived, Pokémon Go was able to reach low-activity populations—whereas four leading mobile health apps we looked at for comparison largely drew from an already very active population.

Natural Experiments and Causal Inference

A fundamental challenge with observational (that is, non-experimental) studies is using correlational data to infer causality. Understanding the causes of health outcomes is necessary for improving as opposed to merely predicting such outcomes. Accurately inferring causality from observational data is difficult because different conditions likely were not assigned randomly, leading to confounding and biased estimates.

To circumvent this, researchers increasingly apply matching methods, which match every treated unit to one or more non-treated units with similar observable characteristics (exact, almost-exact, distance- or propensity-score-based).[4,5] By matching treated units to similar, non-treated units, matching enables a comparison of outcomes among treated and non-treated units to estimate the effect of the treatment-reducing bias due to confounding.

However, there might exist unobserved variables that we cannot control for, even when this would be desirable. Although sensitivity analyses can alleviate such concerns in part, it might be more promising to attempt to identify variation in the data that could be used as an instrument or natural experiment to overcome the limitations of observational data. For example, researchers can leverage weather variation, changes in built environments due to relocation, or other potentially exogenous events such as strikes of public transport workers or closings of parks, to disentangle various effects. In some of my own work on the social influence effects on exercising behavior, we leveraged random variation capturing the delay in the formation of friendship connections (how long did it take for the receiver of a friendship request to press "accept"?) to disentangle intrinsic motivation to exercise more from actual social influence.[4] This allowed us to demonstrate a causal, positive effect of online friendship connections on offline physical activity.

Large-scale data facilitates the use of such methods. For example, events such as user relocation or strikes are quite rare. However, in large data, we might still observe enough of these events to enable statistically meaningful analyses. Identifying these instruments and natural experiments in large data typically requires a mix of domain expertise (for example, knowing what events would cause plausibly exogenous variation in the treatment assignment) as well as data mining methods.

Population Bias

User populations of wearable and tracking devices, smartphones, and web applications might not be representative of national populations, even when they are very large. In fact, we have found that the users of popular wearables and tracking apps tend to be biased toward young, more affluent, gender-skewed populations.[2–4,6,7] For example, some of the datasets from fitness wearables we have studied were skewed towards male users, while mobile applications focused on weight loss were skewed towards a female user base.

Researchers can check their findings for robustness against these biases—for example, by stratification into subgroups (for example, by age, gender, and income level)[2–4,7] or by reweighting the sample to match a target population.[2] In fact, as long as the data provides sufficient support for all relevant subgroups, reweighting methods can approximate nationally representative populations. Being able to validate findings across many subgroups to investigate heterogeneous treatment effects can be an advantage over representative yet small study populations (which might be statistically underpowered for such analyses).

The study population can also be compared to traditional medical research data—for example, data from the National Health and Nutrition Examination Survey (www.cdc.gov/nchs/nhanes) or from the World Health Organization's Global Health Observatory (www.who.int/gho/en)—on key behavioral or health covariates such as the timing or length of sleep or the volume and intensity of physical activity. Such data is often available on a subpopulation level, but it's often based on subjective survey measures, which could be vastly different from sensor-defined objective measures, limiting comparisons.

As smartphones and other mobile or wearable devices become more prevalent, we can expect population bias to decrease. Furthermore, this drawback is not specific to population-scale pervasive health studies; it also applies to all scientific studies that largely draw from WEIRD (Western, Educated, Industrialized, Rich, and Democratic) subjects. Leveraging widely used devices and applications might even help in understanding historically less-represented populations.

Engagement and Retention

Participant engagement and retention often diminish quickly. This holds true in traditional in-lab or in-person studies but can be exacerbated in commercial device and app settings where subjects are unpaid, are not bound by a study protocol, and quickly move on when they do not perceive a clear value. These dynamics are observed widely and were at the heart of many articles about Pokémon Go. The mobile game was spectacularly successful with 28.5 million daily users shortly after release, but after a few months, approximately 80 percent of them had moved on.[8] These numbers highlight both the unprecedented promise of large-scale behavioral interventions and great challenges in retaining engagement levels.[3]

An important consequence of low user retention rates is that researchers studying the same application at different times may in fact study different user populations. These populations are all worth studying. For example, early adopters are key to developing early prototypes into mature applications, while the coming and going of short-term users may help us better understand the appeal and value propositions of our applications. Furthermore, even after a historical drop-off in engagement, millions of people around the world still play Pokémon Go, allowing us to better understand whether and when such games could lead to sustained behavior change.[3,7]

Multisite Studies to Disentangle Individual Behavior and the Environment

Many existing mobile health and social media datasets cover a large number of geographical locations.[9] Some even have global coverage.[2] This means that users of the same device or application can reside in vastly different environments. While this can complicate comparative analyses between the users, it also creates an unprecedented opportunity to study the effect of different environments on human behavior. To what degree is an individual's behavior truly individual or dictated by one's environment?

For example, cities without prevalent and safe sidewalks and footpaths or close-by stores, schools, and parks make it much harder to be physically active. The multisite nature of large-scale datasets let us disentangle individual behavior and environmental influences (requiring adequate control of potential confounders), and might enable us to design cities more conducive to their inhabitants' health.[2]

Augmenting Large Sensor Data with Context

Although continuous sensor data paints a rich picture of our behaviors, much of this data is useless without context. If somebody records very few

PERVASIVE HEALTH

steps on a given day, is it because of the person's age, weight, recent surgery, non-walkable neighborhood, scorching climate, or preference for other activities? While smaller, in-person studies let us collect such information through surveys and other measures, much of this critical context is missing in already-collected, large-scale datasets from phones and wearables. Thus, to unleash the true power of these datasets, researchers need to augment the data with contextual information.

A common approach for data augmentation is to leverage an individual's geolocation, often available through self-report, GPS, cell tower location, and IP addresses. This enables augmenting the sensor data with population-level census or health outcome data.

Beyond location, researchers have brought in valuable context to sensor data through combinations with web search and online social network data. For example, Web search queries allowed differentiation of Pokémon Go players from non-players in a large sample of wearable users.[3] In another study, web search queries enabled the non-intrusive measurement of cognitive performance from already-collected search query logs, which could then be related to wearable-based sleep measurements.[6] In both studies, users had connected their web search account (Bing) to their wearable device (Microsoft Band) and agreed to share their data for research purposes. Combinations with online social network data enabled estimation of social influence effects in exercising behavior—that is, whether exposure to and interactions with online friends would have an effect on someone's physical activity levels (spoiler alert: it did!).[4,7] Social network data might be available if the network is part of the mobile application itself, when the application imports external social network data (from Facebook, for example), or when users share their activities and behaviors on public sites such as Twitter, Instagram, or Reddit.[5,9]

RESEARCH OPPORTUNITIES

There are several research directions for increasing the effectiveness and scope of population-scale pervasive health research. These relate to developing more sophisticated computational methodology, acting on inferences made, and data sharing.

Improving Computational Tools

Large-scale behavioral data paired with powerful computational tools bring unprecedented affordances, many of which we likely have yet to uncover. Open research challenges include identification of useful signals and proxies in sensor and Web data to capture behaviors, relevant context, and outcomes. Several studies have used social media posts, messages, and badges as proxies for health outcomes including weight loss, depression, and suicidal thoughts.[5,10] Such proxies are potentially powerful, but we need to better understand how to appropriately identify them and to what degree they reflect clinical measures.

Many future datasets will be observational. Due to the large but uncontrolled nature of the data, it is easy to fall prey to spurious correlations (with very low p-values). Therefore, we need to develop improved computational tools for analysis as well as establish methods and protocols abiding by the highest scientific standards. This is particularly pressing in the realm of causal analyses for which, currently, few if any tools exist that are usable by non-experts. Furthermore, such tools should offer scalability and capabilities beyond simple binary treatment/control scenarios and low-dimensional covariate spaces.

Acting on Inferences Made

The goal of population-scale pervasive health research is to translate inferences from big data into the real world to improve people's lives. This is particularly challenging for data mining researchers who can be far removed from the people they study.

Real-world impact ranges across many settings, including clinical and population health, city and community design, and commercial devices and applications. Many of the datasets currently studied are collected by large technology companies. These companies might be receptive to suggestions that will improve the lives of their users, and they often have the resources to translate ideas and prototypes into practice. However, without support of these companies (for example, due to misaligned incentives or goals or a lack of resources), it can be difficult to act on any inferences made. Still, there are great opportunities to explicitly design the online space including search and social network tools for improved health and well-being.[4–7,9,10]

However, significant challenges exist in moving from passive sensing to acting on data inferences and interacting with users. One might be able to detect traces of depression and self-harm,[5,10] cyberchondria, or fatigue levels severely increasing accident risk. But how should one act on this information? If a user did not ask for it explicitly, it might be inappropriate or unethical to bring up this information. Furthermore, well-being objectives might even conflict with short-term commercial interests of technology companies (for example, some social media posts might drive significant engagement but could negatively influence self-worth and depression). In all these cases, user experience and ethics research are vital for the appropriate implementation and complementation of big data studies.

Data Sharing, Privacy, and Ethics

Scientific progress in population-scale pervasive health studies critically relies on the availability of data. Currently, researchers often gain access to large datasets through industry collaborations or through web scraping of social media sites. Although many of these collaborations have been fruitful, data is rarely shared with outside researchers, limiting reproducibility and scientific progress.

More data sharing is clearly desirable, but there are many challenges to share personal information about behaviors and health.[11] For example, there is a need for methods and best practices to successfully de-identify data without introducing noise or bias into statistical analyses. In many cases anonymization techniques are open to reidentification attacks. There are promising avenues in privacy research based on randomized responses, differential privacy, and homomorphic encryption, but these techniques are not yet used widely in practice or at scale.

Currently, behavioral and health data are largely kept in separate silos. Behavioral signals are predominantly in the hands of tech companies developing wearable devices and smartphone applications while health data typically resides within hospital insurance systems. Great value will come from developing appropriate processes to combine this data to better understand how complex behavioral patterns cause or are caused by specific health outcomes. Large cohort studies hold great promise—such as the All of Us research program (https://allofus.nih.gov), which seeks to collect and combine such data from over one million US volunteers. It will be important to figure out how people's existing devices can be leveraged effectively in these studies, and how researchers can provide valuable insights or financial incentives to motivate participants to share their personal data.

There are great opportunities for cross-pollination between pervasive computing and population-scale pervasive computing research. For example, laboratory-based and in-person pervasive computing research can inform data mining researchers about what can be measured and correlated in more controlled settings, highlighting the human factors at play. Conversely, large-scale data mining can inform pervasive computing about what types of inferences are possible, informing our understanding of potential public health applications of pervasive health technology at scale. Furthermore, population-scale pervasive computing might change how epidemiological and population health research is conducted by enabling continuous, objective measurements of dynamic behaviors and environmental factors.

Advances and proliferation of mobile and sensor technology are driving the creation of large behavioral datasets. Population-scale pervasive health research leverages these datasets to enable a better understanding of the relationships between our behaviors, environment, and health outcomes, with great opportunities to impact health and well-being at population-scale.

ACKNOWLEDGMENTS

I would like to thank Stephen Intille, Jesus Favela, Michele Catasta, Emma Pierson, and Rok Sosič for their feedback on the manuscript, and Jure Leskovec, Rok Sosič, Eric Horvitz, Ryen White, Scott Delp, Jen Hicks, Abby King, Munmun de Choudhury, Emre Kiciman, Elad Yom-Tov, Ingmar Weber, and Yelena Mejova for countless enlightening discussions. This work was supported by a SAP Stanford Graduate Fellowship and the NIH BD2K Mobilize Center (U54 EB020405).

REFERENCES

1. J.M. McGinnis, P. Williams-Russo, and J.R. Knickman, "The Case for More Active Policy Attention to Health Promotion," *Health Affairs*, vol. 21, no. 2, 2002, pp. 78–93.
2. T. Althoff et al., "Large-Scale Physical Activity Data Reveal Worldwide Activity Inequality," *Nature*, vol. 547, no. 7663, 2017, pp. 336–339.
3. T. Althoff, R.W. White, and E. Horvitz, "Influence of Pokémon Go on Physical Activity: Study and Implications," *J. Medical Internet Research*, Dec. 2016, p. e315.
4. T. Althoff, P. Jindal, and J. Leskovec, "Online Actions with Offline Impact: How Online Social Networks Influence Online and Offline User Behavior," *Proc. 10th ACM Int'l Conf. Web Search and Data Mining*, 2017, pp. 537–546.
5. M. De Choudhury et al., "Discovering Shifts to Suicidal Ideation from Mental Health Content in Social Media," *Proc. 2016 CHI Conf. Human Factors in Computing Systems*, 2016, pp. 2098–2110.
6. T. Althoff et al., "Harnessing the Web for Population-Scale Physiological Sensing: A Case Study of Sleep and Performance," *Proc. 26th Int'l Conf. World Wide Web*, 2017, pp. 113–122.
7. A. Shameli et al., "How Gamification Affects Physical Activity: Large-Scale Analysis of Walking Challenges in a Mobile Application," *Proc. 26th Int'l Conf. World Wide Web Companion*, 2017, pp. 455–463.
8. S. Arif "The Number of Pokémon Go Players Still Logging in Every Day Has Plummeted by over 80%," *VG 24/7*, 3 Apr. 2017; www.vg247.com/2017/04/03/the-number-of-pokemon-go-players-still-logging-in-every-day-has-plummeted-by-over-80.
9. F. Ofli et al., "Is Saki #Delicious? The Food Perception Gap on Instagram and Its Relation to Health," *Proc. 26th Int'l Conf. World Wide Web*, 2017, pp. 509–518.
10. T. Althoff, K. Clark, and J. Leskovec, "Large-Scale Analysis of Counseling Conversations: An Application of Natural Language Processing to Mental Health," *Trans. Assoc. Computational Linguistics*, 2016, pp. 463–476.
11. K. Caine, "Privacy Is Healthy," *IEEE Pervasive Computing*, vol. 15, no. 4, 2016, pp. 14–19.

Tim Althoff is a PhD candidate in computer science at Stanford University. Contact him at althoff@cs.stanford.edu.

This article originally appeared in IEEE Pervasive Computing, *vol. 16, no. 4, 2017.*

Editor: Jimmy Lin, jimmylin@uwaterloo.ca

In Defense of MapReduce

Jimmy Lin • University of Waterloo

Friends, developers, researchers, lend me your ears! I come to praise MapReduce, not to bury it!

Google Fellow Luiz Barroso once famously said, "The datacenter is now the computer." David Patterson[1] then wondered: What's the instruction set architecture for the datacenter computer? This was circa 2008, and of course by then we already knew the first answer: for MapReduce,[2] the two instructions are MAP and REDUCE.

MapReduce represents a specific instance of a general class of data-parallel dataflow languages, in which computations are conceptualized as directed graphs, where vertices represent operations on records that flow along the directed edges. From this perspective, MAP and REDUCE are the two operators that MapReduce provides, which define particular configurations of the edges that flow into and out of vertices and specify the computations that occur at the vertices themselves.

Such a dataflow computation model dates back to the 1970s,[3,4] but there's one key difference: in today's conception of dataflow languages, the focus is on data-parallelism. With perhaps the exception of sophisticated machine learning algorithms, the dataflow graphs today themselves are relatively simple. The challenges primarily lie in the requirement to process gazillions of records, thus necessitating distributed processing across warehouse-scale clusters.

Dryad is usually credited with the development of the first general dataflow model for distributed data-parallel computations.[5] It provides a rich vocabulary of operators that can be composed to form complex dataflow graphs and an execution engine for managing the specified computations. Over time, we've witnessed the development of alternative data-parallel dataflow languages that primarily differ in the vocabulary of operators that they provide to the developer. Examples include Cascading (www.cascading.org), Google's FlumeJava,[6] Apache Flink (https://flink.apache.org), and of course, Apache Spark (https://spark.apache.org), which is perhaps the most widely touted general-purpose replacement for MapReduce.

In this column, I present a critical analysis of the dataflow operators provided by MapReduce and Spark. To be precise, I'll be specifically referring to the Hadoop implementation of MapReduce, which is slightly different from Google's original. My message is this: In the fashion-driven world of big data where there's a perpetual rush toward new and shiny objects along with a tendency to pooh-pooh everything that has come before, let's not throw the MapReduce baby out with the bathwater!

There are aspects of the Hadoop MapReduce API that provide a well-conceived balance between flexibility and expressiveness, even if there are serious shortcomings with the overall implementation. In particular, comparisons with Spark are instructive for highlighting the distinction between logical and physical operators, and point to a gaping hole where MapReduce is incomplete. If we recognize MAP and REDUCE as the physical operators that they really are, then comparing MapReduce to Spark is actually like comparing apples to oranges (quite contrary to popular portrayals by blogs, the tech press, and other superficial discussions).

Before I start getting hate mail from Spark fanboys, let me be perfectly clear: I really like Spark. On the whole, it represents a far superior implementation of MapReduce. Resilient Distributed Datasets (RDDs) and lazy transformations support pipelining, plan rewrites, and other optimizations difficult to implement in Hadoop. Caching of RDDs accelerates iterative algorithms while lineage information provides robustness. Having a read-eval-print loop (REPL) is a godsend. Language bindings beyond the Java Virtual Machine democratize data processing tools to the large community of Python and R users. Overall, Spark most definitely deserves the mantle as the successor to MapReduce.

Big Data Bites

"Big Data Bites" is a regular department in *IEEE Internet Computing* that aims to deliver thought-provoking and potentially controversial ideas about all aspects of big data. Interested in contributing? Drop me a line!

—*Jimmy Lin*

But I'm going to complain about the design of some of Spark's dataflow operators anyway. The discussion in the rest of this column is fairly low level and borders on "inside baseball" minutiae, but by design (because that's part of my critique). However, I'll freely admit that this is mostly an academic exercise — although I do think the exercise is instructive in helping us better understand the design of big data analytics platforms in general.

At the end of this article, I'll come back and discuss why none of this particularly matters, and why questions about low-level operator design are inconsequential as big data processing becomes increasingly focused on higher-level abstractions for vertically-informed data manipulation.

The Mappers

Let's start with mappers in MapReduce. In Hadoop, a mapper is instantiated for every partition of the input dataset — more precisely, each input split, which aligns with data blocks on the Hadoop Distributed File System (HDFS) in the canonical case. The mapper lifecycle begins with a `setup` method and ends with a `cleanup` method; in between the `map` method is called for each key-value pair in the input collection. Specifically, the framework initializes a `RecordReader`, iteratively calls the reader to materialize the next key-value pair, and then calls the `map` method in the mapper.

In Spark, there are a few comparable "map" transformations: `filter`, `map`, `flatMap`, and `mapPartitions`. Why do we need all of them? The `map` transformation takes `f: T ⇒ U` to transform inputs of type `T` to outputs of type `U`. Importantly, `map` in Spark produces exactly one output per input and thus is less flexible than the map method in MapReduce, which can generate zero, one, or more intermediate key-value pairs. This is why Spark additionally needs `filter` (to *not* generate any intermediate records) and `flatMap` (to generate a list of intermediate records, which the framework then flattens). Finally, `mapPartitions` is needed to provide the equivalent of what `setup` and `cleanup` do in MapReduce: `mapPartitions` takes `f: Iterator[T] ⇒ Iterator[U]`, which allows developers to sneak in setup and cleanup.

From the perspective of a functional purist, I can argue that the MapReduce abstraction leaks in two ways. First, the mappers encapsulate per-record processing, and the fact that the input collection is divided into partitions: well, that's an implementation detail. Having `setup` and `cleanup`, in effect, hard codes the existence of some partitioning scheme and a particular physical organization. Second, having control over the mapper lifecycle allows the developer to retain state across map calls and engage in monkey business that breaks the functional abstraction.

With respect to the first point: data processing languages must operate in the real world, and in the real world programmers do need to manage their object lifecycles. The most common use for the `setup` method is to acquire an external resource (for example, a database connection) or load in side data (for example, a dictionary). The `cleanup` method is used to release these resources after all the records have been processed. Because it makes no sense to perform these heavyweight operations on a per-record basis, the `filter`, `map`, and `flatMap` transformations in Spark can't handle this common design pattern.

It appears that the developers of Spark realized this fact somewhat later, as `mapPartitions` wasn't in the original Spark API;[7] see Apache JIRA ticket SPARK-341. Thus, both MapReduce and Spark are adulterated in having abstractions that aren't functionally pure. In Spark, there's actually also something called `mapPartitionsWithIndex`, which provides an integer value representing the index of the partition, which I react with a #facepalm (If you don't get the meme, search the web. I especially love Captain Piccard doing it). You don't need it in MapReduce because in the `setup` you have access to the `Context` object, which gives you a lot of reflection-like information about the state of the mapper and the data it's running on.

With respect to the second point — the ability in MapReduce to retain state across map calls and engage in monkey business — this actually allows you to do some neat optimizations that substantially increase performance via more efficient local aggregations, for example, what I've called the "in-mapper combining" pattern.[8,9] I suppose someone might argue that such techniques are hacky and shouldn't be allowed, but then, the same complaint surely would apply to Spark, because it has `mapPartitions`. Whatever sneaky monkey business you can do in MapReduce, you can do in Spark also.

Here's another way to think about it: the mapper in MapReduce is actually a physical operator, while transformations in Spark aim to be logical operators. The conventional understanding is that a physical operator specifies a particular implementation, whereas a logical operator expresses the computation at a more abstract level (for example, relational operators in the case of SQL).

As discussed previously, mappers in MapReduce make explicit the sequence of computations (method calls) that occur while processing a collection of key-value pairs. In fact, in Hadoop you call the `Context.write()` method to actually emit intermediate key-value pairs. The map transformations `filter`, `map`, `flatMap` in Spark, on the other hand, don't tie the framework to any particular implementation. There's no reason, for example, why each input record can't be processed by a separate thread, which might make sense if Spark were implemented in Erlang due to its support for lightweight threads.

But here's where the rubber of clean abstraction design meets the road of real-world constraints: `mapPartitions` and `mapPartitionsWithIndex` in Spark hard code aspects of physical execution, which ruin the elegance of Spark transformations. The ability for developers to manipulate physical operators in MapReduce affords a high degree of flexibility (as is generally the case when physical operators are accessible); this control is ceded for cleaner logical operators in Spark, but unfortunately Spark isn't able to completely deliver on elegance.

The Reducers

Now let's turn our attention to reducers. The reducers in MapReduce have a signature of `g: (T, Iterator[U]) ⇒ Seq[(R, S)]`. That is, the reducer `g` takes an object of type `T` (the intermediate key) and an iterator over values of type `U` (the values associated with that intermediate key), and returns any number of output key-value pairs (of any arbitrary type). In MapReduce there are also combiners, with the same signature. Combiners perform per-key aggregation on the output of the mappers, prior to the network shuffle. In reality, however, Hadoop also sneaks in combiner execution on the reduce end, post-shuffle. And oh, one more detail: the `T`'s are sorted (more on this later).

Similar to the mappers, developers can manage the reducer and combiner lifecycles via the `setup` and `cleanup` API hooks. To accomplish the equivalent in Spark, you'd have to do something like `groupByKey` followed by `mapPartitions`. At that point, you're basically just writing MapReduce in Spark, which is fine — but just don't be hatin' MapReduce. (Not to mention that a straight-up `groupByKey` isn't particularly efficient because it doesn't do map-side aggregation; of course, you can add in map-side aggregation, but then, my original point remains — it's basically back to MapReduce.)

Okay, so in MapReduce we have reducers and combiners, and that's it. Oh, there are also partitioners, which simply divide up the intermediate key space, but you can't get away without having something like that (and Spark has partitioners also). In Spark, there's a number of reduce-like operations (leaving aside joins and cogrouping for now): `groupByKey`, `reduceByKey`, and `aggregateByKey`. What do these all do?

The `reduceByKey` transformation takes `g: V × V ⇒ V`. In other words, `V` forms a commutative monoid with `g` as its associative binary operation (more precisely, a commutative semigroup, because left unspecified is the identity element). Implicit in the semantics of `reduceByKey` is that `g` must be associative and commutative, because otherwise the execution simply wouldn't be correct: Spark's documentation was previously muddled (see SPARK-12844) but it has since been fixed. In contrast, a system like Summingbird renders the algebraic properties of the types explicit.[10]

Leaving issues with precise execution semantics aside, `reduceByKey` is actually quite restrictive, because the input and output types can't change. This is both a plus and a minus. The positive is that Spark can take the function `g` that goes into `reduceByKey` and move it over to the map side, that is, before the network shuffle, in a completely transparent manner. So, there's no need to explicitly write combiners — the framework optimizes for you. The downside is that all the values have to be type `V` and the function must also operate on that type.

What if I want to be a bit more flexible on the output type? This is where `aggregateByKey` comes in: the transformation takes `f: U × V ⇒ U`, `g: U × U ⇒ U`; in other words, `f` allows you to convert from type `V` (the input type) to type `U` (the output type) while performing intermediate aggregation, and `g` specifies how you combine values of type `U` together. With this setup, `f` automatically can be pushed over to the map side for efficient intermediate aggregation.

What if you still can't get all the types to work out correctly? Well, then you're back to `groupByKey` followed by a map-like transformation — get the framework to group together all the values with the same intermediate key for you, and then apply whatever computation you want yourself. At that point, though, the framework can't perform any optimizations behind the scenes, and performance will suffer because of the shuffling of lots of intermediate data across the network.

One way to think about these reduce-like operations in Spark is in terms of the tradeoff between simplicity, flexibility, and performance. With `reduceByKey`, you get simplicity and performance, at the cost of flexibility. With `aggregeteByKey`, you gain a bit of flexibility at the cost of simplicity, but still get good performance. With `groupByKey`, you get simplicity and maximum flexibility, but at the cost of performance. Contrast this with MapReduce, where you only get reducers and combiners, but there isn't anything you fundamentally can't do because you have low-level control over the physical execution. It's not clear that the Spark panoply of reduce-like transformations is easier to understand or use.

What do we see if we compare reducers in MapReduce with reduce-like transformations in Spark through the lens of the logical/physical distinction discussed previously? The reduce in MapReduce most definitely describes a physical operator: keys arrive at the reducer in sorted order, which all but prescribes how the shuffle group-by must be implemented. (In reality, it was closer to the other way around: the framework developers came up with an efficient sort-based shuffle grouping implementation, and then shrugged, "well, we might as well expose this in the API.")

In contrast, the Spark reduce-like transformations leave open the implementation, and indeed Spark can select between a hash-based and a sort-based shuffle scheme. In practice, however, hash-based shuffling suffers from scalability limitations beyond a certain point, and thus sort-based shuffling has been the default since Spark 1.2. So, while reduce-like transformations are nominally logical operators in Spark, they're severely constrained by the practicalities of execution performance. Once again, the rubber of clean design meets the road of implementation realities.

Other Spark Transformations

At this point, I've compared Spark's map-like transformations with mappers in MapReduce and Spark's reduce-like transformations with combiners and reducers in MapReduce, arguing that there's a certain elegance in the simplicity of MapReduce, particularly in providing developer access to physical operators, including the ability to explicitly manage object lifecycles. But Spark has many transformations beyond map-like and reduce-like transformations, and this is where Spark really shines.

In Spark, joins are first-class citizens expressed concisely at the logical level. Spark transformations let the developer explicitly reference the two different RDDs that are participating in the join (inner, left, right, or full outer). In contrast, getting MapReduce to do joins is a huge kludge. In essence, the developer must code up the actual physical join plan using only map and reduce, unless a SQL-on-Hadoop platform like Hive is used.

In a standard MapReduce reduce-side join, because we can only map over a single input, to join R and S we'd have to mash together R and S in the input specification, and then in the mapper figure out if we're dealing with a record from R or a record from S (for example, by examining the input path). The join key is emitted as the intermediate key with the record as the value; the framework brings all records with the same join key together in the reducers, where the join processing actually happens.

On the other hand, if we wanted to do a copartitioned sort–merge join, often called a map-side join in MapReduce parlance, the implementation is even uglier. Typically, we map over one of the collections (say, R) and inside the mapper read (directly from HDFS) records from the other collection (S). Yuck!

Here, Spark claims a mic-drop moment. A join is simply `r.join(s)`. That's it. The join is specified logically, and thus Spark is able to figure out the best physical join plan behind the scenes. Beyond joins, Spark's set-like operations (`union`, `intersect`, `distinct`, `cartesian`) also have no equal in MapReduce. It's doable in MapReduce, but ugly.

So What Do We Really Need?

If MapReduce specifies two physical operators — partitioned per-record processing (map) and partitioned sort-based shuffle grouping (reduce) — we might wonder what's missing in its repertoire?

The major glaring hole in the design of MapReduce is the lack of operators that process two distinct datasets — which is why joins are so painful to implement. In particular, I see the need for two operators: a shuffle cogrouping operator (let's call it coreduce) and a copartitioned, cogrouping operator (let's call it comap). Both operators would take two collections of key-value pairs, `R: (K, V1)` and `S: (K, V2)`, and guarantee that values with the same key from both collections are available for processing together, something like `process(K, Iterator<V1>, Iterator<V2>)`. The coreduce operator would provide a general implementation via shuffling, whereas comap would assume that the input collections are copartitioned. With coreduce and comap, we can efficiently implement joins as well as the set-like transformations that Spark provides.

Once again, the rubber of clean design meets the road of implementation realities.

Interestingly, with map, comap, reduce, and coreduce, we arrive at something that's pretty close to Spark's actual physical operators. If we take a look at Figure 4 in the original Spark paper,[7] the wide dependencies are essentially reduce and coreduce, and the narrow dependencies are essentially map and comap. Unfortunately, in Spark, you don't have access to these. Often, I wish I did.

Given this discussion, we arrive at another way to think about the relationship between MapReduce and Spark in the context of data-parallel dataflow languages: MapReduce provides two physical operators that specify exactly how to wire up a dataflow graph for execution. The two physical operators are impoverished, which makes wiring up dataflow graphs to accomplish certain tasks (for example, joins) rather painful.

Spark, on the other hand, provides a set of logical transformations on collections that afford different physical execution plans — in other words, it provides a higher level of abstraction. My nitpicky complaint, summarizing the previous discussion, is that these logical transformations are in some cases inelegant and adulterated with abstraction-breaking assumptions about physical execution.

Thus, comparing MapReduce to Spark is a bit like comparing apples to oranges. Better points of comparison are actually Pig[11] and DryadLINQ,[12] both of which provide high-level transformations for manipulating collections of records. However, a thorough analysis is perhaps better left for another column.

As I intimated at the beginning of this column, although this discussion might be interesting from an academic or historic perspective, none of these issues particularly matter. In the evolution of big data processing technologies, the broader trend is toward increasing levels of abstraction that isolate the data scientists from the particulars of execution. In the Spark ecosystem, for example, there's a growing emphasis on DataFrames[13] as the default abstraction for manipulating datasets, as opposed to raw transformations on RDDs (which basically makes all discussion in this column irrelevant).

Today, these increasing levels of abstraction naturally segment into different verticals. As my colleague Ihab Ilyas opines, "data analytics will go vertical" — tackling specific market segments such as financial, pharmaceutical, healthcare, energy, the Internet of Things, and so on. I hear similar musings from investors: general infrastructure plays are becoming increasingly difficult in today's already crowded space, and going vertical is one avenue for differentiation.

At a high level, this column is a navel-gazing critique about instruction-set architectures for the datacenter computer. If the reduced versus complex instruction-set computing (RISC versus CISC) wars of the 1980s are a guide, ultimately, it doesn't matter,[14] and people only care about applications in the end. Well, not quite: people who write compilers still care — and if we follow this analogy, analytics infrastructure builders are the compiler writers of the 21st century for datacenter computers. We'll quibble about how high-level data science directives (such as "Train this machine-learning model!") translate into physical execution on warehouse-scale clusters; obviously important, but mostly relegated to a highly-specialized and esoteric craft.

Acknowledgments

I thank Semih Salihoglu, Reynold Xin, and Matei Zaharia for comments on previous drafts of this column.

References

1. D.A. Patterson, "The Data Center Is the Computer," *Comm. ACM*, vol. 52, no. 1, 2008, p. 105.
2. J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Proc. 6th Usenix Symp. Operating System Design and Implementation*, 2004, pp. 137–150.
3. J. Dennis, "First Version of a Data Flow Procedure Language," *Programming Symp.*, B. Robinet, ed., LNCS 19, 1974, pp. 362–376.
4. J. Dennis, "Data Flow Supercomputers," *Computer*, vol. 13, no. 11, 1980, pp. 48-56.
5. M. Isard et al., "Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks," *Proc. ACM Sigops/EuroSys European Conf. Computer Systems*, 2007, pp. 59-72.
6. C. Chambers et al., "FlumeJava: Easy, Efficient Data-Parallel Pipelines," *Proc. ACM Sigplan Conf. Programming Language Design and Implementation*, 2010, pp. 363–375.
7. M. Zaharia et al., "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing," *Proc. 9th Usenix Symp. Networked Systems Design and Implementation*, 2012; www.usenix.org/system/files/conference/nsdi12/nsdi12-final138.pdf.
8. J. Lin and C. Dyer, *Data-Intensive Text Processing with MapReduce*, Morgan & Claypool, 2010.
9. J. Lin and Michael Schatz, "Design Patterns for Efficient Graph Algorithms in MapReduce," *Proc. 8th Workshop Mining and Learning with Graphs*, 2010, pp. 78-85.
10. O. Boykin et al., "Summingbird: A Framework for Integrating Batch and Online MapReduce Computations," *Proc. Very Large Data Bases*, vol. 7, no. 13, 2014, pp. 1441–1451.
11. C. Olston et al., "Pig Latin: A Not-So-Foreign Language for Data Processing," *Proc. ACM Sigmod Int'l Conf. Management of Data*, 2008, pp. 1099–1110.
12. Y. Yu et al., "DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language," *Proc. 8th Usenix Symp. Operating System Design and Implementation*, 2008; www.usenix.org/legacy/event/osdi08/tech/full_papers/yu_y/yu_y.pdf.
13. M. Armbrust et al., "Scaling Spark in the Real World: Performance and Usability," *Proc. Very Large Data Bases*, vol. 8, no. 12, 2015, pp. 1840–1843.
14. E. Blem et al., "ISA Wars: Understanding the Relevance of ISA Being RISC or CISC to Performance, Power, and Energy on Modern Architectures," *ACM Trans. Computer Systems*, vol. 33, no. 1, 2015, article no. 3.

Jimmy Lin holds the David R. Cheriton Chair in the David R. Cheriton School of Computer Science at the University of Waterloo. His research lies at the intersection of information retrieval and natural language processing, with a particular focus on big data and large-scale distributed infrastructure for text processing. Lin has a PhD in electrical engineering and computer science from MIT. Contact him at jimmylin@uwaterloo.ca.

This article originally appeared in IEEE Internet Computing, *vol. 21, no. 3, 2017.*

IEEE computer society

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 13 magazines, 19 transactions, and one letters. 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 two software developer credentials. For more information, visit www.computer.org/certification.

NEXT BOARD MEETING

7-8 June 2018, Phoenix, AZ, USA

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: hq.ofc@computer.org
Los Alamitos: 10662 Los Vaqueros Circle, Los Alamitos, CA 90720 **Phone:** +1 714 821 8380
Email: help@computer.org

MEMBERSHIP & PUBLICATION ORDERS
Phone: +1 800 272 6657 • **Fax:** +1 714 821 4641 • Email: help@computer.org
Asia/Pacific: Watanabe Building, 1-4-2 Minami-Aoyama, Minato-ku, Tokyo 107-0062, Japan
Phone: +81 3 3408 3118 • **Fax:** +81 3 3408 3553
Email: tokyo.ofc@computer.org

IEEE

revised 7 Feb. 2018

EDITOR: Irena Bojanova, NIST, irena.bojanova@computer.org

Big Data and Big Money

The Role of Data in the Financial Sector

Jennifer Q. Trelewicz, *Deutsche Bank Technology Centre*

When we think of industry sectors driven by high tech, for some people, perhaps, banking is not the first that comes to mind. However, when we consider the 3Vs of big data[1]—volume, velocity, and variety—it is hard to think of many sectors whose requirements fit so nicely into the guidelines. For example, in April 2016 alone, the foreign exchange (ForEx) markets averaged US$5.1 trillion per day.[2] The ForEx markets provide real-time exchange rates between currencies across the world, facilitating global business and settlements.

In this article, I discuss the relevance of big data approaches to the financial sector, outlining challenges to adoption as well as future opportunities for technology development. Because of its transaction and money volumes, I focus on corporate banking (financial markets, corporate credit, trading, and so on), although many application areas are also relevant to consumer finance.

Big Data in the Financial Sector

Let us first examine the relevance of the 3Vs to finance:

- Volume is considered to reach big data levels at many Tbytes or even Pbytes of data. The financial industry produces a huge volume of quotes, market data, and historical trade data. The New York Stock Exchange (NYSE) alone writes more than a Tbyte per day.[3]
- Velocity suits big data when the speed of data storage or processing is on the order of 105 transactions per second or more. Generating data at this speed is no challenge for the financial markets. Moreover, the faster systems can process trade data, the faster they can manage trading.
- Variety implies that big data algorithms do well with various formats and data sources. In corporate banking, institutions work with reference data (about legal entities, for example), trade and market data, requests from clients (by electronic and voice means), and many other sources.

What makes the financial sector even more interesting from a big data standpoint is the constant stream of new regulations and reporting standards that bring new data sources and more complex metrics into financial systems. This makes the sector a very interesting place for the data scientist.

The ForEx markets, as mentioned earlier, trade 24 hours per day, from morning in Sydney to evening in New York, except for a small window during the weekend. Additionally, algorithmic trading has been used in the financial markets for a long time in one form or another. The NYSE introduced its Designated Order Turnaround (DOT) system in the early 1970s for routing orders to trading desks, where the orders were executed manually. Now, algorithmic trading systems break very large orders into smaller pieces that are executed automatically based on time, price, and volume, optimized for market parameters.

On a continuous basis, the processing of large volumes of data is used for reporting purposes in financial institutions:

- Banking and financial market regulations more and more often require the calculation of various complex metrics, such as XVA (valuation adjustments of derivative instruments, based on counterparty credit risk, cost of funding, margin, and so on). Such metrics are used, for example, to

 2469-7087/18/$33.00

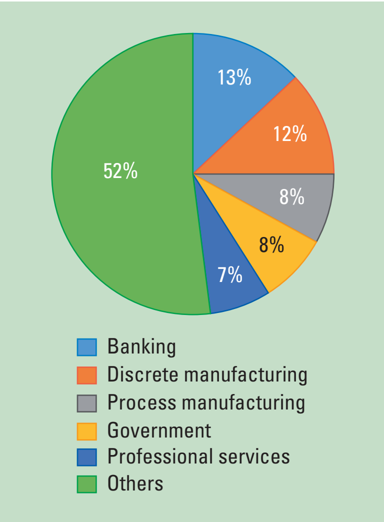


Figure 1. The 2016 market for big data analytics (US$130.1 billion). (Original data from IDC[5])

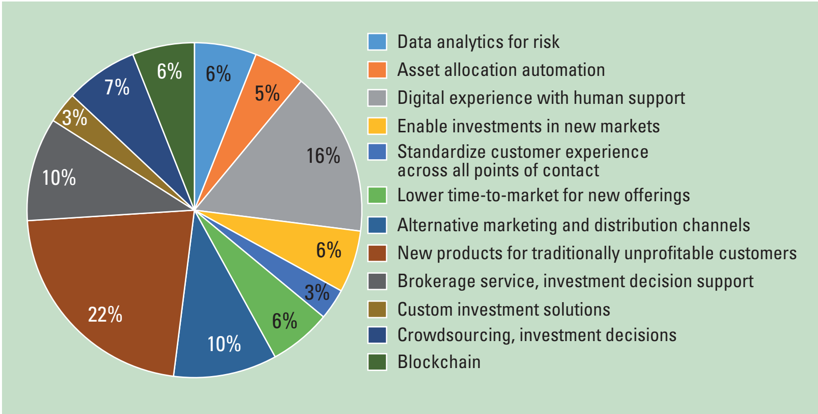


Figure 2. Approximate proportion of the FinTech market by area. (Original data from PwC[6])

set the minimal capital reserves of a bank, which directly influences the bank's profitability.

- Time-sequenced transactional data is analyzed to model market and customer behavior. For example, mapping trade volume with time could help to predict the probability of a default on credit, saving a bank lost resources on a loan.

Some large financial institutions have been slow to adopt big data approaches, but market research from PwC has clarified some of the organizational and cultural inhibitors to adoption in these institutions,[4] many of which are relevant in other industrial sectors as well. First, some financial-sector managers feel that big data algorithms solve technical problems, but not business problems. However, when the data is generated by the business, and the results are used by the business, it is clear that technology is supporting the business. Some do not understand how to gain value from their datastreams, while some feel that big data approaches improve technical efficiency but do little for the bottom line. However, the deep analysis that big data approaches can provide can directly support business growth and improved effectiveness. The financial sector has not traditionally been a destination for data scientists, so some institutions have met with difficulty in finding and attracting the needed skills to their organizations. Finally, even when the will for transformation is in place, it might not be clear how and where to start transforming an enterprise to utilize big data approaches.

However, banking is among the top industry sectors investing in big data analytics, according to a recent study from IDC,[5] as Figure 1 shows. Moreover, financial technology, or FinTech, companies are developing solutions and products for a range of banking needs for asset and wealth management; Figure 2 breaks out the proportion of FinTech companies per area according to PwC.[6] Following this trend, there is a growing body of research and algorithm development around other uses of financial data for increasing business effectiveness. We next examine a few of these.

Market Trading Patterns

Adaptive models of market trading patterns can provide input to investment strategies for buying and selling certain types of assets. This section explores one example.

Ex-dividend Day (also called X-Day) is the first trading day when the seller of a depository receipt (DR) has the right to receive the last dividend payout. Before X-Day, the buyer of the DR would receive the dividend payout. American DRs (ADRs) are financial instruments traded in the US market by non-American companies. As such, there is a dual tax burden on the dividends: the US taxes and the tax withholding in the country that issued the ADR. As a result, investors are motivated to sell ADRs before X-Day and to buy after X-Day. Naturally, tax policy has a strong influence on the stability of the ADR market.

In recent work, Bi-Huei Tsai examined the ADR market to understand market trading volumes.[7] Analysis of such markets by such a class of algorithms could be used to suggest optimal trading times based on recent market volumes. The author analyzes excess ratios of ADR volume (the daily trading volume minus the "normal" daily trading volume) during the ex-dividend period (X-Day +/– 10 days), positively correlated to dividend taxes, providing a model of tax policy's

This article originally appeared in
IT Professional, *vol. 19, no. 3, 2017.*

influence on the ADR market. Both traders and government tax authorities could use such models to create strategy.

Real-Time Credit Ratings

An application that isn't specific to financial markets but has relevance to banking for consumers and small and medium businesses is processing data to produce credit scores for applicants in real time. For example, FinTech companies such as Klarna, Lenddo, and Credit Karma provide services related to online credit scoring and verification. In recent work, Ying Wang, Siming Li, and Zhangxi Lin examine the potential for real-time credit scoring for e-commerce.[8]

Anyone who has applied for a significant amount of credit will be familiar with the timeline of the process. Traditionally, banks collect information about the applicant from both the application form and other sources. Specialists analyze this information to create a credit proposal for the client, which includes the interest rate and terms of repayment. There might be some negotiation between the applicant and the bank, including tradeoffs between various loan parameters for better overall terms. After the credit contract is signed, the client can engage in his or her financial activity and further pay off the loan.

Not only do data collection and terms negotiation take time, but two factors complicate the picture. First, many of the central data stores used for credit ratings are updated only monthly, so the client's recent financial problems might not be known to the bank from such sources. Moreover, nonfinancial factors could play a role in the risk of default. The authors examined several such factors relating to a large e-commerce platform: frequency of login to the platform, provision of additional contact data (including mobile telephone number), volume of transactions in the last month, number of successful transactions overall, time as a client of the platform, the client's business sector, and so on. The authors use linear regression analysis over groups of these parameters on historical client data to derive a correlation between the probability of default and the various parameters. The result is a model that could facilitate real-time credit ratings for the e-commerce platform, based on online behavior. Such parameters are not even present in traditional, central credit-rating databases.

Banking Becomes More High-Tech

The two examples described are by no means a limit to the applications of big data algorithms in the financial sector. Although no one can predict the future markets a hundred percent, deep analysis of historical data and current market parameters provide sophisticated, adaptive models of tendencies and behaviors in the markets. In turn, such models facilitate better-informed and faster decisions by traders (including trading systems), financial institutions, and other players.

The opportunity for *IT Pro*'s readers is to develop new technologies and solutions for a fast-growing sector. Funding for FinTech more than doubled between 2014 and 2015,[6] indicating both opportunity and a need for such products and solutions. The broad categories shown in Figure 2 leave a great deal of room for innovation in product, process, and customer experience. It might not be long before banking comes to mind first when we think of industry sectors driven by high tech.

References

1. D. Laney, *3D Data Management: Controlling Data Volume, Variety and Velocity*, MetaGroup, 2001.
2. *Triennial Central Bank Survey: Foreign Exchange Turnover in April 2016*, Bank for Int'l Settlements, Sept. 2016; www.bis.org/publ/rpfx16fx.pdf.
3. *Where Have You Been All My Life? How the Financial Services Industry Can Unlock the Value in Big Data*, PwC FS Viewpoint, Oct. 2013.
4. T. Nath, "How Big Data Has Changed Finance," Investopedia, 9 Apr. 2015; bit.ly/1atP3jC.
5. *Semiannual Big Data and Analytics Spending Guide*, IDC Worldwide, 3 Oct. 2016.
6. *Blurred Lines: How FinTech Is Shaping Financial Services*, PwC Global FinTech Report, Mar. 2016.
7. B.-H. Tsai, "Examination of Ex-Dividend Day Trading Using Big Data of American Depositary Receipts," *Proc. 2nd Int'l Conf. Advanced Cloud and Big Data* (CBD), 2014, pp. 34–38.
8. Y. Wang, S. Li, and Z. Lin, "Revealing Key Non-Financial Factors for Online Credit-Scoring in e-Financing," *Proc. 10th Int'l Conf. Service Systems and Service Management* (ICSSSM), 2013, pp. 547–552.

Jennifer Q. Trelewicz *is the technical risk officer and CTO for Enterprise Risk Technology at Deutsche Bank Technology Centre. Her research interests include big data, high-performance numerical algorithms, and innovation management. Trelewicz has worked in industry-leading technology companies such as IBM, Microsoft, and Google, has numerous granted patents in various countries and publications in international journals and refereed conferences, and volunteers actively with IEEE. She has a PhD in signal processing from Arizona State University, and is a life member of the international honor societies Tau Beta Pi and Phi Kappa Phi. Contact her at jentre@ieee.org.*

Silver Bullet Talks with Nicole Perlroth

Gary McGraw | Synopsis

Hear the full podcast and find show links, notes, and an online discussion at www.cigital.com/silverbullet.

Nicole Perlroth covers cybersecurity for the *New York Times* and the Bits Blog. Before joining the San Francisco bureau in 2011, she was deputy editor at *Forbes*, where she covered venture capital and web start-ups. Perlroth is the recipient of several journalism rewards for her reporting on efforts by the Chinese government to steal military and industrial trade secrets.

According to rumor, the first pitch you ever did landed you on the front page of Sunday's *Post*, so do tell.

Yes, at the time, I hadn't done anything in journalism. I didn't work for a student paper or have ambitions of being a journalist. After Princeton, I took some jobs that a lot of graduates take. I was a consultant for a little while. I worked on Capitol Hill for a little while. I worked in marketing for Coach, the handbag company, for a little while. And I just thought all of these jobs were completely mind-numbing. I just missed any kind of intellectual stimulation.

So I ended up taking one of those adult continuing-studies classes at NYU at night—a feature-writing class. The guy who taught it, a business columnist at the *New York Post*, said "you know you have some skills here, I think you should try freelancing."

Actually, he gave me an assignment. Some of your listeners may remember someone found rats doing cartwheels in the back of this Taco Bell/KFC. So he said, "Why don't you do a freelance story for the *New York Post* about how it's not just Taco Bell. All these expensive restaurants probably have rat problems too." So I said, "Okay."

By day I'm working at this luxury handbag company and by night I'm going through the Department of Health's restaurant records to see which of these nice restaurants had rat problems. And I found that a restaurant that I really like had one of the most horrific health records I'd seen; it was horrendous. I couldn't wrap my head around it because I'd just been there, and it was really clean.

So I called them and I said, "I happened upon your health record, and I really can't parse this because I was just at your restaurant. It seemed pretty clean and sanitary to me." And they said, "Well, thanks for calling. Actually, the health inspector came to our restaurant to do his review, and he ended up getting drunk at the bar and passing out for two hours. And we think he just made a bunch of stuff up on our report to justify to his supervisor why it took him so long to do the inspection." So I said, "You have got to be kidding. If only there were proof of this. We should get this guy." And they said, "Well actually, we have a camera in the restaurant that caught some of this."

So there was proof?

There was proof. I had no idea what kind of story I had. I sheepishly walk into the *New York Post* and say, "I'm really sorry I didn't get to do this assignment the way you envisioned it, but I did end up getting video footage of the health inspector drunk, passed out at the bar, and making up this whole report." It went on the cover of the Sunday's *New York Post* with a great headline that said something like "Rat Nap Inspector Snoozoo."

After that I was hooked. The guy was fired. The Department of Health had to have new rules for inspections, and there had to be some follow up with the restaurants. To see the impact of something that started out as such a silly assignment was pretty cool.

I was going to ask you what got you into covering information security, but if you start with rats, why not just keep on going?

Exactly. I mean I didn't choose information security. While at *Forbes*, I was based in Silicon Valley,

About Nicole Perlroth

Nicole Perlroth covers cybersecurity for the *New York Times* and the Bits Blog (www.nytimes.com/column/bits). Before joining the San Francisco bureau in 2011, she was deputy editor at *Forbes*, where she covered venture capital and web start-ups. Perlroth is the recipient of several journalism rewards for her reporting on efforts by the Chinese government to steal military and industrial trade secrets. She has a BA in political science and near Eastern studies from Princeton and an MA in journalism from Stanford, and is a native of the Bay Area where she still lives.

covering venture capital during the heat-up ahead of Facebook's IPO, including a lot of the investors and some of the private-share sales. So I got a few cover stories in *Forbes*. One day, I got a call from a blog editor at the *New York Times*, who said, "We're looking at you for a job; it's cybersecurity." And I remember thinking, you want to take me off this gravy train to cover cybersecurity. I honestly didn't think I was qualified. I told myself, well it's an honor to get invited into the *New York Times* building and go to New York for these interviews, so I'll just be myself and see what comes of it. But lo and behold, I ended up getting the job, and the rest is history.

So how does venture capital compare to the others you've covered?
Venture capital is interesting because it's very self-promotional. I get a lot of calls from venture capitalists trying to get me to write them a glowing profile, which is similar in some ways to information security because we constantly get pitched by cybersecurity firms or firms that want to slap the word cyber onto their website, looking for promotion. Dealing with the self-promoters was very good training for me.

It's no longer even the reporting and writing that's the hardest part of my job these days. It's the day after the story publishes, on Twitter. When you're writing for a layperson audience, the technical audience is never going to be happy with how you're covering information security. Dealing with such a hypercritical, vocal, philosophical, almost religious-like community has been the hardest part; nothing prepared me for that.

You've been involved with some really big stories, including the recent "Russian Election Hacking Efforts, Wider Than Previously Known."
As you know, you can look out ahead of the 2018 elections and see what's being done to prevent another situation like the one we had in 2016. I started digging around and found that there were still many more unresolved issues from the 2016 election than had been covered by the mainstream media.

One of the things that was so soothing after the election was we knew Russia had done a series of disinformation campaigns. We knew about the propaganda efforts, although at that point, we didn't know the extent of them. But for the most part, the intelligence report that came out last January delivered the message: Despite all those efforts to influence the 2016 campaign, Russia stopped short of hacking the actual tallies.

What I've learned in the reporting is that this conclusion came predominantly from *spies*, spies that we have and digital intercepts of Russian communications. Someone likely told someone else, we didn't hack the vote count, and they were really surprised that Trump had won without those efforts.

But no real forensic effort had been made to ensure that some of the systems that were hacked on the back end didn't impact the votes.

One place that kept coming up in my conversations was Durham, North Carolina. The county used an electronic poll book vendor called VR Systems that we know, from a leaked NSA report, had successfully been hacked by the Russian GRU. A lot of the problems on Election Day fit the MO of someone trying to create chaos or prevent people from voting. Durham is a predominantly blue county in a swing state. As we dug deeper, we found that there were instances of people showing up with their registration cards and being told that they were no longer registered and they've been marked as inactive, or they voted early when they hadn't, or they voted absentee when they hadn't.

Fishy.
It was written off as a glitch, but no one had ever done a forensics investigation. Then I found out the county had hired a local security company to do some forensics investigation. I got my hands on that report; it was like an old cop report where they had gone to poll workers and written, "at 6:09 I interviewed Judy from precinct number three." But there was no actual forensics analysis. The US Department of Homeland Security and FBI never analyzed the systems in Durham, because they have to be invited by the county and the state, and the state had rebuffed their efforts. I started unraveling this and realized that this tale was more common than we knew.

I hope that story gets more attention. It seems easy to cover spectacular failures like the Equifax breach, because humans love to read about disasters. But how do we get coverage for important but not really sexy computer security stuff like software security?
As far as software security, I think people are getting more interested in it. In the book I'm writing about the exploit market [*This Is How They Tell Me the World Will*

This article originally appeared in
IEEE Security & Privacy, *vol. 15, no. 6, 2017.*

End], I'm hoping to end on software security because there's now more awareness of vulnerabilities and how big an impact human error and sloppiness can have. And there was a big focus on the Equifax breach, at least initially, on the vulnerability.

Another thing that gave me hope was during Facebook's IPO—I know this from my venture IP days—their philosophy was "move fast and break things." And recently, while I was at Facebook headquarters, I saw that motto had been replaced: there were signs on the wall that said, "Move slowly and fix your sh*t."

And the CSO at Facebook has had a good influence over there.

Oh yeah. And at Google, the amount of fuzzing that's going on … they say that their motto since the 2010 Chinese attack that happened there has been "never again," that security and software security is extremely important to them. How that trickles down to Android I don't know. But it's something that's getting talked about pretty seriously at the executive level at some tech companies, which is a good thing.

Do you think that CSOs are doomed to be the guy before the guy, or are you seeing forward progress?

I think that was the case in the past. But now that we've finally admitted that we've all been hacked, we want a veteran of serious nation-state attacks to protect our business because they understand that compliance checklists aren't going to cut it. I think that now it's not the CSO that gets fired; it's the CEO.

From your perspective, what one thing should we all do to encourage more women to join cybersecurity?

I don't think we play up the sex appeal of cybersecurity enough. You're not just coding; you're playing cops and robbers. I talked earlier about how little intellectual stimulation there was in some of the jobs I've had. The cybersecurity world is full of intellectual stimulation. It's amazing—three days ago, I knew nothing about North Korean counterfeiting operations, but I had to go quickly study up on it because of the financially motivated attacks that North Korea has been launching to make up for the fact that it's counterfeiting operations are no longer as effective.

There's a real political bent to a lot of the nation-state attacks that we're seeing. I don't think people realize it's not just code and it's not just hackers in their basements—there are real opportunities here to be on the front lines of history. And talk about employment security. You know the problem is only getting worse.

I actually met a young woman the other day who was going to Johns Hopkins, and she had sought out an internship related to cybersecurity for the summer. That was the first I'd heard of a freshman co-ed seeking out a job like this. Most people just fall into it. That was a nice thing to hear.

All right, very last question: where's your favorite place to dive on the planet?

I've skydived in really ugly places, so New Jersey and Lodi, California—I couldn't tell you which one was better.

The Silver Bullet Podcast with Gary McGraw is cosponsored by Cigital (part of Synopsys) and this magazine and is syndicated by SearchSecurity. ■

Gary McGraw is vice president of security technology at Synopsys. He's the author of *Software Security: Building Security In* (Addison-Wesley 2006) and eight other books. McGraw received a BA in philosophy from the University of Virginia and a dual PhD in computer science and cognitive science from Indiana University. Contact him via garymcgraw.com.

Editor: **Eoin Woods**
Endava
eoin.woods@endava.com

Should Architects Code?

Eoin Woods

A FEW SOFTWARE architecture questions always light up the Twittersphere with controversy when asked:

- What is architecture, and is it just design?
- Do you need architecture in agile development?
- Should architects code?

I examined the first two questions in previous columns[1,2] and in this column address the third, which is an intriguing question without an obvious answer.

When people ask, "Should architects code?" or talk about "the coding architect," they might be referring to anything from an architect or designer keeping a working knowledge of the technology in use (and being able to review and write code if necessary) to an architect spending significant time writing a system's production code.

Let's assume that the question is simply whether the people performing the system's architecture work should also develop some of the system's production code.

Personal Motivations

Often, architecture work naturally diverts architects from spending large amounts of time developing a system's production code. Architecture is a technical management activity that involves a range of work, not just coding. So what are some personal reasons for architects to continue coding work?

First, to lead a technical team, architects must build and maintain technical credibility so that other team members respect their opinions. Displaying strong coding skills can help build these technical credentials.

Second, architects should continue coding to maintain and improve their development skills—to not only achieve personal satisfaction but also set high yet realistic standards for others. Nothing clarifies expectations about "quality" and "craftsmanship" better than a well-written example.

Finally, many individuals became software engineers because they like developing software. Coding can increase motivation while keeping skills current.

What Are the Benefits?

There are potential benefits to architects performing code development for their systems.

First, coding work offers a useful reality check about the experience of working as a developer on the system. Are the technologies easy for developers to use? Is the build-and-release pipeline working effectively? Are there any serious impediments to developer effectiveness? By working as a developer, architects can get a good perspective on such questions.

Performing implementation work also lets architects see their architecture's realization. This helps them more deeply understand their architectural decisions' implications and spot possible problems and those inevitable places where the implementation strays from the plan.

Development work also helps architects stay current with their system's technologies. Over time, technologies

 Published by the IEEE Computer Society 2469-7087/18/$33.00

This article originally appeared in IEEE Software*, vol. 34, no. 5, 2017.*

get replaced or evolve. When architects stop coding regularly, they can lose sight of these important details.

What Are the Drawbacks?

Coding while performing architecture work poses some difficulties as well.

First, the architects' priorities can become muddled—whereas their architecture work serves to make the team more effective, their development work reflects more personal objectives. They must think about coding time in terms of its return on investment (ROI). The first few hours a week will likely yield a high ROI, but how about the 20th hour? By then, there are almost certainly other tasks architects should be doing to make the team more effective.

A project's scale will affect the ROI estimation. The larger the team, the larger the delayed architecture work's impact. This is why I've reluctantly shrunk my coding time to almost zero on some projects—too many other high-priority issues required my attention.

Second, development and architecture work differ fundamentally. Development work demands significant periods of focused attention. Interruptions make developers less effective. In contrast, architecture work often involves reacting to questions or concerns and identifying and responding to risks or problems. It's difficult to work in both ways at once.

Combined, these factors create the risk that a coding architect will block the project's critical path. An architectural decision might not be made quickly because the architect is racing to finish a critical module. Or, an important feature might not be delivered because the architect was constantly interrupted while trying to finish an important part of the code.

A final factor that we architects might not want to admit is that, perhaps, we aren't as effective at coding as we used to be. Both technology and our individual skills change over time. If we're not 100 percent focused on development tasks, are we truly still as productive as we once were?

How Can Architects Stay Involved?

By keeping their development work off the critical path, architects can mitigate problems caused by conflicting or changing priorities. To remain closely involved in their system's implementation while avoiding the problems I've discussed, architects can do the following:

- *Fix bugs*. Fixing defects can be instructive and directly valuable to the project. It provides insight into the developer experience and the strengths and weaknesses of the architecture and code.
- *Refactor*. Technical debt nearly always accumulates, so architects might tackle it in small, safe steps. They'll quickly uncover any weaknesses in the architecture, implementation consistency, or tests.
- *Investigate problems*. Architects can get involved in debugging and problem investigation. Whether it's a performance problem, poor scalability, or a subtle intermittent error, they can offer a valuable perspective while learning about the qualities their architecture provides.
- *Test*. Architects might well find that testing isn't as thorough or sophisticated as they'd like. So, another opportunity for involvement is to improve automated tests. This will let architects continue to code while helping to develop a shared understanding with the team of how to test the system.
- *Create architectural spikes*. Perhaps the most obvious task to choose is carrying out the proof-of-concept exercises that support architectural decision making. Doing so can deepen architects' knowledge of their decisions' implications.

Architects should pair with developers whenever possible on these tasks. Not only can they share expertise, but the architects can also learn from those closer to the state of the art.

It's still important to keep an eye on the schedule, even for tasks off the critical path. If architects notice they're running out of time or are about to be distracted by another priority, they must quickly reassign development tasks—no project manager likes a surprise that has become difficult to mitigate.

So, should architects code? My experience is that there's generally a positive ROI when architects do carefully selected implementation work, whether it's testing, refactoring, architectural spikes, or simply some part of the system where they're the best person for the job. Provided the project's scale allows it, doing some coding helps to root architecture work, keep architects' technology knowledge up to date, and sometimes save their sanity!

References

1. E. Woods, "Return of the Pragmatic Architect," *IEEE Software*, vol. 31, no. 3, 2014, pp. 10–13.
2. E. Woods, "Aligning Architecture Work with Agile Teams," *IEEE Software*, vol. 32, no. 5, 2015, pp. 24–26.