The Beckman Report on Database Research

Daniel Abadi, Rakesh Agrawal, Anastasia Ailamaki, Magdalena Balazinska, Philip A. Bernstein, Michael J. Carey, Surajit Chaudhuri, Jeffrey Dean, AnHai Doan, Michael J. Franklin, Johannes Gehrke, Laura M. Haas, Alon Y. Halevy, Joseph M. Hellerstein, Yannis E. Ioannidis, H.V. Jagadish, Donald Kossmann, Samuel Madden, Sharad Mehrotra, Tova Milo, Jeffrey F. Naughton, Raghu Ramakrishnan, Volker Markl, Christopher Olston, Beng Chin Ooi, Christopher Ré, Dan Suciu, Michael Stonebraker, Todd Walter, Jennifer Widom

Abstract

Every few years a group of database researchers meets to discuss the state of database research, its impact on practice, and important new directions. This report summarizes the discussion and conclusions of the eighth such meeting, held October 14-15, 2013 in Irvine, California. It observes that Big Data has now become a defining challenge of our time, and that the database research community is uniquely positioned to address it, with enormous opportunities to make transformative impact. To do so, the report recommends significantly more attention to five research areas: scalable big/fast data infrastructures; coping with diversity in the data management landscape; end-to-end processing and understanding of data; cloud services; and managing the diverse roles of people in the data life cycle.

1. INTRODUCTION

A group of database researchers meets periodically to discuss the state of the field and its key directions going forward. Past meetings were held in 1989 [BDD+89], 1990 [SSU91], 1995 [SSU96], 1996 [SZ96], 1998 [BBC+98], 2003 [AAB+05], and 2008 [AAB+09]. Continuing this tradition, twenty-eight database researchers and two invited speakers met in October 2013 at the Beckman Center on the University of California-Irvine campus for two days of discussions (see `http://beckman.cs.wisc.edu`). The meeting attendees represented a broad cross-section of interests, affiliations, seniority, and geography. Meeting attendance was capped at thirty so that the meeting would be as interactive as possible.

This year, meeting participants quickly identified Big Data as a defining challenge of our time. Big Data emerged due to the confluence of three major trends. First, it has become much cheaper to generate a wide variety of data, due to inexpensive storage, sensors, smart devices, social software, multiplayer games, and the emerging Internet of Things, which connects homes, cars, appliances, and other devices. Second, it has become much cheaper to process large amounts of data, due to advances in multicore CPUs, solid state storage, inexpensive cloud computing, and open source software. Finally, in a trend called the democratization of data, not just database administrators and developers, but many more types of people have become intimately involved in the process of generating, processing, and consuming data – decision makers, domain scientists, application users, journalists, crowd workers, and everyday consumers.

As a result of these accelerating trends, there is now a widespread realization that an unprecedented volume of data can be captured, stored, and processed, and that the knowledge gleaned from such data can benefit everyone: businesses, governments, academic disciplines, engineering, communities, and individuals. In a sense, the rest of the world has now caught on to the importance of what the database community has been advocating and doing for years.

The new era of Big Data has drawn many communities into the "data management game." There has been a groundswell of efforts in these communities to develop custom data management solutions, such as Hadoop and NoSQL. Many of these early solutions were not based on database management system (DBMS) principles. However, as these solutions have gained popularity and been applied to more data management scenarios, DBMS principles have been increasingly recognized as important and incorporated into solutions. For example, Hive, which manages data declaratively, has become far more popular than MapReduce; new drop-in alternatives to Hive look like parallel DBMSs; NoSQL tools are now moving to high-level languages and ACID transactions; and a new generation of systems is emerging that look like massive relational database management systems running on top of key-value stores that span data centers (e.g., the F-

1 system that powers Google's ad infrastructure).

Thus, today the database community is entering a time of unprecedented excitement. We are now squarely at the center of the Big Data revolution. The world has adopted a vision of a data-driven society, and other communities are adopting DBMS principles. As the community that has been pushing the limits of processing big data sets for 45 years, we can build on a wealth of results, lessons, and experience to help the data-driven world move forward. Our community therefore is uniquely positioned to address Big Data. The opportunity for us to make transformative impact is enormous.

But we also face enormous challenges. Big Data requirements will cause massive disruptions to the ways that we design, build, and deploy data management solutions. The main characteristics of Big Data are volume, velocity, and variety. Our community has worked on volume and velocity for decades, and has developed solutions that are mission-critical to virtually every commercial enterprise on the planet. Big Data, however, brings unprecedented scale that will force us to radically rethink existing solutions. Variety means integrating and analyzing data that come from diverse sources, with varying formats and quality, a topic that we have also been working on for years. However, it is still an extremely labor-intensive journey from raw data to actionable knowledge. This problem will be exacerbated by Big Data, causing a major bottleneck in the data processing pipeline. Hence, we need to intensify our effort to develop end-to-end solutions that scale, are easy to use, and minimize human effort. Big Data also brings wide variety in hardware infrastructures; processing frameworks, languages, and systems; programming abstractions; degrees of user sophistication; and user preferences. Designing data management solutions that can cope with such extreme variety will be a difficult challenge.

Moving beyond the "three V's," many Big Data applications will be deployed in the cloud, both public and private, on a massive scale. Many applications will involve people, e.g., to help solve semantic problems that still bedevil current automatic solutions. The scale of human involvement can range from a single domain expert to a crowd of workers, a whole user community, or in some cases the entire connected world (e.g., Wikipedia). These new trends raise novel and important research challenges for our community.

Finally, Big Data brings important community challenges. We must rethink our approach to teaching data management technologies, reexamine our research culture, and consider the emergence of data science as a discipline. Other aspects of the Big Data revolution, such as the impact on privacy, new ideas about data valuation and ownership, and the emerging data economy, also need to be considered and may affect our training programs and research agendas. However, we will not focus on these aspects in this report.

Over the two days of the Beckman meeting, participants extensively discussed the above issues. Sections 2 and 3 summarize the discussions about research and community challenges, respectively. Section 4 concludes the report.

2. RESEARCH CHALLENGES

The meeting identified five Big Data challenges: scalable big/fast data infrastructures; coping with diversity in the data management landscape; end-to-end processing and understanding of data; cloud services; and the roles of people in the data life cycle. The first three challenges deal with the volume, velocity, and variety aspects of Big Data, while the remaining two deal with deploying Big Data applications in the cloud and managing the involvement of people in these applications.

These Big Data challenges are not an exclusive agenda to be pursued at the expense of other existing work. In recent years our community has strengthened core competencies in RDBMSs, and branched out into many new data management directions, in collaboration with other communities (e.g., systems, AI, KDD, HCI, and e-science). These thriving directions require continued investigation. In addition, important issues that were raised repeatedly during the meeting include security, privacy, data usage and pricing, data attribution, social and mobile data, spatio-temporal data, personalization and contextualization, energy constraints, and scientific data management. Many of these issues cut across the identified challenges and are captured in various aspects of the discussion below.

2.1 Scalable Big/Fast Data Infrastructures

Our community has long been developing systems for processing data in volumes that push the limits of current hardware. Hardware continues to evolve, bringing new processor, storage, and networking technologies. We must continue to address the challenge of building scalable systems to manage bigger data sets that arrive at increasing speed, leveraging these new and improved technologies.

In the database world, the parallel processing of large structured data sets has been a major success, leading to several generations of commercial SQL-based products that are widely used by enterprises.

The distributed computing field has achieved success in scaling up data processing for less structured data on large numbers of unreliable, commodity machines through the use of constrained programming models such as MapReduce. Higher level languages, inspired by declarative database languages such as the relational algebra and SQL, have followed. These have been layered on top of the earlier constrained models, to enable a broader audience of developers to use scalable Big Data platforms. Today, open source platforms such as Hadoop – with its MapReduce programming model, large-scale distributed file system (HDFS), and higher level languages (e.g., Pig and Hive) – are seeing rapid adoption for processing less structured data, even in the traditional enterprise world.

Given the enthusiastic adoption of declarative languages for processing Big Data, there is a growing recognition that more general, database-style query processing techniques are needed. These include cost-aware query optimizers and set-oriented query execution engines. Processing much higher data volumes with acceptable response times will require very high degrees of parallelism. Effective query processing strategies will need to fully exploit large clusters of many-core processors, scaling both "up" and "out" in order to meet the anticipated needs. This will create challenges not only for query optimization and execution, but also for progress monitoring, so that a user can diagnose and manage queries that are running too slowly or consuming excessive resources. To adapt to the characteristics of previously unseen data, as well as to reduce the cost of data movement between stages of data analysis, query processors will need to integrate data sampling, data mining, and machine learning computations into their flows.

At data center scale, the ratio between the speed of sequential processing and interconnects is changing with the advent of faster networks, full bisection bandwidth networks between servers, and remote direct memory access (DMA) capabilities. In addition to clusters of general-purpose multicore processors, more specialized processors should be considered. Commercially successful database machines have demonstrated the potential of hardware-software co-design for data management. Researchers should continue to explore ways of leveraging specialized processors, e.g., graphics processing units (GPUs), field-programmable gate arrays (FPGAs), and application specific integrated circuits (ASICs), for processing very large data sets. These changes in communications and processing technologies will require a reconsideration of parallel and distributed query processing algorithms, which have traditionally focused on more homogeneous hardware environments.

Turning to storage, the database research community must learn how best to leverage emerging memory and storage technologies. Relative to commodity magnetic disks, solid-state disks are expensive per gigabyte but cheap per I/O operation. Various non-volatile random-access memory (NV-RAM) technologies are under development, all with different speed, power, and durability characteristics. Both server-attached and network-attached storage architectures need to be considered. Distributed file systems like HDFS, which are server-attached yet shared across the network, are a hybrid of both approaches. How best to use this range of storage configurations reopens many questions reminiscent of past debates of shared memory vs. shared disk vs. shared nothing, questions that many have considered to be "closed" for traditional parallel relational systems.

To process data that arrives at ever higher speeds, new scalable techniques for ingesting and processing streams of data will be needed. Algorithms will need to be tuned carefully according to the behavior of hardware, e.g., to cope with non-uniform memory access (NUMA) and limited transfer rates across layers of the memory hierarchy. In addition, the very high speed of some data sources, often with lower information density, will require some data to be processed online and then discarded without being persisted in its entirety. Rather, samples and aggregations of such data will need to be selected to be stored persistently in order to answer certain categories of queries that arrive after the raw data is no longer available. For such data, progressive query processing will be increasingly important to provide incremental and partial results with increasing accuracy as data flows through the processing pipeline.

For data that is persisted but processed just once (if ever), it makes little sense to store and index the data first in a database system. For such data, schema-on-read may make more sense than traditional schema-on-write, which imposes unnecessary overhead at ingestion time. At that time one may just want to dump the bits without hassle, returning if and when one wants to interpret the bits. Further, the appropriate way of interpreting the data for a given query may depend on the query, and hence may be unknown at write time. Raw files (i.e., array-of-characters or array-of-bytes) are the least common denominator for interoperation among the wide variety of systems being brought to bear on

data today. As a result, we need to develop tools and languages to assist with schema-on-read, and query engines that run efficiently over raw files.

In addition to much broader data analysis requirements, today's world brings new requirements for data capture, updates, and fast (but simple) data access. Handling high rates of data capture and updates for schema-less data has led to the development of NoSQL systems. The current Big Data platform landscape contains a number of such systems, with nearly as many transaction models and data consistency guarantees as there are examples of such systems. Most provide only basic data access capabilities and weak atomicity and isolation guarantees, making it difficult to build and reason about reliable applications. As a result, a new class of Big Data system has emerged, one that provides full-fledged database-like features over key-value stores or similar substrates. For some applications, the stored data is still managed and updated as "the source of truth" for an enterprise. In other cases, such as the Internet of Things, the stored data reflects and needs to keep up with events and changes occurring in the outside world so that applications can respond to important events or recognize situations of interest. This creates an opportunity for our community to revisit its thinking about data currency and consistency and to design new models and techniques for developing robust applications.

Finally, scalability should be measured not only in terms of petabytes of data and queries per second, but also total cost of ownership (including management, energy use, etc.), end-to-end processing speed (i.e., time from raw data arrival to eventual insights), brittleness (for example, the ability to continue despite failures such as partial data parse errors), and usability (especially for entry-level users). To measure progress against such broader metrics, new types of benchmarks will be required.

2.2 Diversity in the Data Management Landscape

In addition to high data volumes and data arrival rates, today's data-driven world involves a much wider and much richer variety of data types, shapes, and sizes than traditional enterprise data.

In the enterprise world, data has traditionally been stored and analyzed in a data warehouse that has been carefully designed and optimized for repetitive and ad-hoc analysis tasks. In today's more open world, data is often stored in different representations managed by different software systems with different APIs, query processors, and analysis tools. It seems unlikely that a single, one-size-fits-all, Big Data system will suffice for this degree of diversity. Instead, multiple classes of systems will likely emerge, with each addressing a particular class of need (e.g., data deduplication, analysis of large graphs, diverse scientific experiments, real-time stream processing) or exploiting a particular type of hardware platform (e.g., clusters of inexpensive machines, large multicore servers). For these scenarios, database researchers should apply our expertise in set-oriented parallel processing and in efficiently handling data sets that do not fit in main memory.

It remains to be seen how many different types of systems may be needed – e.g., what an appropriate system's scope may turn out to be – but the need for coexistence of multiple Big Data systems and analysis platforms is certain. Thus, another diversity challenge is helping data analysts combine and analyze data across systems. To support Big Data queries that span systems, platforms will need to be integrated and federated. This will involve not only hiding the heterogeneity of data formats and access languages, but also optimizing the performance of accesses that span diverse Big Data systems and of flows that move data between them. We also face a challenge of managing Big Data systems that run on a multitude of diverse devices and reside within or span large data centers. Disconnected devices will also become increasingly common, raising challenges related to reliable data ingestion, query processing over these devices, and data inconsistency in such sometimes-connected, wide-area environments.

Moving up a level, in a diverse and data-driven world, we must manage diverse programming abstractions against very large data sets. Rather than expecting to develop "the" data analysis language for Big Data, perhaps by extending SQL or another popular language, we must let users analyze their data in the medium they find most natural. For example, this may be SQL, Pig, R, Python, a domain-specific language, or a lower-level constrained programming model such as MapReduce or Valiant's bulk synchronous processing model. This requires developing reusable middle-layer patterns, such as scalable matrix multiplication, list comprehension, or iterative execution paradigms, with support for multiple language-specific bindings or embeddings. Another potentially fruitful focus is tools for the rapid development of new domain-specific data analysis languages – tools that simplify the implementation of new scalable, data-parallel languages.

To handle data diversity, then, we need modular platforms that can span both "raw" and "cooked" data, systems where the cooked data can take many forms, e.g., tables, matrices, or graphs. Such systems will run end-to-end dataflows and workflows that mix multiple types of data processing, e.g., querying data with SQL and then analyzing it with R. To help unify systems that access data in such diverse ways, lazy computation is sometimes beneficial – including lazy data parsing/conversion/loading, lazy indexing/view construction, or just-in-time query planning. Big Data systems should become more interoperable and interconnectable, like "Lego bricks." Frameworks like Mesos and now YARN provide inspiration at the systems level, as do workflow systems for the Hadoop ecosystem and tools for managing scientific workflows.

2.3 End-to-End Processing and Understanding of Data

To meet the needs of a data-driven world, the database research community needs to focus on end-to-end processing and understanding of data. Despite years of R&D, surprisingly few tools can process data end-to-end, going from raw data all the way to extracted knowledge, without significant human intervention at each step. Moreover, for most steps, the intervening people need to be highly computer savvy. Few tools in this area are open source. Most are expensive proprietary products that address certain processing steps. As a result, existing tools cannot easily benefit from ongoing contributions by the data integration research community. To overcome this situation, we recommend a focus not just on improving data integration technologies (such as data cleaning, schema matching, and data deduplication), but also on fusing these puzzle pieces into end-to-end solutions.

What should such end-to-end tools look like? At its core, the raw-data-to-knowledge "pipeline" will look much like it always has. Its major steps will continue to be: data acquisition; selection, assessment, cleaning, and transformation (also called "data wrangling"); extraction and integration; mining, OLAP, and analytics; and result summarization, provenance, and explanation. What has significantly changed is the much greater diversity of data and users, and much greater scale. Data today comes in a wide variety of formats. Combinations of structured and unstructured data are appearing that users want to use together in a structured fashion. Further, a wide range of people from many different domains are now building data tools that exploit human feedback in almost every step of the analytical pipeline. Data tools are increasingly used directly by subject-matter experts, not just by IT experts. For example, a journalist with a CSV file of crime statistics may want to clean, map, and publish his or her data. An entirely new class of people devoted to data analysis, called data scientists, has emerged. Once an analytic result has been produced, it is likely to be consumed by a much wider variety of people than before. Finally, data tools are now being used at every imaginable scale, from extracting and combining data from just a few Web pages to mining petabytes of system, network, and application logs and event streams.

Our community should seek to build effective, useful, and impactful tools that can work together, end-to-end. There will likely be no one-size-fits-all tool for the wide variety of data analysis scenarios ahead. We should thus develop multiple tools, each solving some piece of the raw-data-to-knowledge puzzle, which can be seamlessly integrated and be easy to use for both lay and expert users. When possible, we should aim to open source data analysis "building blocks," to be combined and reused by others, and provide best practice guidance on when to use each tool. Tools should handle the range from a small amount of data up to very large volumes. In an increasingly collaborative world for data sharing and analysis, each step of the data analysis pipeline should be interactive and be able to exploit feedback from individuals, teams, and even crowdsourcing.

Tools should be able to exploit domain knowledge, such as dictionaries, knowledge bases, and rules, and be easy to customize to a (new) domain. With a large volume of data to analyze, tool designers should consider using machine learning to partially automate the customization process. Handcrafted rules will remain important, though, as many analysis applications require very high precision, such as e-commerce. In such applications, analysts often write a large number of rules to cover "corner cases" that are not amenable to learning and generalization. To be truly end-to-end and easy to use, tools should provide support for writing, evaluating, applying, and managing hand-crafted rules.

Explanation, provenance, filtering, summarization, and visualization requirements crop up in all steps of the raw-data-to-knowledge pipeline. They will be critical to making analytic tools easy to use. Capturing appropriate meta-information is key to enable explanation, provenance, and reuse. Furthermore, visualization provides an essential way to interact with and solicit input from people, and it can be especially effective when coupled with automatic analysis techniques. Visual analytics is re-

ceiving growing attention in the database, HCI, and visualization communities, for visualizing database queries, visual data mining, and data wrangling. This area would benefit from attention, as it is a must for coping with Big Data volumes.

Analytical data management is knowledge-intensive. The more knowledge we have about a target domain, the better that tools can support the domain's analyses. As a result, there has been a growing trend to create, share, and use domain knowledge to better *understand* data. Such knowledge is often captured in knowledge bases (KBs) that describe the most important entities and relationships in a domain. For example, a community of domain scientists, say in biomedicine, may build a large KB that contains profiles of tens of thousands of biomedical researchers along with their publications, affiliations, and patents. Such KBs are used for improving the accuracy of the raw-data-to-knowledge pipeline, answering queries about the domain, and finding domain experts. Many companies have also built KBs for answering user queries, annotating text, supporting e-commerce, and analyzing social media.

The KB trend will likely accelerate, leading to a proliferation of "knowledge centers" built, maintained, and used by online communities, companies, and others. Such centers will contain knowledge bases as well as tools to query, share, and use them for data analysis. Many of these tools will be invocable in the cloud, allowing users and applications in the same domain and beyond to use the knowledge centers. End-to-end processing from raw data to knowledge will require our community to pay increased attention to this trend, as it can be viewed as using domain knowledge, often a great deal of it, to better understand the raw data in terms of the entities and relationships in its domain. To date we have made some inroads into this topic (e.g., efforts to build KBs in various domains) and have had some significant success (e.g., YAGO). However, more needs to be done, including developing solutions to let a group of users build and maintain a domain-specific KB, to let raw-data-to-knowledge tools utilize such KBs, and to allow users, from layman to experts, to easily query and share such KBs.

2.4 Cloud Services

Cloud computing has become mainstream. Enterprises have a multitude of cloud providers to choose from. Cloud computing has a wide variety of forms, including IaaS (Infrastructure as a Service), PaaS (Platform as a Service), and SaaS (Software as a Service). Moreover, the distinctions among IaaS, PaaS, and SaaS have started to blur. For example, IaaS providers nowadays provide manageability features that begin to resemble PaaS. From a data platform perspective, the ideal goal is to provide PaaS in its truest form. In a world with true PaaS for data, users would be able to upload data to the cloud, query it exactly as they do today over their SQL databases on the intranet, and selectively share the data and results easily, all without worrying about how many instances to rent, what operating system to run on, how to partition the databases across servers, or how to tune them. Despite the emergence of services such as Database.com, Google Big Query, Amazon Redshift, and Microsoft Azure SQL Database, we are still far away from that vision. Below we outline some of the critical challenges for our community in realizing the vision of Data Platform as a Service in the cloud.

The first challenge is elasticity. While computation is elastic in many cases, data is not. In today's architectures, data can be prohibitively expensive to move. Given this reality, if we want to build an elastic Data Platform as a Service, how should it be architected, keeping in mind the evolution of storage and networking? Should storage be server-local or network-attached? Can the same cloud storage service support both transactions and analytics? How does caching fit into the picture? Handling elasticity also requires leveraging the availability of additional resources as well as preemption of existing resources. Database engines and analysis platforms for a Data Platform as a Service will need to operate on top of elastic resources that can be allocated quickly during workload peaks but possibly pre-empted for users paying for premium service.

Data replication is another challenge. Although data replication has been studied extensively in the past, it is important to revisit it in the context of the cloud, keeping in mind the need for high availability, load balancing, and cost. Both elasticity and replication need to be considered not just within, but also across, geographically distributed data centers.

System administration and tuning is a third challenge. A data platform in use as a cloud service will need extreme auto-tuning. In the world of Data Platform as a Service, the traditional roles of database and system administrators simply do not exist. Therefore, all administrative tasks such as capacity planning, resource provisioning, physical data management, and admission control policy setting need to be automated while dealing with the variance that arises due to the elasticity of resources and their availability in the cloud setting.

Multitenancy is a key technical challenge in man-

aging elasticity for data-related services. To be competitive, the provider of a Data Platform as a Service must offer a cost structure comparable to or better than an on-premises solution. This requires providers to pack multiple tenants of a database service together to share physical resources on the same server to smooth demand and reduce cost. However, multitenancy introduces two problems. First, providers must be able to provide performance isolation so that a burst of demand from one tenant does not unduly degrade the performance of others. This requires careful governance of CPU, I/O, memory, and network resources. Second, users of a database service must be given security guarantees against information leakage across tenants.

Service Level Agreements (SLAs) are critical but challenging in the world of cloud services. For a multitenant Data Platform as a Service, the elasticity of global resource availability as well as the need for resource governance impact the availability of resources for a tenant. In turn, such variations can affect the quality of service. We are just beginning to understand the interaction between multitenant resource allocation and quality of service. Such an understanding would help form the basis for differentiated SLAs for Data Platform as a Service. Today, SLAs primarily focus on availability. To make Data Platform as a Service ubiquitous, it is important to understand this key question deeply, as it has implications not only for cost structures for tenants, but also for the development of QoS-aware database applications based on cloud services. In addition, cost structures for differentiated services must be easy to comprehend in user terms.

Data sharing is another key challenge, as the cloud enables it at an unprecedented scale. The database community should seek to develop novel services that harness this potential. We have already seen services that enable collaborative productivity tools as well as the ability to share results of data analysis or visualization. There is a great opportunity for us to actively explore richer ideas in the context of data analytics. For example, what would collaborative data analytics look like in the future? To realize such a vision, we must understand how we can support essential services such as data curation and provenance when we want to perform such activities collaboratively in the cloud. Data sharing in the cloud will also raise new issues in leveraging data sets, such as how to find useful public data, how to correlate your own data with public data to add context, how to find high-quality data in the cloud, and how to share data at fine-grained levels, as well as business issues, such as how to distribute costs when sharing computing and data and how to price data. The cloud will create new life-cycle challenges, such as how to protect data if the current cloud provider fails, or how to preserve data for the long term when it lives "somewhere out there." The cloud will also drive innovation in tools for data governance, such as auditing, enforcement of legal terms and conditions, and explanation of user policies.

Hybrid clouds bring a new set of challenges as well. Today, this entails support for sharing and seamless operation between database services and servers that reside on-premise and those in a single cloud provider. In the future, data sharing services will need to be federated across mobile devices, on-premise resources, and multiple cloud providers. We also need to support common patterns of hybrid clouds, e.g., organizations may run applications in their private cloud during normal operation, but then tap into a public cloud at peak times or when unanticipated events bring surges in load. Another example is cyber-physical systems, as in the Internet of Things, where, e.g., cars will upload data into a cloud and obtain control information in return. Cyber-physical systems involve data streaming from multiple sensors and mobile devices, and must cope with intermittent connectivity and limited battery life, which pose difficult challenges for real-time and perhaps mission-critical data management in the cloud.

2.5 Roles of Humans in the Data Life Cycle

Back when data management was an enterprise-driven activity, it was clear who did what: developers built databases and database-centric applications, business analysts queried databases using (SQL-based) reporting tools, end users generated data and queried and updated databases, and database administrators tuned and monitored databases and their workloads. Today, the world has dramatically changed. A single individual may now play multiple roles in the data life cycle, and many Big Data applications involve people in many different roles. The database research community must address this change, managing not just the data, but the people as well.

There has been a growing recognition of the increasing role of people in the data life cycle, of course, such as the work done in our community and elsewhere on crowdsourcing. However, the new need to "manage the people" is not just about crowdsourcing or micro-tasks (i.e., tasks that take a crowd worker a few minutes to perform). Today's land-

scape requires the consideration of people (and human factors) as they relate to query understanding and refinement, identifying relevant and trustworthy information sources, defining and incrementally refining the data processing pipeline, and visualizing relevant patterns and obtaining query answers, all in addition to making the various micro-tasks doable by domain experts and end users. We can classify people's roles into four general categories: producers of data, curators of data, consumers of data, and community members. Below we discuss each category and its associated data management research challenges.

Many people today are data producers, as virtually anyone can generate a torrent of data now through the sharing of tables, the use of mobile phones, social platforms and applications (e.g., Facebook, Twitter), and an increasing collection of wearable devices (e.g., Fitbit). One key challenge for the database community is to develop algorithms and incentives that guide people to produce and share the most useful data, while maintaining the desired level of data privacy. For instance, when people produce data, how can we help them add metadata quickly and accurately? As one example, when a user uploads an image, Facebook automatically identifies faces in the image so that users can optionally tag them. As another example, there are tools to automatically suggest tags for a tweet. What else can we do, and what general principles and tools can we provide?

More people are becoming data curators. In today's data-driven world there is less central control over data. Data is no longer just in databases controlled by a DBA and curated by the IT department. Instead, as mentioned earlier, a wide variety of data is now being generated, and a wide variety of people are now empowered to curate it. In particular, crowdsourcing has emerged as a promising curation solution. Another key challenge, then, is to obtain high-quality data sets from a process based on often-imperfect human curators. Two related challenges are building platforms that allow people to curate data easily and extending relevant applications to incorporate such curation. For these people-centric challenges, data provenance and explanation will be crucial, as will considerations of privacy and security.

People are data consumers as well. Increasingly, people want to use messier and messier data in complex ways. This raises many challenges. In the enterprise, data consumers have usually been people who know how to ask SQL queries, via a command-line interface or a graphical query tool, over a structured database. Today's data consumers may not know how to formulate a query at all – e.g., a journalist who wants to "find the average temperature of all cities with population exceeding 100,000 in Florida" over a structured data set. Our community's challenge is to make it possible for such people to get their answers themselves, directly. This requires new query interfaces, e.g., interfaces based on multitouch, not just console-based SQL interfaces. We need interfaces that combine visualization, querying, and navigation. Many data consumers may not know what queries to ask, and the available data may or may not support their needs. When the query to ask is not clear, people need other ways to browse, explore, visualize, and mine the data. We must build tools and infrastructures that make the data consumption process easier, including the notions of trust, provenance, and explanation, and we must target the diverse user base of the emerging data-driven world.

People are community members. Numerous communities exist online, with more being created daily. Members of such communities often want to create, share, and manage data, and it is becoming increasingly easy for them to do so. In particular, members may want to collaboratively build community-specific knowledge bases, wikis, and tools to process data. For example, many researchers have created their own pages on Google Scholar, thereby contributing to this "community" knowledge base. Our challenge is to build tools to help communities produce usable data as well as to exploit, share, and mine it.

3. COMMUNITY CHALLENGES

In addition to research challenges, the meeting also discussed a host of community issues. These include database education, research culture, and data science and scientists. Some of these issues are new, brought about by Big Data. Other issues, while not new, are exacerbated by Big Data and are becoming increasingly important for our community to address.

One issue that meeting participants discussed was database education. The way we teach database technology today is increasingly disconnected from reality. We still teach the technology of the 1980's, when memory was small relative to data sizes, making I/O a costly portion of database operation, and when computation was also quite expensive. Today, however, the world looks very different. Technological advances have turned many previous design constraints upside-down. For example, databases can now be entirely memory resident for some classes

of applications; new storage technologies eliminate some of the sequential vs. random I/O issues of the past; and advances in distributed computing have brought us self-managing distributed file systems, scalable parallel data processing techniques, and new declarative languages. While influenced by database languages, these new languages relax some of SQL's rigidity and are compiled down to MapReduce jobs on existing execution platforms.

Despite these changes, we still base our teaching on the architectural blueprint born in the 1970's and 1980's. Similarly, our data model and query language teachings focus on relations and SQL. There was a widely shared sense at the meeting that change in database education is overdue, but no consensus on what this change should be. Some suggested that we start with new technologies, e.g., columnar instead of row-based storage, while others felt that we should move to teaching top-down and explore the alternate technologies available at each architectural decision point, or to start with notions of data quality and value in the bigger picture of going from raw data to knowledge. In addition, functionality once limited to databases and hidden under SQL is now appearing in different contexts and becoming available in smaller, more specialized systems (e.g., key-value stores, stream databases), as well as outside of databases (e.g., the use of hash-based parallelism and scalable external sorting in systems like Hadoop). As a result, there was a feeling that we should be teaching the principles, patterns, and algorithms that have come from years of database research as things whose modern applicability is much broader than just SQL system internals. Other questions raised include: how do we "parcel out" the nice "nuggets" buried in the relational tradition so they are not continually reinvented? What about the role of this material in a computer science education – should it be "ghettoized" in a database class, or be pushed into introductory curricula alongside recursion, divide-and-conquer, and object-oriented programming? If we achieve this, what other material should we substitute in the database class?

Besides database education, there is also a concern regarding our research culture. In recent years there has been an alarming increase in emphasis on publication and citation counts instead of research impact. This discourages large systems projects, end-to-end tool building, and sharing of large data sets due to the longer times required and the resulting lower publication density. Program committees (PCs) often value novelty over utility or potential impact. (Publication pressure has also led to huge PCs with no face-to-face meetings, reducing individual accountability and making it difficult for junior PC members to learn from more senior ones.) These problems jeopardize our Big Data agenda. To pursue this agenda effectively, it is important that we develop and share large systems, end-to-end tools, and data sets, to help evaluate and drive our research, and to have practical impact. The field should strive to return to a state where fewer publications per researcher per time unit is the norm, and where large systems projects, end-to-end tool sets, and data sharing are more highly valued. However, there was no consensus on how best to get there from here – something to grapple with over the incoming years.

Another major change is that Big Data has generated a rapidly growing demand for data scientists: individuals with skills to transform large volumes of data into actionable knowledge. Data scientists need skills not just in data management and large-scale data processing tools and platforms, but also in business intelligence, computer systems, mathematics, statistics, machine learning, and optimization. They also need an ability to work closely with domain experts. In response to this need, some universities are creating data science institutes and degree programs to foster collaboration and assemble the requisite interdisciplinary knowledge and course offerings. The database research community has much to offer such efforts, and we should actively do so. Data science is a cross-disciplinary movement, so participation will require collaborations with domain specialists. Big Data presents computer science with an opportunity to influence the curricula of chemistry, earth sciences, sociology, physics, biology, and many other fields. The small computer science parts in those curricula could be grown and re-adjusted in their focus to give data management and data science a more prominent role.

4. GOING FORWARD

This is an extremely exciting time for database research. In the past we have been guided by, but also restricted by, the rigors of the enterprise, its relational data, and our relational database system architectures. The rise of Big Data and the vision of a data-driven world raise many exciting opportunities and pose many new challenges for the database research community. Being the community that has traditionally dealt with all things related to data, there is a golden opportunity for us to play a central role in this emerging world. There is an abundance of research opportunities related to handling the many challenges of Big Data; of data diversity; of

new hardware, software, and cloud-based platforms; of addressing the data life cycle, from the creation of data to analysis and sharing; and of facing the diversity, roles, and number of people related to all aspects of data. It is also time to rethink our approach to education, our degree of involvement with the consumers of our work, and our value system and its impact on what (and how) we disseminate and how we fund our research.

Acknowledgments: The Beckman meeting was supported financially by donations from the Professor Ram Kumar Memorial Foundation, Microsoft Corporation, and @WalmartLabs.

5. REFERENCES

[BDD+89] Philip A. Bernstein, Umeshwar Dayal, David J. DeWitt, Dieter Gawlick, Jim Gray, Matthias Jarke, Bruce G. Lindsay, Peter C. Lockemann, David Maier, Erich J. Neuhold, Andreas Reuter, Lawrence A. Rowe, Hans-Jorg Schek, Joachim W. Schmidt, Michael Schrefl, Michael Stonebraker. "Future Directions in DBMS Research - The Laguna Beach Participants." *ACM SIGMOD Record*, 18(1):17-26, 1989.

[SSU91] Avi Silberschatz, Michael Stonebraker, Jeff Ullman. "Database Systems: Achievements and Opportunities." *Communications of the ACM*, 34(10):110-120, 1991.

[SSU96] Avi Silberschatz, Mike Stonebraker, Jeff Ullman. "Database Research: Achievements and Opportunities into the 21st Century." *ACM SIGMOD Record*, 25(1):52-63, 1996.

[SZ96] Avi Silberschatz, Stan Zdonik, et al. "Strategic Directions in Database Systems – Breaking Out of the Box." *ACM Computing Surveys*, 28(4):764-778, 1996.

[BBC+98] Phil Bernstein, Michael Brodie, Stefano Ceri, David DeWitt, Mike Franklin, Hector Garcia-Molina, Jim Gray, Jerry Held, Joe Hellerstein, H. V. Jagadish, Michael Lesk, Dave Maier, Jeff Naughton, Hamid Pirahesh, Mike Stonebraker, Jeff Ullman. "The Asilomar Report on Database Research." *ACM SIGMOD Record*, 27(4):74-80, 1998.

[AAB+05] Serge Abiteboul, Rakesh Agrawal, Phil Bernstein, Mike Carey, Stefano Ceri, Bruce Croft, David DeWitt, Mike Franklin, Hector Garcia Molina, Dieter Gawlick, Jim Gray, Laura Haas, Alon Halevy, Joe Hellerstein, Yannis Ioannidis, Martin Kersten, Michael Pazzani, Mike Lesk, David Maier, Jeff Naughton, Hans Schek, Timos Sellis, Avi Silberschatz, Mike Stonebraker, Rick Snodgrass, Jeff Ullman, Gerhard Weikum, Jennifer Widom, Stan Zdonik. "The Lowell Database Research Self-Assessment." *Communications of the ACM*, 48(5):111-118, 2005.

[AAB+09] Rakesh Agrawal, Anastasia Ailamaki, Philip A. Bernstein, Eric A. Brewer, Michael J. Carey, Surajit Chaudhuri, AnHai Doan, Daniela Florescu, Michael J. Franklin, Hector Garcia-Molina, Johannes Gehrke, Le Gruenwald, Laura M. Haas, Alon Y. Halevy, Joseph M. Hellerstein, Yannis E. Ioannidis, Hank F. Korth, Donald Kossmann, Samuel Madden, Roger Magoulas, Beng Chin Ooi, Tim O'Reilly, Raghu Ramakrishnan, Sunita Sarawagi, Michael Stonebraker, Alexander S. Szalay, and Gerhard Weikum. "The Claremont Report on Database Research." *Communications of the ACM*, 52(6):56-65, 2009.