The Claremont Report on Database Research

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, Gerhard Weikum

Abstract

In late May, 2008, a group of database researchers, architects, users and pundits met at the Claremont Resort in Berkeley, California to discuss the state of the research field and its impacts on practice. This was the seventh meeting of this sort in twenty years, and was distinguished by a broad consensus that we are at a turning point in the history of the field, due both to an explosion of data and usage scenarios, and to major shifts in computing hardware and platforms. Given these forces, we are at a time of opportunity for research impact, with an unusually large potential for influential results across computing, the sciences and society. This report details that discussion, and highlights the group's consensus view of new focus areas, including new database engine architectures, declarative programming languages, the interplay of structured and unstructured data, cloud data services, and mobile and virtual worlds. We also report on discussions of the community's growth, including suggestions for changes in community processes to move the research agenda forward, and to enhance impact on a broader audience.

1. A Turning Point in Database Research

Over the last twenty years, small groups of database researchers have periodically gathered to assess the state of the field and propose directions for future research [BDD+89, SSU91, ASU95, AZ+96, BBC+98, AAB03]. Reports of these meetings were written to serve various functions: to foster debate within the database research community, to explain research directions to external organizations, and to help focus community efforts on timely challenges.

This year, the tenor of the meeting was unusual and quite clear: database research and the data management industry are at a turning point, with unusually rich opportunities for technical advances, intellectual achievement, entrepreneurship and impact on science and society. Given the large number of opportunities, it is important for the research community to address issues that maximize impact within the field, across computing, and in external fields as well.

The sense of change in the air emerged quickly in the meeting, as a function of several factors:

1. **Breadth of excitement about Big Data.** In recent years, the number of communities working with large volumes of data has grown considerably, to include not only traditional enterprise applications and Web search, but also "e-science" efforts (in astronomy, biology, earth science, etc.), digital entertainment, natural language processing, social network analysis, and more. While the user base for traditional Database Management Systems (DBMSs) is growing quickly, there is also a groundswell of efforts to design new custom data management solutions from simpler components. The ubiquity of Big Data is significantly expanding the base of both users and developers of data management technologies, and will undoubtedly shake up the field.
2. **Data analysis as a profit center:** In traditional enterprise settings, the barriers between the IT department and business units are quickly dropping, and there are many examples of companies where the data *is* the business. As a consequence, data capture, integration and analysis are no longer considered a business cost; they are the keys to efficiency and profit. The industry supporting data analytics is growing quickly as a result. Corporate acquisitions of Business Intelligence (BI) vendors

alone last year totaled over 10 billion dollars, and that is only the "front end" of the data analytics toolchain. The market pressures for better analytics also bring new users and demands to the technology. Statistically sophisticated analysts are being hired in a growing number of industries, and are increasingly interested in running their formulae on the raw data. At the same time, a growing number of non-technical decision-makers want to "get their hands on the numbers" as well.

3. **Ubiquity of structured and unstructured data.** There is an explosion of structured data available on the Web and on enterprise intranets. This data comes from a variety of sources beyond traditional databases: large-scale efforts to extract structured information from text, software logs and sensors, and crawls of Deep Web sites. There is also an explosion of text-focused semi-structured data in the public domain in the form of blogs, Web 2.0 communities and instant messaging. And new incentive structures and web sites have emerged for publishing and curating structured data in a shared fashion as well. Current text-centric approaches to managing this data are easy to use, but ignore latent structure in the data that can add significant value. The race is on to develop techniques that can extract useful data from mostly noisy text and structured corpora, enable deeper explorations into individual datasets, and connect datasets together to wring out as much value as possible.
4. **Expanded developer demands.** Programmer adoption of relational DBMSs and query languages has grown significantly in recent years. This has been accelerated by the maturation of open source systems like MySQL and PostgreSQL, and the growing popularity of object-relational mapping packages like Ruby on Rails. However, the expanded user base brings new expectations for programmability and usability from a larger, broader and less specialized community of programmers. Some of these developers are unhappy or unwilling to "drop into" SQL, and view DBMSs as heavyweight to learn and manage relative to other open source components. As the ecosystem for database management evolves further beyond the typical DBMS user base, opportunities emerge for new programming models and for new system components for data management and manipulation.
5. **Architectural shifts in computing.** At the same time that user scenarios are expanding, computing substrates for data management are shifting rapidly. At the macro scale, the rise of "cloud" computing services suggests fundamental changes in software architecture. It democratizes access to parallel clusters of computers: every programmer now has the opportunity and motivation to design systems and services that can scale out incrementally to arbitrary degrees of parallelism. At a micro scale, computer architectures have shifted the focus of Moore's Law from increasing clock speed per chip to increasing the number of processor cores and threads per chip. In storage technologies, major changes are underway in the memory hierarchy, due to the availability of more and larger on-chip caches, large inexpensive RAM, and flash memory. Power consumption has become an increasingly important aspect of the price/performance metric of large systems. These hardware trends alone motivate a wholesale reconsideration of data management software architecture.

Taken together, these factors signal an urgent, widespread need for new data management technologies. The opportunity for impact is enormous.

Traditionally, the database research community is known for impact: relational databases are emblematic of technology transfer. But in recent years, our externally visible impact has not evolved sufficiently beyond traditional database systems and enterprise data management, despite the expansion of our research portfolio. In the current climate, the community must recommit itself to impact and breadth. Impact is evaluated by external measures, so success will involve helping new classes of users, powering new computing platforms, and making conceptual breakthroughs across computing. These should be the motivating goals for the next round of database research.

To achieve these goals, two promising approaches that came up in discussion are what we call *reformation* and *synthesis*. The reformation agenda involves deconstructing core data-centric ideas and systems, and re-forming them for new applications and architectural realities. Part of this entails focusing outside the traditional RDBMS stack and its existing interfaces, emphasizing new data management systems for growth areas like e-science. In addition, database researchers should take data-centric ideas (declarative programming, query optimization) outside their original context in storage and retrieval, and attack new areas of computing where a data-centric mindset can have major impact. The synthesis agenda is intended to leverage good research ideas in areas that have yet to develop identifiable, agreed-upon system architectures, e.g., data integration, information extraction, data privacy, etc. The time is ripe for various

sub-communities to move out of the conceptual and algorithmic phase, and work together on comprehensive artifacts (systems, languages, services) that combine multiple techniques to solve complex user problems. Efforts toward synthesis can serve as rallying points for the research, will likely lead to new challenges and breakthroughs, and can increase the overall visibility of the work.

2. Research Opportunities

After two days of intense discussion, it was surprisingly easy for the group to reach consensus on a set of research topics to highlight for investigation in coming years. This is indicative of unusually exciting times.

Before presenting those topics, we stress a few points regarding what is *not* on this list. First, while we tried to focus on new opportunities, we do not propose they be pursued at the expense of existing good work. A number of areas we deemed critical were left out of this list because they have already become focus topics in the community. Many of these were mentioned in a previous report of this sort (see the Appendix), and/or are the subject of significant efforts in recent years. These ongoing efforts require continued investigation and funding. Second, we chose to keep the list short, favoring focus over coverage. Most of the authors have other promising research topics they would have liked to discuss at greater length here, but we chose to focus on topics that attracted the broadest interest in the group.

In addition to the list below, the main issues and areas that were raised repeatedly during the meeting include management of uncertain information, data privacy and security, e-science and other scholarly applications, human-centric interactions with data, social networks and Web 2.0, personalization and contextualization of query- and search-related tasks, streaming and networked data, self-tuning and adaptive systems, and the challenges raised by new hardware technologies and energy constraints. Most of these issues are in fact captured in some aspect of the discussion below, and many of them cut across multiple highlighted topics.

2.1. Revisiting Database Engines

System R and Ingres pioneered the architecture and algorithms of relational databases, and current commercial databases are still based on their designs. But the many changes in applications and technology described in Section 1 demand a reformation of the entire system stack for data management. Current big-market relational database systems have well-known limitations. While they provide a broad range of features, they have very narrow regimes where they provide peak performance: OLTP systems are tuned for lots of small, concurrent transactional debit/credit workloads, while decision-support systems are tuned for few read-mostly, large join and aggregation workloads. Meanwhile, there are many popular data-intensive tasks from the last decade for which relational databases provide poor price/performance and have been rejected: critical scenarios include text indexing, serving web pages, and media delivery. New workloads are emerging in sciences and Web 2.0-style applications, among other environments, where database engine technology could prove useful, but not as bundled in current database systems.

Even within traditional application domains, the current marketplace suggests that there is room for significant innovation. In the analytics markets for business and science, customers can buy petabytes of storage and thousands of processors, but the dominant commercial database systems cannot scale that far for many workloads. Even when they can, the cost of software and management relative to hardware is exorbitant. In the on-line transaction processing (OLTP) market, business imperatives like regulatory compliance and rapid response to changing business conditions raise the need to address data lifecycle issues such as data provenance, schema evolution, and versioning.

Given all these requirements, the commercial database market is wide open to new ideas and systems, and this is reflected in the funding climate for entrepreneurs. It is hard to remember a time when there were so many database engine startup companies. The market will undoubtedly consolidate over time, but things are changing fast right now, and it is a good time to try radical ideas.

Some research projects have begun taking revolutionary steps in database system architecture. There have been two distinct directions: broadening the useful range of applicability for multi-purpose database systems (e.g., to incorporate streams, text search, XML, information integration), and radically improving performance by designing special-purpose database systems for specific domains (e.g., streams, read-mostly analytics, and XML.) Both directions have merit, and the evident commonality of focus suggests that these efforts may be synergistic: special-purpose techniques (e.g., new storage/compression formats) may be reusable in more general-purpose systems, and general-purpose architectural components or harnesses (e.g., extensible query optimizer frameworks) may enable new special-purpose systems to be prototyped more quickly.

Important research topics in the core database engine area include: (a) designing systems for clusters of many-core processors, which will exhibit limited and non-uniform access to off-chip memory; (b) exploiting remote RAM and Flash as persistent media, rather than relying solely on magnetic disk; (c) treating query optimization and physical data layout as a unified, adaptive, self-tuning task to be carried out continuously; (d) compressing and encrypting data at the storage layer, integrated with data layout and query optimization; (e) designing systems that embrace non-relational data models, rather than "shoehorning" them into tables; (f) trading off consistency and availability for better performance and scaleout to thousands of machines; (g) designing power-aware DBMSs that limit energy costs without sacrificing scalability.

That list of topics is not exhaustive. One industrial participant noted that this is a time of particular opportunity for academic researchers: the landscape has shifted enough that access to industrial legacy code provides little advantage, and large-scale clustered hardware is now rentable in "the cloud" at low cost. Moreover, industrial players and investors are actively looking for bold new ideas. This opportunity for academics to lead in system design is a major change in the research environment.

2.2. Declarative Programming for Emerging Platforms

Programmer productivity is a key challenge in computing. This has been acknowledged for many years, with the most notable mention in the database context being in Jim Gray's Turing lecture of ten years ago. Today, the urgency of the problem is literally increasing exponentially as programmers target ever more complex environments, including manycore chips, distributed services, and cloud computing platforms. Non-expert programmers need to be able to easily write robust code that scales out across processors in both loosely- and tightly-coupled architectures.

Although developing new programming paradigms is not a database problem per se, ideas of data independence, declarative programming and cost-based optimization provide a promising angle of attack. There is significant evidence that data-centric approaches can have major impact on programming in the near term.

The recent popularity of Map-Reduce is one example of this potential. Map-Reduce is attractively simple, and builds on language and data-parallelism techniques that have been known for decades. For database researchers, the significance of Map-Reduce is in demonstrating the benefits of data-parallel programming to new classes of developers. This opens opportunities for our community to extend its impact, by developing more powerful and efficient languages and runtime mechanisms that help these developers address more complex problems.

As another example, new declarative languages, often grounded in Datalog, have recently been developed for a variety of domain-specific systems, in fields as diverse as networking and distributed systems, computer games, machine learning and robotics, compilers, security protocols, and information extraction. In many of these scenarios, the use of a declarative language reduced code size by orders of magnitude, while also enabling distributed or parallel execution. Surprisingly, the groups behind these various efforts have coordinated very little – the move to revive declarative languages in these new contexts has grown up organically.

A third example arises in enterprise application programming. Recent language extensions like Ruby on Rails and LINQ encourage query-like logic in programmer design patterns. But these packages have yet to seriously address the challenge of programming across multiple machines. For enterprise applications, a key distributed design decision is the partitioning of logic and data across multiple "tiers": web clients, web servers, application servers, and a backend DBMS. Data independence is particularly valuable here, to allow programs to be specified without making a priori, permanent decisions about physical deployment across tiers. Automatic optimization processes could make these decisions, and move data and code as needed to achieve efficiency and correctness. XQuery has been proposed as one existing language that can facilitate this kind of declarative programming, in part because XML is often used in cross-tier protocols.

It is unusual to see this much energy surrounding new data-centric programming techniques, but the opportunity brings challenges as well. Among the research questions we face are language design, efficient compilers and runtimes, and techniques to optimize code automatically across both the horizontal distribution of parallel processors, and the vertical distribution of tiers. It seems natural that the techniques behind parallel and distributed databases – partitioned dataflow, cost-based query optimization – should extend to new environments. However, to succeed, these languages will have to be fairly expressive, going beyond simple Map-Reduce and Select-Project-Join-Aggregate dataflows. There is a need for "synthesis" work here to harvest useful techniques from the literature on database and logic programming languages and optimization, and to realize and extend them in new programming environments.

To have impact, our techniques also need to pay attention to the softer issues that capture the hearts and minds of programmers, such as attractive syntax, typing and modularity, development tools, and smooth interactions with the rest of the computing ecosystem (networks, files, user interfaces, web services, other languages, etc.)

Attacking this agenda requires database research to look outside its traditional boundaries and find allies across computing. It is a unique opportunity for a fundamental "reformation" of the notion of data management: not as a storage service, but as a broadly applicable programming paradigm.

2.3. The Interplay of Structured and Unstructured Data

A growing number of data management scenarios involve both structured and unstructured data. Within enterprises, we see large heterogeneous collections of structured data linked with unstructured data such as document and email repositories. On the World-Wide Web, we are witnessing a growing amount of structured data coming primarily from three sources: (1) millions of databases hidden behind forms (the *deep web*), (2) hundreds of millions of high-quality data items in HTML tables on web pages, and a growing number of mashups providing dynamic views on structured data, and (3) data contributed by Web 2.0 services, such as photo and video sites, collaborative annotation services and online structured-data repositories.

A significant long-term goal for our community is to transition from managing traditional databases consisting of well-defined schemata for structured business data, to the much more challenging task of managing a rich collection of structured, semi-structured and unstructured data, spread over many repositories in the enterprise and on the Web. This has sometimes been referred to as the challenge of managing dataspaces.

On the Web, our community has contributed primarily in two ways. First, we developed technology that enables the generation of domain-specific ("vertical") search engines with relatively little effort. Second, we developed domain-independent technology for crawling through forms (i.e., automatically submitting well-formed queries to forms) and surfacing the resulting HTML pages in a search-engine index. Within the enterprise, we have recently made contributions to enterprise search and the discovery of relationships between structured and unstructured data.

The first challenge we face is to extract structure and meaning from unstructured and semi-structured data. Information Extraction technology can now pull structured entities and relationships out of unstructured text, even in unsupervised web-scale contexts. We expect hundreds of extractors being applied to a given

data source. Hence we need techniques for applying and managing predictions from large numbers of independently developed extractors. We also need algorithms that can introspect about the correctness of extractions and therefore combine multiple pieces of extraction evidence in a principled fashion. We are not alone at these efforts; to contribute in this area, the community should continue to strengthen its ties with the Information Retrieval and Machine Learning communities.

A significant aspect of the semantics of the data is its context. The context can have multiple forms, such as the text and hyperlinks that surround a table on a web page, the name of the directory in which data is stored and accompanying annotations or discussions, and relationships to physically or temporally proximate data items. Context helps interpret the meaning of data in such applications because the data is often less precise than in traditional database applications since it is extracted from unstructured text, is extremely heterogeneous, or is sensitive to the conditions under which it was captured. Better database technology is needed to manage data in context. In particular, we need techniques to discover data sources, to enhance the data by discovering implicit relationships, to determine the weight of an object's context when assigning it semantics, and to maintain the provenance of data through these various steps of storage and computation.

The second challenge is to develop methods for effectively querying and deriving insight from the resulting sea of heterogeneous data. A specific problem is to answer keyword queries over large collections of heterogeneous data sources. We need to analyze the query to extract its intended semantics, and route the query to the relevant sources(s) in the collection. Of course, keyword queries are just one entry point into data exploration, and there is a need for techniques that lead users into the most appropriate querying mechanism. Unlike previous work on information integration, the challenges here are that we do not assume we have semantic mappings for the data sources and we cannot assume that the domain of the query or the data sources is known. We need to develop algorithms for providing *best-effort* services on loosely integrated data. The system should provide some meaningful answers to queries with no need for any manual integration, and improve over time in a "pay-as-you-go" fashion as semantic relationships are discovered and refined. Developing index structures to support querying hybrid data is also a significant challenge. More generally, we need to develop new notions of correctness and consistency in order to provide metrics and to enable users or system designers to make cost/quality tradeoffs. We also need to develop the appropriate systems concepts around which to tie these functionalities.

In addition to managing existing data collections, we also have an opportunity to innovate on *creating* data collections. The emergence of Web 2.0 creates the potential for new kinds of data management scenarios in which users join ad-hoc communities to create, collaborate, curate and discuss data online. Since such communities will rarely agree on schemata ahead of time, they will need to be inferred from the data and will be highly dynamic; however they will still be used to guide users to consensus. Systems in this context need to incorporate visualizations effectively, because visualizations drive the exploration and analysis. Most importantly, these systems need to be extremely easy to use. This will probably require compromising on some typical database functionality and providing more semi-automatic "hints" that are mined from the data. There is an important opportunity for a feedback loop here – as more data gets created with such tools, information extraction and querying could become easier. Commercial and academic prototypes are beginning to appear in this arena, but there is plenty of space for additional innovation and contributions.

2.4. Cloud Data Services

Economic factors are leading to the rise of infrastructures providing software and computing facilities as a service, typically known as *cloud* services or cloud computing. Cloud services can provide efficiencies for application providers, both by limiting up-front capital expenses, and by reducing the cost of ownership over time. Such services are typically hosted in a data center, using shared commodity hardware for computation and storage. There is a varied set of cloud services available today, including application services (salesforce.com), storage services (Amazon S3), compute services (Google App Engine, Amazon EC2) and data services (Amazon SimpleDB, Microsoft SQL Server Data Services, Google's Datastore). These services represent a variety of reformations of data management architectures, and more are on the horizon. We anticipate that many future data-centric applications will leverage data services in the cloud.

A cross-cutting theme in cloud services is the trade-off that providers face between functionality and operational costs. Today's early cloud data services offer an API that is much more restricted than that of traditional database systems, with a minimalist query language and limited consistency guarantees. This pushes more programming burden on developers, but allows cloud providers to build more predictable services, and offer service level agreements that would be hard to provide for a full-function SQL data service. More work and experience will be needed on several fronts to explore the continuum between today's early cloud data services and more full-functioned but possibly less predictable alternatives.

Manageability is particularly important in cloud environments. Relative to traditional systems, it is complicated by three factors: limited human intervention, high-variance workloads, and a variety of shared infrastructures. In the majority of cases, there will be no DBAs or system administrators to assist developers with their cloud-based applications; the platform will have to do much of that work automatically. Mixed workloads have always been difficult to tune, but may be unavoidable in this context. Even a single customer's workload can vary widely over time: the elastic provisioning of cloud services makes it economical for a user to occasionally harness orders of magnitude more resources than usual for short bursts of work. Meanwhile, service tuning depends heavily upon the way that the shared infrastructure is "virtualized". For example, Amazon EC2 uses hardware-level virtual machines as the programming interface. On the opposite end of the spectrum, salesforce.com implements "multi-tenant" hosting of many independent schemas in a single managed DBMS. Many other virtualization solutions are possible. Each has different visibility into the workloads above and platforms beneath, and different abilities to control each. These variations will require revisiting traditional roles and responsibilities for resource management across layers.

The need for manageability adds urgency to the development of self-managing database technologies explored in the last decade. Adaptive, online techniques will be required to make these systems viable, while new architectures and APIs – including the flexibility to depart from traditional SQL and transactional semantics when prudent – may motivate increasingly disruptive approaches to adaptivity.

The sheer scale of cloud computing presents it own challenges. Today's SQL databases simply cannot scale to the thousands of nodes being deployed in the cloud context. On the storage front, it is unclear whether to address these limitations with different transactional implementation techniques, different storage semantics, or both. The database literature is rich in proposals on these issues. Current cloud services have begun to explore some simple pragmatic approaches, but more work is needed to synthesize ideas from the literature in modern cloud computing regimes. In terms of query processing and optimization, it will not be feasible to exhaustively search a plan space that considers thousands of processing sites, so some limitations on either the plan space or the search will be required. Finally, it is unclear how programmers will express their programs in the cloud, as mentioned in Section 2.2. More work is needed to understand the scaling realities of cloud computing – both performance constraints and application requirements – to help navigate this design space.

The sharing of physical resources in a cloud infrastructure puts a premium on data security and privacy, which cannot be guaranteed by physical boundaries of machines or networks. Hence cloud services provide fertile ground for efforts to synthesize and accelerate the work our community has done in these domains. The key to success in this arena will be to specifically target usage scenarios in the cloud, seated in practical economic incentives for service providers and customers.

As cloud data services become popular, we expect that new scenarios will emerge with their own challenges. For example, we anticipate the appearance of specialized services that are pre-loaded with large data sets, e.g., stock prices, weather history, web crawls, etc. The ability to "mash up" interesting data from private and public domains will be increasingly attractive, and will provide further motivation for the challenges in Section 2.3. This also points to the inevitability of services reaching out across clouds. This issue is already prevalent in scientific data "grids", which typically have large shared data servers at multiple different sites, even within a single discipline. It also echoes, in the large, the standard proliferation of data sources in most enterprises. Federated cloud architectures will only enhance the challenges described above.

2.5. Mobile Applications and Virtual Worlds

There is a new class of applications, exemplified by mobile services and virtual worlds, characterized by the need to manage massive amounts of diverse user-created data, synthesize it intelligently, and provide real-time services. The data management community is beginning to understand the challenges these applications face, but much more work is needed. Accordingly, the discussion about these topics at the meeting was more speculative than about those of the previous sections, but we felt they deserve attention.

In the mobile space, we are witnessing two important trends. First, the platforms on which to build mobile applications (i.e., the hardware, software and network) are maturing to the point that they have attracted large user bases, and can ubiquitously support very powerful interactions "on the go". Second, the emergence of mobile search and social networks suggests an exciting new set of mobile applications. These applications will deliver timely information (and advertisements) to mobile users depending on their location, personal preferences, social circles and extraneous factors (e.g., weather), and in general the context in which they operate. Providing these services requires synthesizing user input and behavior from multiple sources to determine user location and intent.

Virtual worlds like Second Life are growing quickly in popularity, and in many ways mirror the themes of mobile applications. While they began as interactive simulations for multiple users, they increasingly blur the distinctions with the real world, and suggest the potential for a more data-rich mixture. The term *co-space* is sometimes used to refer to a co-existing space for both virtual and physical worlds. In a co-space, locations and events in the physical world are captured by a large number of sensors and mobile devices, and materialized within a virtual world. Correspondingly, certain actions or events within the virtual world can have effects in the physical world (e.g., shopping or product promotion and experiential computer gaming). Applications of co-space include rich social networking, massive multi-player games, military training, edutainment and knowledge sharing.

In both of these areas, large amounts of data are flowing from users, being synthesized and used to affect the virtual and/or real world. These applications raise new challenges, such as a need to process heterogeneous data streams in order to materialize real-world events, the need to balance privacy against the collective benefit of sharing personal real-time information, and the need for more intelligent processing to send interesting events in the co-space to someone in the physical world. The programming of virtual actors in games and virtual worlds requires large-scale parallel programming, and declarative methods have been proposed as a solution in that environment as discussed in Section 2.2. These applications also require the development of efficient systems as suggested in Section 2.1, including appropriate storage and retrieval methods, data processing engines, parallel and distributed architectures, and power-sensitive software techniques for managing the events and communications across huge number of concurrent users.

3. Moving Forward

In addition to research topics, the meeting involved discussions of the research community's processes, including the organization of publication procedures, research agendas, attraction and mentorship of new talent, and efforts to ensure research impact.

Prior to these discussions, a bit of ad hoc data analysis was performed over database conference bibliographies from the DBLP repository. While the effort was not scientific, the results indicated that the database research community has *doubled in size over the last decade.* Various metrics suggested this: the number of published papers, the number of distinct authors, the number of distinct institutions to which these authors belong, and the number of session topics at conferences, loosely defined. This served as a backdrop to the discussion that followed.

The community growth is placing pressure on research publications. At a topical level, the increasing technical scope of the community makes it difficult to keep track of the field. As a result, survey articles and tutorials are becoming an increasingly important contribution to the community. They should be encouraged both informally within the community, and via professional incentive structures such as tenure

and promotion. In terms of processes, the reviewing load for papers is growing increasingly burdensome, and there was a perception that the quality of reviews had been decreasing over time. It was suggested that the lack of face-to-face PC meetings in recent years has exacerbated the problem of poor reviews, and removed opportunities for risky or speculative papers to be championed effectively over well-executed but more pedestrian work. Recent efforts to enhance the professionalism of papers and the reviewing process were discussed in this context. Many participants were skeptical that these efforts have had a positive effect on long-term research quality, as measured in intellectual and practical impact. At the same time, it was acknowledged that the community's growth increases the need for clear and clearly-enforced academic processes. The challenge going forward is to find policies that simultaneously reward big ideas and risk-taking, while providing clear and fair rules for achieving those rewards. The publication venues would do well to focus on the first of those goals as much as they have focused recently on the second.

In addition to tuning the mainstream publication venues, there is opportunity to take advantage of other channels of communication. The database research community has had little presence in the relatively active market for technical books. Given the growing population of developers working with big data sets, there is a need for approachable books on scalable data management algorithms and techniques that programmers can use to build their own software. The current crop of college textbooks is not targeted at that market. There is also an opportunity to present database research contributions as big ideas in their own right, targeted at intellectually curious readers outside the specialty. In addition to books, electronic media like blogs and wikis can complement technical papers, by opening up different stages of the research lifecycle to discussion: status reports on ongoing projects, concise presentation of big ideas, vision statements and speculation. Online fora can also spur debate and discussion, if made appropriately provocative. Electronic media underscore the modern reality that it is easy to be widely published, but much harder to be widely *read*. This point should be remembered in the mainstream publication context as well, both by authors and reviewers. In the end, the consumers of an idea define its impact.

Given the growth in the database research community, the time is ripe for ambitious community-wide projects to stimulate collaboration and cross-fertilization of ideas. One proposal is to foster more data-driven research by building a globally shared collection of structured data, accepting contributions from all parties. Unlike previous efforts in this vein, the collection should not be designed for any particular benchmark – in fact, it is likely that most of the interesting problems suggested by this data are yet to be identified. There was also discussion of the role of open source software development in the database community. Despite a tradition of open-source software, academic database researchers at different institutions have relatively rarely reused or shared software. Given the current climate, it might be useful to move more aggressively toward sharing software, and collaborating on software projects across institutions. Information integration was mentioned as an area in which such an effort is emerging. Finally, interest was expressed in technical competitions akin to the Netflix challenge and KDD Cup competitions. To kick this effort off in the database domain, two areas were identified as ripe for competitions: system components for cloud computing (likely measured in terms of efficiency), and large-scale information extraction (likely measured in terms of accuracy and efficiency). While it was noted that each of these proposals requires a great deal of time and care to realize, several participants at the meeting volunteered to initiate efforts in these various directions. That work has begun, and participation from the broader community will be needed to help it succeed.

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-Jörg Schek, Joachim W. Schmidt, Michael Schrefl, and Michael Stonebraker. "Future Directions in DBMS Research - The Laguna Beach Participants". *SIGMOD Record* 18(1): 17-26, 1989.

[SSU91] Abraham Silberschatz, Michael Stonebraker, and Jeffrey D. Ullman. "Database Systems: Achievements and Opportunities". *CACM* 34(10): 110-120, 1991.

[ASU95] Abraham Silberschatz, Michael Stonebraker, Jeffrey D. Ullman: Database Research: Achievements and Opportunities Into the 21st Century. *SIGMOD Record* 25(1): 52-63 (1996)

[AZ+96] Avi Silberschatz, Stan Zdonik, et al., "Strategic Directions in Database Systems—Breaking Out of the Box," *ACM Computing Surveys*, Vol. 28, No. 4 (Dec 1996), 764-778.

[BBC+98] Philip A. Bernstein, Michael L. Brodie, Stefano Ceri, David J. DeWitt, Michael J. Franklin, Hector Garcia-Molina, Jim Gray, Gerald Held, Joseph M. Hellerstein, H. V. Jagadish, Michael Lesk, David Maier, Jeffrey F. Naughton, Hamid Pirahesh, Michael Stonebraker, and Jeffrey D. Ullman. "The Asilomar Report on Database Research". *SIGMOD Record* 27(4): 74-80, 1998.

[AAB+03] Serge Abiteboul, Rakesh Agrawal, Philip A. Bernstein, Michael J. Carey, Stefano Ceri, W. Bruce Croft, David J. DeWitt, Michael J. Franklin, Hector Garcia-Molina, Dieter Gawlick, Jim Gray, Laura M. Haas, Alon Y. Halevy, Joseph M. Hellerstein, Yannis E. Ioannidis, Martin L. Kersten, Michael J. Pazzani, Michael Lesk, David Maier, Jeffrey F. Naughton, Hans-Jörg Schek, Timos K. Sellis, Avi Silberschatz, Michael Stonebraker, Richard T. Snodgrass, Jeffrey D. Ullman, Gerhard Weikum, Jennifer Widom, and Stanley B. Zdonik. "The Lowell Database Research Self-Assessment". *CoRR* cs.DB/0310006, 2003. Also in *CACM* 48(5): 111-118, 2005.

Appendix: Topics From Past Self-Assessments

Meetings to assess the state of database research were held in 1988 [BDD+89], 1990 [SSU91], 1995 [ASU96], 1996 [AZ+], 1998 [BBC+98], and 2003 [AAB+03]. Each report describes changes in the application and technology landscape that motivate the need for new research. We summarize the driving forces in Table 1.

Table 1 External Forces Driving the Database Field in Each Assessment

Year	Driving Forces
1988	Future Applications: CASE, CIM, images, spatial, information retrieval
1990	Future Applications: NASA data, CAD, genetics, data mining, multimedia
1995	Future Applications: NASA data, e-commerce, health care, digital publishing, collaborative design Technology Trends: hardware advances, database architecture changes (client-server, object-relational), the Web
1996	Future Applications: instant virtual enterprise, personal information systems
1998	Technology Trends: the Web, unifying program logic and database systems, hardware advances (scale up to megaservers, scale down to appliances)
2003	Future Applications: cross-enterprise applications, the sciences Technology Trends: hardware advances, maturation of related technologies (data mining, information retrieval)

Each report then goes on to enumerate particular research problems that need more investigation. Not surprisingly, many database research problems reappear in multiple reports. Usually, each occurrence is in the context of a different application scenario. For example, information integration has been recommended in the context of heterogeneous distributed databases (1990), better information distribution (1995), web-scale database integration (1998) and on-the-fly fusion of sensor data (2003). Although the topic recurs, the technical goals in each scenario usually differ. In Table 2, we summarize these recurring topics.

In many cases, these topics later became major database research fields. Examples include data mining, multimedia, integrating information retrieval and databases, data provenance, sensors and streaming, and probabilistic databases. It is impossible to know the extent to which these reports were a factor in these developments.

Some reports were more outwardly focused to non-database researchers. These reports summarized the field's major accomplishments and pointed to worthwhile on-going research topics. We did not include

them in Table 1, which focuses only on areas that were felt to be under-researched at the time of the assessment report.

Necessarily, we applied a fair bit of editorial judgment in grouping topics. There were some topics that were recommended in one report but did not naturally group with topics in other reports. They are listed here for completeness: logical DB design tools, accounting and billing, site autonomy, operating system support for databases, personalization, and scientific data management.

Table 2 Recurring Topics in Database Research Assessment Meetings

	1988	1990	1995	1996	1998	2003
Version & configuration management, repositories	×	×	×			
More data types: Image, spatial, time, genetics, …	×	×	×			
Information retrieval	×		×			×
Extendible DBMSs, object-oriented DBMSs	×			×		
Exploit hardware advances	×				×	
Query optimization	×			×	×	×
Parallelism, scale-up, scale-out	×	×				×
Automated database administration	×		×		×	×
High availability, replication	×		×			
Workflow models, long transactions, workflow engines	×	×	×	×	×	
Active databases, rules, scalable trigger system	×	×			×	×
Heterogeneous DBMSs, interoperation, semantic consistency, data fusion, data provenance, data warehouses, mediators, info discovery, information exchange	×	×	×	×	×	×
Uncertain and probabilistic data, data quality, query processing as evidence accumulation		×	×	×	×	×
Schema-less DBs, integrating structured & semi-structured data, DBMS architecture to integrate text, data, code and streams				×	×	×
Security and privacy, trustworthiness			×	×		×
Data mining		×	×	×		×
Easier application development, visual programming tools, programming language interfaces, component models	×			×	×	
Tertiary storage, 100 year storage		×				×
Real-time DBs, streams, sensor networks	×					×
Multimedia: quality of service, queries, UI support		×	×	×		×
User interfaces for DBs	×		×			×