NSF Workshop on Industrial/Academic Cooperation in Database Systems

Home Page: http://www.ccs.neu.edu/groups/IEEE/ind-acad/webpage.html

Authors: Mike Carey (IBM Almaden), Len Seligman (MITRE)

Attendees: Betty Salzberg (chairperson), Ashish Gupta, Joe Hellerstein, Mike Carey, Anil Nori, Eric Brewer, Hamid Pirahesh, James Hamilton, John Cherniavsky, Maria Zemankova, Marie-Ann Neimat, Mike Ubell, Paul Durdik, Paula Hawthorn, Phil Bernstein, Todd Walters, Brad Adelberg, Clement Yu, Daniel Barbara, Eric Hanson, Jeff Ullman, Ken Ross, Len Seligman, Meral Ozsoyoglu, Panos Chrysanthis, Peter Buneman, Praveen Seshadri, Richard Snodgrass, Shashi Shekhar, Vassilis Tsotras, Victor Vianu, Wesley Chu

1. Introduction

Many academic researchers in computer science want their research to be of relevance to industry. They would like to work on topics that enhance products and data-intensive applications. However, often they are unable to discover which topics fall in that category. In addition, academics have incentives to write papers which will be easy to get accepted for publication in good journals and conferences. This normally requires algorithmic and/or mathematical content. The problem is to find projects which yield publishable papers yet are of interest to industry. One solution is for academics to listen to industrial researchers describe projects and problems with significant intellectual, possibly mathematical or scientific, content. But researchers in industry have little incentive to give such talks. They must produce products and patents, not papers.

In an effort to bridge this gap, the National Science Foundation sponsored a workshop on Industrial/Academic Cooperation in Database Systems. The workshop was held on October 29-30, 1998 in Los Gatos, California. The objectives of the workshop were:

- To enable technical people in DBMS companies to share with academics some of the interesting intellectual problems which have arisen while implementing large-scale DBMSs,
- To find problems of mutual interest for possible collaboration,
- To discuss ways that DBMS companies can enable academics to make their research reflect advances in current practice, and
- To discuss mechanisms for longer term exchanges of information.

The organizing committee consisted of Betty Salzberg – chairperson (Northeastern Univ.), Mike Carey (IBM Almaden), Joseph Hellerstein (U.C. Berkeley), and Ashish Gupta (Amazon.com).

This report describes highlights of the workshop, including industrial presentations, a panel, presentations by academic attendees, and recommendations for activities to foster industrial/academic cooperation in the future.

2. Industrial Presentations

Most of the workshop consisted of invited presentations by technical leaders in the database industry. Slides from the presentations are available at:

http://www.ccs.neu.edu/groups/IEEE/ind-acad/presentations.html

Phil Bernstein (Microsoft)

The first speaker was Phil Bernstein, and his topic was repositories and metadata management research. Phil said this is an underdeveloped area of database research, representing a largely untapped business opportunity of $1B or more. He cited a number of applications that need serious help with managing metadata, including scientific data management, CAD/CAM, warehouse design, web sites, workflow, documents, heterogeneous database integration, configuration management, and application development systems/tools.

Phil defined a Repository as being a database of information about engineered artifacts, global across tools, containing metadata and other design information. He explained that a typical architecture has 4 layers; from the bottom up, the layers are: (1) a standard OODBMS/RDBMS, (2) the repository manager (which adds relationships, properties, etc.—i.e., a semantic model—plus powerful facilities for configuration and version management), (3) an information model including predefined types useful for key repository applications (e.g., data warehousing), and (4) model-driven tools including a browser, scripting languages, data translators, a model editor, and a component manager. He listed related technologies, which include DBMS catalogs, OS software registries, directory services, web site metadata, type libraries, TP monitors, and heterogeneous database systems (given that heterogeneity is the norm in the repository world).

Phil described interesting open problems in the repository area, including Well-formed Problems (i.e., "develop a fast algorithm to do X"), Needs (i.e., "it hurts here"), and (3) Opportunities (i.e., "wouldn't it be great, if ...?"). One of the well-formed problems is support for additional relationship semantics. Phil cited the work of James Rumbaugh as being relevant here. Phil mentioned that many repository operations need to propagate to incident relationships and objects, and that a framework for this would be very helpful. Propagation semantics need to be specified at the schema level. He would like to see customizable operations; challenges include dealing with termination, well-definedness, and efficiency of propagation. He also mentioned version management as a problem that remains open to this day.

Another open problem is support for schema merging. Support is needed for merging of complex structures, including an import model, and model evolution. He would like generic (parameterized, with the right parameters) schema merge facilities with user feedback. At this point Peter Buneman asked why schema merging research hasn't had industrial impact. Another area mentioned by Phil relates to repository replication, sharing, and check-in/check-out. Support is needed for layers of versioned repositories, including object-level versions, nested configurations, and for disconnected operation. There are some commercial products in this space, and Phil said that an in-depth and/or comparative study of such products could be a very useful contribution. Yet another area is closure management. Phil listed computation of closure semantics, and handling things like nested security groups, nested workspaces, and configuration-scoped relationships, as issues in this area.

Phil mentioned that there are several products with some repository functionality, but there is not a good framework for evaluating them. Products include repository engines, such as Platinum, Viasoft, Unisys, and Softlab, as well as model-driven tools, such as Rational Rose, Platinum Paradigm Plus, ERwin, and Select Component Factory. A valuable contribution would be a survey paper describing, in some depth, actual repository products. Writing such a paper would require a model for what products do (i.e., framework development), which would also be a contribution.

Next, Phil described areas of great need (as opposed to particular well-defined problems). The first of these was data scrubbing, which is the majority of work involved in data warehousing today. The state of the art is very primitive; typically, spreadsheets are used to list all the data sources and document—in text form—the data transformations involved in loading and updating the warehouse. This needs to be automated, and model-driven, in the future; this involves schema merging and constraint checking, etc. Several members of the audience (Todd Walters and Peter Buneman) added that the problem is even worse than Phil portrayed it: there are virtually no tools used for this today, data sources are highly varied, etc. The next area presented was large-scale schema evolution. Phil explained that large application products (e.g., SAP/Baan/Peoplesoft) involve hundreds or thousands of tables and classes. Different users customize these at their sites, causing a major release-to-release migration problem—i.e., how can application vendors provide new releases, making schema changes, without forcing their customers to re-customize the new release from scratch? Tools and approaches are needed here. Next, Phil discussed the problem of semantic data integration. Users today need to integrate complex, independently-developed databases and applications. Flexibility is a must today due to corporate mergers, ever-changing government regulations, competition, and increasing time-to-market pressures. Data warehousing technology helps, as do SQL data migration tools, but data integration is still far too painful today. Phil cited distributed application management as another area in need of more attention. Developing a transaction processing application today is a mess; it involves programs in various programming languages, web browsers, TP monitors, and database systems. As a result, there are numerous pieces to today's solutions, with complex inter-dependencies, making it hard to keep track of everything.

Lastly, Phil discussed an opportunity, in the area of system engineering. He pointed out that most IT people engineer systems, not components. Today, there is little science, and few technically deep tools exist (e.g., Microsoft Word templates are the #1 tool of a leading system engineering consulting firm!). Engineering large I/T systems for a particular degree of reliability, security, or manageability is beyond the state of the art. He asked why we don't build software systems like Boeing builds airplanes, or like Intel builds chips—putting forth the idea of repository-based system engineering, based on a "metadata vision." Phil gave an example scenario where metadata is key—and where better metadata management would save lots of time and money—namely, managing a data warehouse. He pointed out that we really need tools for back-tracing through data transformations, identifying changes, etc., for example to figure out why something in a data warehouse that used to make sense no longer does. Today, there is almost no tool support for this kind of analysis, and it can take weeks to perform.

Paula Hawthorn (Independent Consultant)

The next speaker was Paula Hawthorn, previously of Informix and Andromedia, whose topic was database software development problems. Paula is currently taking a year off "to figure out why it's so darn hard to get innovations into software products." She made the statement that the database software industry currently has some "inherently unsafe" algorithms and practices, making an analogy with Ralph Nader's *Unsafe at Any Speed* expose of the auto industry. Paula stated that the reality in the database industry is that releases of any given database product tend to be dominated by bug fixes. Database software bugs include misunderstood features, mistakes in asynchronous logic, unacceptable performance degradation, and C-related problems (e.g., stray/uninitialized pointers, memory overruns). As a result, industrial developers tend to be unable to incorporate new research results into their systems, telling their research colleagues one of three things: (1) "We're too busy." (2) "Maybe in a few years." (3) "Come and fix some bugs if you want to work with us."

Proposed solutions to this problem are many, but she said most are inadequate. One approach is to hire smarter programmers, getting better requirements/specifications documents and better designs, and going slower. Another is to use a better language (e.g., not C). Another is to use performance modeling tools, and/or to use system specification & modeling tools. Paula cited several failed approaches, including the use of system-specific languages, reusable modules, and doing things in hardware instead of software. As a result of the current problems, Paula said that we are in a state where all current database systems are inherently not robust! Potentially useful work, which might help with this in the future, includes work on designing systems with self-checking by module, "try-again" transaction logic, built-in performance monitors, and simpler system architectures. Other work that Paula mentioned involved work on better C programming tools, better training for programmers (e.g., in areas such as design for robustness, exception handling, tracing, asynchronous logic, and tool use). She also mentioned the value of formal proofs of correctness when possible. (Paula admitted to being a dreamer here, but cited the work of Cherniak and Zdonik on proving the correctness of query rewrites.) Paula concluded with the statement that database research is not useful unless it includes how to reliably implement the results.

Eric Brewer (Inktomi & UC Berkeley)

Eric Brewer spoke about "real internet services" and why they don't use DBMSs for most of their data management. Eric's background and "home" community is the operating system community; he stated that the DB and OS communities are not well enough connected.

Eric founded Inktomi, a company that builds web search engines (e.g., Hotbot). Inktomi builds two kinds of web-related data management systems: global search engines and distributed web caches. However, Eric stated that "neither uses DBMS technology in any significant way." The underlying hardware basis for Inktomi software is scalable/parallel computing technology (i.e., NOW systems, in Berkeley parlance). Eric explained that Inktomi provides both software and hardware infrastructure for web indexing and searching for many different front-end companies (e.g., Lycos, Yahoo, and others). They have a 166-node cluster in Santa Clara, plus they also have two other 100-node systems. System availability is a critical issue for them.

Eric explained how to think about search engines as managing a database. Typical parameters today are: 110 million documents (think of them as being in one big table), with about 10 million distinct words (which he said to think of as yielding 10 million smaller tables, though some attendees weren't 100% clear why); the data size ends up being about a terabyte of data (mostly text). Typical performance requirements include a 250ms average response time (with contractual obligations to meet this target) and 99.99% availability (also with contractual obligations). Their systems can handle over 25 million queries per day; updates are partially batched, occurring several times a day. Their approach is extensible (for non-text data types). Significant other challenges include the need to handle (a priori) unknown but large system growth, being truly available (no offline time), and being able to constantly evolve the system (in a fast-moving web timeframe). They use a DBMS for managing some data (e.g., advertising info), but most of the Inktomi data management does *not* use a DBMS. This is because they see DBMSs as being too slow to meet the stringent performance requirements, because their application needs availability more than it needs consistency, and degradation must be graceful. By doing their own data management, they are able to optimize for their requirements. For example, they support only very simple atomic updates ("table replacement" = pointer swap) rather than multi-object/multi-statement transactions. Also, in the web search world, it is okay to temporarily lose small random subsets of the index data due to faults (or to take small random subsets offline for updates).

Eric's main point for the database community is that database systems today focus mostly on providing ACID transaction semantics, while Inktomi forfeits the C (consistency) and I (integrity) features of the ACID model in favor of higher availability, graceful degradation, and performance. Eric calls their approach BASE (which stands for *Basically Available Soft-state, Eventual consistency*) to contrast it with ACID. His claim is that getting near-perfect availability is good enough for a lot of customers on the web, and that we need to recognize and exploit this. Eric then proceeded to compare and contrast BASE with ACID semantics. He explained that BASE systems are characterized by weak consistency (stale data is okay), availability first, best effort, approximate answers, aggressive and optimistic approaches, and are thus simpler, faster, and more evolvable. ACID systems take a position at the opposite end of the spectrum. He asked whether or not there is a spectrum between ACID and BASE; he thinks so, but isn't sure. Comments came from the audience at this point. One person commented that data warehouses are probably in the middle somehow. Another comment was that asynchronous replication, as provided in RDBMSs today, is probably another example of a point in between the ACID and BASE extremes.

Eric then presented the CAP Theorem: You can only ever have two of the three features in the set {Consistency, Availability, network Partition-tolerance}. Eric asserted that having all three at once is impossible, but that any two can be realized together. There are many examples of C +

A without P (e.g., today's database systems). C + P examples are distributed database commit protocols. A + P examples include web caching, Coda, and DNS. Eric closed by claiming that ACID versus BASE involves important, real tradeoffs, and that real internet systems are a careful mix of both ACID and BASE; he said that almost no work has been done in this area. OS versus DB community separation is one reason for this problem. (Eric mentioned that OS people find it painful to use DBMSs.) Finally, Eric said that we must admit to ourselves that we can't get to 100% reliability in many cases, and then base (pun intended?) new work on that in order to solve important classes of problems.

Paul Durdik (Townsend & Townsend & Crew)

Paul Durdik, a patent attorney in Palo Alto, then educated workshop attendees about intellectual property law issues. He started by explaining that the major forms of legal protection are patents, copyrights, trademarks, and trade secrets; these forms can be used simultaneously. Paul then explained the major forms in more detail. Patents are for protecting inventions. (An invention, in the eyes of patent law, is a novel, non-obvious composition of matter, article of manufacture, or machine/process.) Patents prevent others from performing certain acts (make, use, sell, import) related to an invention. They are issued for a fixed term (20 years from filing or 17 years from issue). Independent creation is not a defense for patent infringement, and they involve civil liability. Paul explained that the power of patents is that no prior relationship needs to exist between the patent owner and infringer, and copying is not needed to infringe. Independent design is not a defense, nor is ignorance, nor is prior secret work. Paul mentioned that patents are used by venture capitalists to help them assess startup companies, while big companies most commonly use them defensively (i.e., for cross-licensing). To get a patent, one prepares and files a patent application. Patents can cover new functionality or new ways to achieve old functions (better, faster, cheaper, etc.). Infringement is to be avoided, and detection of infringement must be possible to make a patent effective for protection. Patents are commercially valuable. The owner of a patent is by default the inventor, with exceptions due to employment clauses or shop right. In the US, one must file for a patent within one year from the public disclosure, use, or offering for sale of the invention. Elsewhere, there is no grace period; filing must precede such disclosing events. Some notable software patents include: overlapping windows, a cursor that changes shape, pull-down menus, modem escape sequences, and public key encryption.

Copyrights protect original works of authorship. They prevent others from copying the work. They have a limited term (life of the author plus 50 years, or 75 years for joint work). Unlike patents, independent creation is a defense for copyright infringement, but it is obviously highly unlikely. Both civil and criminal liability are involved. With copyrights, protection begins upon creation of the work. No copyright notice is required on a work, but one is needed for statutory damage claims. One can register commercially important works; Paul mentioned that normally the first and last ten pages of source code are registered for programs, which led to some amused comments from the audience. With copyrights, the author/creator of a work owns the intellectual property. Exceptions are work made for hire or work covered by contractor agreements.

Trademarks protect indicia of origin of goods and services. They are for prevention against likelihood of confusion and last for a potentially infinite term. Independent adoption is not a defense against infringement, and both civil and criminal liability are involved. Trademarks are used as a mark on goods or services. Proper use is as an adjective, not a noun. (E.g., this use is right: Quattro Pro software). One must identify the mark with the TM symbol at least once, and register it with the USPO. Rights to a trademark belong to the creator, and the creator must control the use of a trademark (through a licensing program).

Trade secrets are for protecting confidential information having commercial value. No formalities are required; one must just keep the information secret. They have a potentially infinite term, and both civil and criminal liability are involved. Proper handling involves the use of confidentiality provisions in employment agreements, the provision of notice (e.g., on secret documents), and reasonable measures must be taken to protect the secrets. Trade secrets are owned by the creator of the information, and are governed and controlled by employment agreements. Paul closed by saying that databases (i.e., collections of data) present special problems in the area of intellectual property. Patents are not given for ideas or information only, copyrights require originality (Paul cited Feist vs. Rural Telephone), a trademark protects (at best) the use of the mark (e.g., Reuters(tm)), and the concept of a trade secret does not apply to databases. Currently, control of database content is by contract rights (i.e., fee-based access) in the US. In Europe, there is an EU database directive that extends the copyright notion to database contents, adding the right to prevent unfair extraction.

John Cherniavsky (NSF)

John Cherniavsky spoke on university/industry cooperation models. John first cited several NSF cooperation models that work, including Engineering Research Centers, Industry/University Research Cooperative Centers, Science and Technology Centers (to some extent), GOALI, and SBIR. NSF Engineering Research Centers get their funding from industry (for the majority of the funds), the university site, and NSF. They must have at least six industrial participants. SRI did a study of the benefits of these centers. Respondents said that their most significant benefits to industry include access to new ideas, interactions with other ERC representatives, access to equipment, and (cited as the most important benefit) access to students; other lesser benefits were also cited. Respondents said that the most significant benefit to universities is the impact of the center on students' educations. The I/UCRC program involves less money from NSF, with the focus being on funded partnerships. Intellectual property is to be shared among the center members. There are few computer science instances of this type of center to date; there is also a similar program (S/I/UCRC) involving states. The GOALI (Grant Opportunities for Academic Liason with Industry) program covers extended faculty visits to industry, joint projects with industry, graduate student support for visits to industry, and post-doc support for 1-2 years for visiting industry. The SBIR (Small Business Innovative Research) program is a congressionally mandated program that gets 2% of NSF's overall research funds. SBIR grants have three phases: a $100K feasibility phase, a $400K refinement phase, and then a $0K commercialization phase (where no NSF support is provided). The SBIR program is being reformulated as to areas of interest; it is too small and too specialized to be a good industry/university cooperation model. NSF Science and Technology Centers are probably not great models for this type of collaboration. They are large centers ($2M-$5M/year for multiple years). An example is the DIMACS (Discrete Mathematics and Computer Science) center at Rutgers, which involves participation from Bellcore, AT&T Labs, NEC Research, and Lucent Technologies. The industrial value added is access to students and a "place for science" within a company; it is not entirely clear what the added value is for the university. After describing the above models, John listed several cooperation models that don't work well for university/industry cooperation. Defense transition grants are too applied, while ATP and other technology transition grants are too short-term for universities. John mentioned that it is very important to avoid inappropriate roles (i.e., universities and industry can't do each others' jobs) and inappropriate values (e.g., where industry sends "deadwood" to universities, or university faculty members consult only for the money). John closed by discussing cooperation barriers, including intellectual property issues (e.g., university policies, lack of source code access), lack of time for interaction (e.g., project deadlines in industry, lack of rewards in universities), and privacy and security issues.

Mike Ubell (Informix)

Mike Ubell described what it was like to take Postgres, a system developed at UC Berkeley, and turn it into a commercial product, the Illustra object-relational DBMS. Illustra was later purchased by Informix, but Mike focused on the Postgres to Illustra experience in his talk. Postgres had a few hundred users, and the original code was written in Lisp. The system had a number of novel features, including classes, a no-overwrite storage manager, and extensible interfaces (which Mike summarized as "read the code, call what you like"). It also had a number of simplifications, compared to industrial products, including a process-per-user architecture and table-level locking. The bottom line is that it was a research system. Mike then proceeded to discuss the transformation of Postgres into a commercial product, and issues that arose along the way. The first issue discussed was stability. The starting point for Postgres was that the system shut down if any process faulted. Mike talked about various scenarios: the probability of a process doing damage (in shared memory) and then faulting, or doing damage and then not faulting, or not doing damage and faulting, and so on. He asked if you cause a problem in shared memory, will it make it to disk? Another starting point for Postgres was that the system's client/server protocol was not robust. Simply adding auto-reconnect support helped with this. They also added exception handling (including graceful backout and meaningful messages). Another problem was that the system's spin lock code was not good; this is due to the fact that concurrent programming is still very hard (more an art than a science). At the start, Postgres had no backup/restore utilities; this was added via logging (to the Illustra "diary") and a dump/restore (to tape) utility. Other cleanup work related to stability was the addition of consistency checking utilities to the system's index manager code (B-trees/R-trees) and the elimination of various obscure, overly-layered coding conventions (which resulted partly from the system's being built as a set of graduate student projects). The next issue discussed was testing. Mike emphasized the importance of hiring QA (quality assurance) team members early, rather than near the end of the development process. He also talked about the importance of using code coverage utilities (saying that there are number of good ones available). Illustra employed nightly builds and Purify usage, building the system on multiple platforms. The model at Illustra was one tester for every two system developers. (Phil Bernstein stated that the Microsoft model is closer to one-to-one.)

The next issue that Mike discussed was usability. Postgres was initially based on an extension of the Quel language (Postquel). Illustra chose to convert to SQL, both for obvious commercial reasons and because SQL3 was then under development. The Illustra work helped debug the standard. ThePostquel-to-SQL transition was done very quickly (one month to demo, though the internals were still messy then). In terms of the client API, Postgres had a new interface called portals. Illustra moved to an ODBC-like interface (which in retrospect should have just been ODBC, Mike said). They used the same API for both server and client functions, and again documented the system's internal API in order to support extensibility. Mike then turned to the topic of performance. Converting Postgres to Illustra involved implementing page-level locking and deadlock detection (where they started with a fixed timeout, then went to an adjustable one, then added a partial deadlock check). The optimizer was extended to employ some heuristics to handle expensive functions (based on university research) and to provide an API for user provision of selectivity information on user-defined types. System catalog caching and indexing was improved. Memory management in the system was initially very expensive, complex, and leaky; it was cleaned up, and memory durations were added for reclamation. To make queries faster, evaluation of SARGs (Selinger et. al., SIGMOD-79) was pushed into heap access method. Postgres did not initially reuse empty space in tables at all; this was added for Illustra. Postgres' implementation of large objects had problems related to file name space/lookup that were fixed in the Illustra conversion. Support for pre-allocated process pools was added for performance, with processes being allowed to die off occasionally as a way of handling storage leaks and other creeping bugs. Mike closed by emphasizing the importance of having good marketing people involved with a product.

Todd Walters (Teradata/NCR)

The last regular industrial talk of the day was given by Todd Walters, on the topic of "Query Complexity." His alternate talk title was: "You can do all these wonderfully complex queries .. but ... how do you know that they are right???" Todd started by asking how one can define what's meant by a "complex" query. The answer is schema dependent, database size dependent, product dependent, user/DBA dependent, and time dependent (as products mature). He gave the following time-dependent definition of complex queries:

- 1986: several aggregates and a complex where clause
- 1990: star and snowflake joins
- 1994: add outer joins in combination with these
- 1996: lots of grouping, nesting of complexity
- 1997: huge queries, all over highly complex views (e.g., phone company rate structure "what if" query)
- 1998: INSERT SELECT <48 columns> FROM query expression with 54 derived tables, 21 left outer joins, 13 unions, 10-deep nesting of cases, 31 tables referenced, most via views! (Todd found out about this last query when a customer mailed it to him along with a question: "The optimizer said this query should take 4 minutes, but it took 6. Why?")

This trend toward increasing query complexity is the result of growing user sophistication (analysis) as well as the use of tools that generate queries (as opposed to hand-written queries). Query complexity, according to Todd, has a number of aspects: combinations of tables, usage of nesting, interactions between subqueries, expression complexity, number of joins, and number of unique tables.

Increasing query complexity has a number of serious impacts. One is query correctness: how does a user know if he or she has asked the right query for given business question? That is, the ability to express, understand, and debug queries is an increasingly important issue. Another is the ability to come up with the best plan to execute a query, and also to comprehend the plan. Another issue is answer correctness: how does one know if the answer to a highly complex query is actually right? Our ability to evaluate answers in this regard is decreasing, both for users and tool vendors. The bottom line is a potential confidence crisis; query complexity is a source of significant costs to both customers and to database system vendors. Below the surface of a database system, complexity raises issues as well, Todd explained. Complex queries are not run standalone; a Teradata system can be expected to run 50 or more concurrent queries. Interaction between queries is common—they share the disk cache, may have common scans/joins, common aggregations, and share physical resources such as disk arms, the system interconnect, the processors, and main memory. As if all this were not enough, DBMSs themselves are getting much more complex, adding objects, user-defined functions, and user-provided Java code, all of which interact with complex queries!

Given this background, Todd said that we are now facing a number of important complexity-induced challenges. First, how do we eliminate regressions (i.e., the introduction of bugs as we change DBMS code to cope with increasing complexity). Second, Todd said that query optimization is not a solved problem, because we have no idea how to find the best query plans for extremely complex queries. Third, how do we avoid having complexity costs eat us up? Finally, how can we ensure correctness and give database system customers and users the confidence they need to move forward? Todd closed by saying that the query complexity crisis means that we need entirely new testing methods and tools, further query planning research that addresses complexity, combinations, and testing methods, and query comprehension tools.

Asilomar Report

At this point, Phil Bernstein summarized the recent Workshop on Database Research held at Asilomar, California. Its goal was to re-think the database research agenda. The Asilomar report appeared in SIGMOD Record and can be found at: http://www.acm.org/sigmod/record/issues/9812/asilomar.html.

Panel Session: Overcoming Barriers

The first day concluded with a panel on university/industry cooperation barriers, issues, and possible solutions. Each panelist gave a short presentation on this topic.

Betty Salzberg (Northeastern) argued that academics need more (good) information. She cited several examples of work based on faulty information, including a 1985 TODS paper with incorrect WORM disk assumptions, a number of optimistic concurrency control papers that ignored transaction undo costs, and an ICDE-99 submission that assumed it would be reasonable to have memory-resident information about every single page of a large database. Betty suggested several mechanisms that academics might use to try and obtain better information. One approach is to submit papers and hope for useful industry feedback. Another approach is to have faculty members spend time in industry. (Brad Adelberg commented that this is not a scalable solution.) Yet another approach is to take a class from someone who knows, being careful to distinguish facts from propaganda. Other approaches are to use the web to find product manuals and white papers online, to read proceedings critically (but Betty commented that most people can't do this), and to read textbooks (but Betty commented that many textbooks repeat old and sometimes faulty information, and don't always provide judgements). Thus, we need to work on better flow of good information from industry to academics.

Mike Carey (IBM Almaden) first described barriers to university/industry cooperation. One issue is that you can't simply reinvent the world from the ground up; DB2 is over a million lines of code, and the world is full of SQL programmers/tools. (Another pitfall is failing to reinvent the world enough, such as deciding to "screen scrape" and query HTML when the underlying data is database-resident.) Another impediment is thinking of the world only "in the small" when doing research -- e.g., adding new data types also impacts database system utilities, client APIs, optimizer costs and statistics, etc. Finally, impact is more than VLDB/SIGMOD acceptances, and what plays well in industry is actual prototyping. Mike then talked about what industry wants from universities. The top priority is that "industry wants your students," especially those with a strong systems sense and skills. (Todd Walters commented that those with "asynchronous sense" were especially desired.) In addition, industry wants practical solutions to real problems. IBM Almaden is interested in OR/OO database technology, business intelligence, and web commerce, among other things. Mike listed zero-administration database systems, automated function and system testing techniques (that apply when new features are added to a system), and utilities and tools as under-studied areas. He suggested partnering through visits as one mechanism to make cooperation work.

Paula Hawthorn (Independent Consultant) asked what cooperation means. She said that to some universities, it means "give me your money." But industry must consider the return on investment (ROI) of such funding and how it compares to ROI on other possible uses of that money. She also asked about the proper role of universities, emphasizing that universities should not lose sight of their long-term role. Given this, she asked if it was appropriate for universities to be doing a lot of systems-oriented research. (Some participants argued that systems-oriented research is essential, because a steady stream of good developers is industry's top priority from universities.)

As an example, Paula cited her efforts at HP Labs to obtain funding for research at Stanford. She pointed out that it's often a zero-sum game, with return-on-investment considerations, for companies. If we want universities to work on further-out things, how do we get industry money flowing to academia? (Ashish Gupta and Praveen Seshadri suggested industrial visits to academia might help.) Paula also talked about the problem of getting industrial knowledge disseminated to the community at large. She mentioned that Jim Gray did this back when he was with IBM Research (e.g., for Ron Obermark's work). She said this isn't happening today because people are all very busy, and that there is no longer any such thing in industry as "your own time". She said that the good news is that researchers in academia are very interested in learning about industry trends. She proposed giving industrial folks sabbatical time at universities periodically (i.e., this could be a company policy for top employees).

Jeff Ullman (Stanford) said that there used to be a good model of collaboration, back in the early days of computer science research. Graduate students, faculty, and industrial researchers were able to talk and work together on basic issues, and everyone benefited from this. Intellectual property was not a barrier (for some reason). With respect to theoretical work without immediately obvious practical uses, Jeff reminded the attendees that LR-k parsing was impractical when it was first proposed by Knuth, but obviously turned out to be very important and practical (after subsequent developments). He pointed out that we must not forget this sort of example. Jeff then talked about barriers. One barrier is the university research model. He said that after WW-II, the federal government turned teaching colleges into "research universities," making them charity-dependent. He said that "research causes wealth" is wrong; "wealth causes research" is how things actually work. This leads to a mismatch. Jeff said that universities haven't changed their business model, and are not inclined to do so. Another barrier is the need for companies to "protect their crown jewels." He said that, in other fields, industrial involvement "works." An example is Stanford's Center for Integrated Systems (which does work on E-CAD systems, packaging, etc.). He believes that this works well, because the participating companies are happy to license something good in an area like design tools—it's cheap to do so and doesn't interfere with their core business. However, Intel obviously couldn't risk having to license the Pentium III design from a university. By analogy, then, it would be great if IBM, Oracle, etc., could all get together and do query optimization research at Stanford, but they'd fear having to license it. Thus, a big problem arises when the topic of cooperative research relates to a company's crown jewels.

Maria Zemankova (NSF) discussed collaboration barriers and how we might remove them. First, it can be difficult for academics to find suitable research problems. To address this, she suggested a web site where industrial people could post significant unsolved problems. In addition, panels at conferences can be helpful. Second, it can be hard to find suitable collaborators. She suggested that there be a place for academics to post their current research problems and also that a "matchmaking" mechanism be established. A third barrier is the ever-present issue of intellectual property. She suggested that government-funded work require royalty-free licenses for products of the research. Also, academic institutions need a better understanding of financial issues (e.g., as summarized nicely in Randy Katz's CRA report), while industry needs a better understanding of academics' open publishing needs. Another difficulty is funding of visits from academia to industry or vice versa, e.g., industry visits by faculty or students; she wonders if companies could support such visits as a form of PK (formerly called PYI) program matching. Still another barrier is the lack of access by university researchers to database system code; she wonders if university researchers could create an open source database system, using an analogy such as Linux. Another barrier is lack of access to realistic data; she wonders if we could make sanitized commercial data available, and/or set up a repository for data/benchmarks. Other barriers include lack of access to realistic platforms and "loss" of graduate students and faculty to industry.

Discussion: Phil Bernstein suggested that an effective mechanism for informing researchers about research problems of interest to industry is "industry perspectives" papers in *SIGMOD Record.* Joe Hellerstein suggested having *SIGMOD Record* publish papers based on academics interviewing people in industry. This addresses two problems: (1) industry people are often too busy to write about their problems and (2) they are not always able to describe their problems in terms that are useful to researchers. Academics (or their students, with guidance) could use interviews to solicit research problems of practical importance. The bottom line from workshop day #1 is that there is a clear need to improve existing, and provide new, mechanisms for university/industry communication and knowledge sharing.

Anil Nori (Oracle)

Anil Nori opened the second day with a presentation on databases for the Internet. (Interestingly, since that time, Anil has left Oracle to start a company in that area.) Anil's presentation was based in part on the comments of a number of customers (e.g., Amazon, Dell, and Cisco) who are running web sites supported by an Oracle DBMS server.

Web applications are typically constructed using a three-tier architecture. At the bottom tier is an object-relational DBMS, along with gateways to other data sources (including OLE/DB). On top of that sits a thick (physical) middle tier that does storage management, query management, etc., and serving as a web application server. At the top of the architecture are tools for web site construction. Despite the fact that many middle tier functions are routinely handled by database servers, databases aren't being used for the middle tier. Instead, most customers are using database systems as "dumb stores."

Anil talked about the characteristics of internet applications. The corporate intranet is important, in addition to the internet. Most applications are middle-tier oriented; the middle tier is really the server to users of internet applications. Most internet applications are distributed and transactional, and they tend to involve complex data types and complex logic. Often they require integration of both data and applications, and include requirements for query and persistence. 24x7 availability is also a common requirement, as is security.

Anil then turned to the question of why databases aren't playing a more prominent role in web applications. He said that data, applications, and transactions all tend to be quite complex, which is a challenge, in internet applications. Availability and scalability are critical, and many users must be supported with appropriate security. He stated that our goal should be to make the DBMS the preferred platform for internet applications. Anil stated that Cisco today has a strong I/T department that rolls their own web application software and solutions, and that they do this because they have to—because existing systems and packages are not meeting their needs—not because they want to. Anil's message was that we should be evolving OR-DBMS systems in a direction that will fix this situation.

Next, Anil talked about the kinds of data that enterprises like Cisco need to deal with. First, he cited multimedia data as a requirement; they need to store, manage, and present data types such as text, audio, video, image, spatial, and timeseries, plus web content multimedia formats such as AVI and Quicktime. Second, he mentioned their need to deal with various forms of "desktop" data, including PowerPoint, Excel, MS Project, and e-mail files; these are typically living on desktop machines and involve compound types. Third, he mentioned what he called "vertical" data: complex data whose type depends on the vertical market of the enterprise. Examples include chemical informatics data, bioinformatics data, asset management, and special purpose proprietary data systems. James Hamilton commented that people would probably use document attributes in products like Microsoft Word if database systems supported queries on them (i.e., if using them added value when looking for them again). Anil Nori commented that most people have simple requirements for text search; they would prefer the speed of a web search engine to the accuracy of something like Oracle's Context engine.

Anil listed a number of different categories of internet applications: web integration, application integration, electronic commerce, web publishing, and vertical applications. He then talked more specifically about each of these.

Web integration involves heterogeneous data sources and data types, with much of the data being dynamic. Their needs include support for rich types, multimedia types, self-describing data, HTML/XML support, and complex data querying (i.e., content-based queries).

A long discussion occurred at this point related to XML and databases—i.e., what can/should our community be doing? A distinction was made between two different cases: unstructured or semistructured data handling versus the need to manage structured (e-commerce) web data. Hamid Pirahesh argued that schemas and data types are important, and that a large fraction of the money to be made on the web will likely be in the e-business arena (where numeric and structured data plays a big role). This led to a discussion of requirements for semistructured data. Peter Buneman argued that some important applications (e.g., bioinformatics) have a need for simple structures plus ancillary structures for annotations. Others indicated that most semistructured data requirements could be met with extremely wide tables populated mostly with Nulls. Phil Bernstein then pointed out that a metadata binding problem arises with XML: for a given piece of data, or collection of data, what schema does it conform to? This sort of information will be important for e-business applications.

Application integration is workflow/message-based, and it involves self-describing data (for messages), heterogeneous data and message formats (e.g., SAP, EDI, Oracle applications, etc.), and business transactions. The requirements for this application category include a messaging framework, support for rich types, message standardization/conversion, XML support, complex query support (e.g., for use in doing associative dequeueing of messages), and support for cross-system transactions, particularly across Enterprise Resource Planning (ERP) systems such as SAP.

At this point, as an aside, Anil mentioned that a number of startup companies are springing up to do "web hosting"—i.e., to provide, support, and manage internet applications for companies that can't do it for themselves.

E-commerce was the next application category discussed. Anil characterized e-commerce as involving and/or needing support for self-describing data, a commerce object model (which can handle heterogeneous formats), efficient messaging over HTTP, business transactions, and content management. Specific needs for this category include secure HTTP, an HTTP-enabled DBMS (so that web servers can talk directly to the DBMS), message standardization/conversion, and XML. This category lead to a discussion of whether or not we can build 2-tier (as opposed to 3-tier) solutions by making the DBMS also be a Java-based web server. Anil stated that Oracle is doing exactly this. A big debate regarding 2- vs. 3-tier architectures ensued at this point.

Web publishing involves content creation and management, personalization (to capture user preferences), and information content extraction. As a result, its requirements include support for the storage of content and preferences, separation of content vs. structure vs. presentation, efficient data exchange and storage, and (obviously) XML/XSL support.

Because of the level of interest in Anil's presentation, time was running short at this point. He skipped the rest of his slides on application categories and their requirements and proceeded to list important research problems that address the emerging needs of internet applications. The list included the following problems:

- General needs: Internet applications need near 100% availability in an easy to manage system; scalability is critical, to handle a web-sized population, and we must address security in the face of millions of (global) users, including encryption support; performance must be acceptable.
- Business/workflow transactions: We need transactions that span multiple DBMS and ERP systems; tools for creating compensating actions are needed, as is support for transformations.
- Advanced queueing support: Systems must be capable of handling heterogeneous messages, with the option to handle messages in a transactional manner; for flexibility, support for querying (based on attribute/value pairs) is needed, as is support for indexing (again, based on attribute/value pairs); support for publish/subscribe models is also very important here.
- Rule engines: The database must handle complex business rules, including customization and profiling (for business domains, and including presentation).
- Repositories: Once Java classes are in the database (e.g., to implement methods for SQL3 types), then database systems must be able to manage Java objects and their interfaces, handle information about application integration, support standard object models, etc. Repositories need to address these issues (see Phil Bernstein's presentation from Day 1).
- XML support: The database community needs to look at XML support, including support for schemas, XML object storage, querying of XML objects, and XML/SQL co-existence. Anil noted that current XML query efforts seem disjoint with SQL.
- Multiple caches: 3-tier architectures bring consistency issues, related to middle-tier cache vs. database consistency, that the database community should address.
- Data mining: More work is needed on integrating data mining algorithms into database servers.
- Web tradeoffs: Basic "game change" tradeoffs brought about by the characteristics of web applications include relevance ranking vs. exact answer models for query results and speed vs. accuracy tradeoffs (see Eric Brewer's talk).

Marie-Anne Neimat (TimesTen)

Marie-Anne Neimat spoke on main memory data management (subtitled "a technology whose time has come"). She started out by summarizing the general architecture for multi-tiered computing today. It's common today to have systems with clients at the top, an application server in the middle—running the enterprise business logic, doing load balancing, handling workflow, and/or performing transaction monitoring duties—and a DBMS server at the bottom, providing storage. She then pointed out that DBMS servers are providing more and more features (in dimensions including objects, parallelism, warehousing, and OLAP, ...) and that it is becoming hard to differentiate competitors; she stated that the database server market is saturated, with flat/limited growth today as a result. Yet, most this is where the database community spends most of its energy today.

Marie-Anne then turned her attention to the application tier, mentioning application types including ERP, supply chain management (SCM), and web applications. She pointed out that there has been a recent flurry of activity and startups in the application tier space. Marie-Anne stated that they face difficult data management and access problems, with mostly ad hoc solutions today. Many of the problems that they face, which drive them to use 3-tier architectures, are due to the need to keep the number of round trips to the database server low, the desire to offload the database server, the desire to map business objects (and methods for business objects) to the relational model, and issues related to load balancing for multiple application servers. (Of course, there are some interesting cache consistency issues caused by such architectures.)

Marie-Anne proceeded to describe the chronology of offered solutions to these problems. OODB systems are one solution that has been tried. OODB systems move objects between the (OO)DBMS and the application tier, storing the objects in memory on the application site, giving the application the data and programming model that it wants. (Though as some pointed out, once a second application is written against this data, the DBMS representation of objects may no longer be identical to that required by the application.) These systems were intended as a replacement for relational RBMS servers, with their own object server (written from scratch); this was their mistake (and their downfall).

The next class of solution is the object-to-relational mappers. These systems map C++/Java objects into SQL data and vice versa. Persistence Software offers a "live object cache," which is a shared middle tier cache that supports private updates. This class of solution supports programmatic navigation over cached objects through the supported OO programming language (C++/Java), with declarative access to the data via the RDBMS server (through SQL against the mapped SQL data). Ardent is another product in this arena; they generate relational schemas from C++ classes. They provide an object cache, ODMG support, and use an ODBC-like layer to access different backend RDBMSs. RogueWave (producer of DBTools.h++) is another player in this category. They also do C++ to relational mapping, but do not support caching.

A different class of offered solution, also addressing the middle tier, is the class of software known as application servers. These systems provide functionality such as load balancing, data caching, mapping of relational data to a particular OO model, a common API for supporting multiple DBMS vendors, replication/distribution of data across multiple application servers, and transaction monitoring. Current commercial players include: NetDynamics (Sun), Kiva (Netscape), and WebLogic (BEA).

Finally, there are various special-purpose solutions running on the middle tier. An important example is SAP (see the SIGMOD '97 case study paper by Kossmann et al). SAP employs an internal (middle tier) SQL cache, with a more recent "LiveCache" being available through Software AG. Most other ERP/SCM vendors take a similar approach. As another example, PointCast achieves scalability through pre-caching of data on application servers (see their SIGMOD '98 industrial session paper). At this point in the presentation, James Hamilton commented that it is important to improve communication in our work, because 50% of SAP-type system cost can be due to network interrupt processing. A discussion ensued about client caching vs. server execution of logic, client/server system splits (division of labor), etc. The point was made that database application builders tend not to be very database-sophisticated, and therefore tend not to use our features all that well. (Again, see Kossman's SIGMOD '97 study.) However, the point was made that this situation should improve over time as competition pushes SAP and others to actually use more modern DBMS features.

Lastly, in terms of existing offerings, Marie-Anne pointed out that Oracle 8 includes a client-side cache. Oracle 8 caches are private, not shared; they support queries as well as navigation. It is up to Oracle 8 applications to maintain consistency (by flagging changes and making flush/refresh calls), as the system does not do this automatically. She also mentioned Microsoft's IMDB (In-Memory DB), which is part of COM+ in NT5.0, serves as a distributed application scratchpad that includes transaction support.

Marie-Anne ended her presentation by identifying the common thread(s) among these seemingly different solutions. They all tend to do main-memory data mgmt in the application server to improve performance (by providing fewer round trips, reduced server load, etc.). Their API/data models are not standardized; one finds a variety of APIs including SQL, proprietary ones, C++, Java, OLE, and so on. The division of labor varies somewhat in terms of navigation vs. queries, whether queries can run over the cache, and so on. In all cases, cache consistency is a big problem, raising questions about who should do it (the application, or the system, automatically) and how to handle consistency (e.g., in terms of the model).

James Hamilton (Microsoft)

James Hamilton spoke next, asking the question: "Are the real tough problems interesting?" He emphasized what he calls "blue collar problems"—i.e., tough problems for which people need solutions today, including both customer and DBMS vendor problems. As an aside, James mentioned that he believes (based on having spent a year in Microsoft's NT group) that it is important for us to listen to the OS community and move away from the "best OS is a dead OS" model (i.e., he believes that DBMSs need to move up in the software stack). As a precursor to his presentation of problems, James made the distinction between fundamental vs. incremental advancements, where the former give orders of magnitude improvements while the latter deliver some factor of improvement. He stated that fundamental advances will always transfer from research into products, but they are very hard to come by (and are thus relatively infrequent). Incremental advances are harder to get into systems, since they offer less benefit, especially if they don't quite fit the system's current model (due to cost/benefit tradeoffs involved in making the needed changes).

James first discussed customer problems. He explained that administrative costs and complexity are becoming the #1 problem (not performance!) for I/T shops. DB vendors today have 100's of parameters, and setting them properly is a big challenge. Another major customer problem is unpredictable scaling due to the brave new world of the web. He gave Charles Schwab as an example, which expected to have 50,000 users of an e-trading system after one year; the actual number of users was 500,000 after only nine months. The internet makes scaling very difficult to predict; meanwhile, most DBMS vendors (and customers) focus more on performance than scaling today. (Paul Hawthorn commented that a critical related issue is handling huge spikes in demand gracefully.) The next customer problem mentioned by James was availability/reliability. For example, he suggested that perhaps the DBMS community could learn something from the Inktomi approach to availability (see Eric Brewer's talk from day #1) to save on the code cost/complexity associated with on-line utilities. James mentioned that query systems are joining TP systems in becoming considered mission-critical; 24x7 operation is increasingly critical, and outages are hugely expensive. Another customer problem is legacy systems and "vendor lock-in"; proprietary stored procedure languages, and SQL DML/DDL differences between vendors, make supporting multiple DBMSs difficult. The last customer problem that James discussed is the need to run queries over all corporate data regardless of its resident data store. Enterprises today have islands of data, where integration is needed, involving all sorts of data. Rick Snodgrass commented that it's hard for academics to tackle (or to want to tackle) multi-component systems problems; a big discussion followed this comment.

Next, James turned to vendor problems. The first problem was database software complexity and size. NT 5.0 contains over 50 million lines of code (MLOC), and typical DBMSs contain over 1.5 MLOC; SAP contains over 37 MLOC. James said that successful software has broad appeal, which typically requires many features. All software appears to either grow or die. How to control growth and manage complexity are therefore major issues for database vendors. A related problem is disk and memory footprint control; release-to-release footprint growth is a huge vendor problem. Another problem for vendors is single code-base scaling. Supporting multiple code bases drives up costs and slows down innovation; James said that DB2 UDB (one code base from Win95 up to large MPP systems) is a good approach. Another vendor problem that James cited is the need to support automatic system administration, given the presence of many tuning parameters and rising administration costs. It's too hard to install and manage the vast majority of today's database systems. Another critical vendor problem is software testing. Testing resource requirements are the single leading reason why function doesn't get shipped in commercial products. Factors making this hard include code base size, features supported, and configurations supported; there is no single typical customer usage pattern. As an interesting approach, James cited the grammar-based query generation tool of Slutz at Microsoft BARC; he also cited techniques that replay actual (multi-user) customer traces at development shop, instrument-driven failures, and tools for test coverage analysis. Knowing when a DBMS is ready to be shipped is difficult, and much more innovation is needed in this area. The final vendor problem discussed was evolution—old brittle code is tough to change. He mentioned that Peter Spiro's approach at Microsoft is to rewrite 25% of SQL Server per release. James also mentioned that "much faster 95% of the time" is not good enough for serious customers (citing Don Haderle). Serious customers running production database-based applications are very sensitive to performance stability, and as a result, systems often get switches to disable new things, further exacerbating the "too many knobs" problem. Code-wise, James mentioned that internal documentation is sorely lacking for most database systems, and that exploding system complexity makes it increasingly hard for concurrent developers to work on extending database system products.

James concluded by listing a number of potentially interesting and useful database research topics, including:

- Auto-administration and fully adaptive database systems
- Disk and memory footprint control (modular/extensible DBMS architectures)
- Support for disconnected clients, replication, and multi-tiered caching. (As an example, he mentioned Lotus Notes.)
- Cost-effective scaling (because we're a long way from linear pricing)
- DBMS exploitation of new hardware/software (e.g., Patterson's IDISK work)
- Software testing
- Data, application, and query integration.

Finally, James mentioned some namespace integration work he did as an experiment while in the NT group at Microsoft. Information on this is included in the slides available from the workshop website.

Hamid Pirahesh (IBM)

The final industrial presenter was Hamid Pirahesh from the IBM Almaden Research center, who talked about his experiences in taking Starburst from a research prototype to being a significant part of the DB2 product family (now known as DB2 Universal Database).

Hamid starting by summarizing the chronology Starburst. Initially, Starburst was an approximately 6-year research project. Technology transfer began when a decision was made to renovate DB2/2 by improving its data manager and replacing the its query compiler (the top half of the DB2/2system) with the Starburst query compiler. This was a major piece of work, as DB2 today is scalable/parallelizable (running on serial, SMP, and MPP platforms with hundreds of nodes); the transfer process took about 3 years. The renovation of DB2/2 took place in phases. First, work was done for serial platforms, then parallelism was addressed, and then advanced "business intelligence" features were added. Before the transfer happened, developers ran benchmarks on Starburst to assess the risks and benefits of proceeding. Releases (so far) occurred in '95, '97, and '98, with a demand for a higher product delivery rate in the future. Hamid stated that his challenge is to focus near-in while keeping a 5-year (or more) timeframe in mind. The initial goals for the Starburst transfer were extensibility and support for complex queries. Today's R&D/product goals for DB2 are for new object-relational and business intelligence features (i.e., even more complex queries).

Hamid also discussed the economics of this work. He explained that a DBMS is typically more than 1.5 MLOC (not including ancillary tools), and that the query compiler is a big chunk (25 to 40% of the system). He stated that transferring technology into a DBMS product is expensive if the impact is fundamental (e.g., optimization, parallelism), so you have to be in the game for awhile to make it feasible/worthwhile to do. The Starburst query compiler grew by a factor of two in transitioning from a very solid prototype into an industrial strength product. The size doubling was due to system interactions and fixing shortcuts that one can take in prototypes but not in products. Hamid also mentioned that the coding rate for small research prototypes is one half to an order of magnitude faster than industrial coding rates.

Hamid then described several concrete examples where research has had a significant impact on the DB2 product. His first example was magic decorrelation techniques and their use for "traditional" SQL query processing. Hamid explained that correlated subqueries are hard to process well, especially for MPP systems (where it is relatively easy to end up running N**2 threads on an N-node system). IBM attacked this problem by working with Stanford and Wisconsin database researchers to apply magic optimization techniques here. Hamid noted that the database research community did lots of complex work in this area based on two-level recursive query examples (e.g., "same generation"), as opposed to looking at more practical uses. The addition of magic to DB2 took about four years. First, there was an initial prototype written that used supplementary magic plus magic (using sideways information passing); it was shown that magic techniques can be used very effectively for "real" SQL queries (nested queries without recursion). Magic sets can be used to turn tuple-at-a-time processing into set-oriented processing for these sorts of queries, and in DB2 magic is implemented using query rewrite optimization rules. Correctness was shown by analyzing rules as individual components (where each rule is a little C program).

Hamid emphasized some lessons learned from this experience. First, it takes a while to grow a good idea. You have to bet on it and invest in it (but be selective), and it's good to involve graduate students (as was the case for magic at IBM). This is a good case study of a successful university/industry collaboration. Praveen Seshadri pointed out that Datalog doesn't have correlation, and so, by themselves, academic researchers would not have discovered this important use of their technology. Only by bringing academic researchers together with industrial ones could this important result have been achieved (because the problem just didn't exist in "Datalog-land"). Second, Hamid said to go after the "big wins." In this case, instead of going for say a 20% performance improvement, previously "impossible" queries became possible. Finally, Hamid emphasized that stability is more important than extracting every last bit of performance.

In a side discussion that followed, Ken Ross mentioned that he also had success with transferring some correlated subquery work to a vendor: Sybase. Work that he did with his student Jun Rao ended up both in Sybase's product code base and in a SIGMOD paper. Ken explained that this happened when he met someone from Sybase at a conference, visited there and gave talks several times, and then sent Jun Rao as a summer student, whose work at Sybase lead to the successful transfer. Interestingly, intellectual properly wasn't an issue; open publication was okay with Sybase.

As another example of a research problem with serious product impact, Hamid talked next about cyclic referential integrity (RI) constraints. Hamid explained that the DBMS must parallelize RI when you have large amounts of data (e.g., terabyte databases). RI can have cycles, e.g., due to employee/department relationships (like works-in and manages). Eliminating orphan children is a recursive problem, and the recursion involves deletes and side effects. Cycles can be of arbitrary length, and RI cycles may even overlap. DB2 UDB supports full SQL RI on MPP, DB2 adds support for general constraint checking and triggers with RI, and it all works. How was this problem solved? Hamid explained that a general mechanism was developed for recursive query processing in DB2, and it was then used for RI as well. This work involved a collaboration with Stanford on recursive query processing as applied to SQL. The work relied heavily on earlier DB research on fixed point semantics, stratification, and linear (not non-linear!) recursion. Hamid said he kept skeptics at bay by pointing out that recursion was useful for a real problem: bill-of-material processing (Rosenthal, SIGMOD '86). In fact, DB2 achieved a 100-fold speedup on a customer bill-of-material problem through this work. Then, when DB2's need for handling and parallelizing cyclic RI constraints arose, this technology was there and ready.

As a side note, Hamid mentioned that he has found it difficult to get papers accepted on this work. (Papers were rejected initially in both cases.) He stated that program committees often don't recognize the value of systems work of this sort, based on applying and extending theoretical work to the point of real system usefulness. They tend to react with "the research has already been done," rejecting systems papers that apply and extend it in practice.

Hamid said that recursive queries are coming back (e.g., in SQL3). He explained that they are useful for drilldown analyses—e.g., of direct/indirect reports for a division manager, which is an OLAP query over an organizational hierarchy. Another example given was cause and effect analysis for railroad schedules. Hamid summarized this part of his talk by noting that cyclic schemas are common, and recursion is very important. Unfortunately, the early DB theory research lacked a compelling example with practical importance, thus leading to an "image problem" for recursive query processing in industry (and eventually even with conference program committees).

Next, Hamid turned his focus to a new problem: business intelligence. He explained how in a layered world today (looking bottom-up) you have: A data warehouse and decision support query tools, OLAP, and data mining. He expects more layers to appear on the top. For example, analytic servers could do heavy statistical processing on top of existing layers (e.g., for supply chain analysis). He mentioned that highly interactive tools tend to be on top, and that they typically need wide accessibility (web/Java). He also mentioned that data cleansing is a big problem that presents an opportunity for new research. (Todd Walters agreed, suggesting that academic researchers should acquire some of the tools, play with them, see what sort of SQL they generate, and then target those sorts of complex queries in their research.) Hamid gave an example of OLAP research that made it into DB2: ROLLUP and CUBE. This is an area where IBM is collaborating with Microsoft (Jim Gray) to define these as SQL extensions (specifically, they are defined as extensions to SQL aggregation). Microsoft SQL Server and DB2 UDB both support these operations in their database engines now. DB2 supports multidimensional/hierarchical ROLLUP queries in the engine, and it even parallelizes them; the research and development work for this took about 1.5 years. Hamid mentioned that, again, the first paper on this was rejected (at VLDB), while at the next SIGMOD there was a whole session on it.

Hamid mentioned one other research problem for people to think about: Today, a DBMS is one component of a bigger networked universe. Thus, DBMSs need to play in a bigger world, and queries need to be supported across all kinds of systems and components. Cooperating optimizers, etc., are needed to solve this problem.

Hamid closed by stating that Starburst had a number of other examples of successful transfer of research technology into DB2. He said that big research impact on products is not common; to achieve it, you must make the right assumptions in your work. He stated that big systems *do* need elegant solutions, and not just a pile of ad hoc components. He said that a research prototype is the place to make mistakes, and that you need to have big sandboxes for systems research. He said that there is a need for university researchers to spend time in industrial labs and vice versa (though this alone is not enough to cause successful technology transfer).

3. Academic Presentations

Next on the agenda, academic attendees gave short presentations on their current work and interests. Brief summaries follow.

Shashi Shekhar, Univ. of Minnesota. He works on spatial databases. His group has worked on clustering of map data via graph partitioning methods. Their methods beat geographical partitioning methods on road map data. They have also looked at Army GIS data, looking at declustering for parallelism using geographic methods. He emphasized that using real data gives important insights.

Peter Buneman, Univ. of Pennsylvania: His current interests are in semistructured data. Peter said that universities should not respond too much to industrial pressures but instead should focus on *ideas*. Also, he noted that all the industrial attendees represented DBMS vendor organizations; he said researchers also need to hear from consumers of data management services. Recently, Peter has focused on molecular biology and linguistics applications, where databases are typically not used; indexed files are more common. These applications have a lot of "mildly" semistructured data, which he said is not well supported by much research in semistructured data. In particular, he emphasized that just because data won't fit neatly into relational systems, we should not throw away the type system. Semistructured DB work at Penn includes new models for semistructured data, a query language for XML (joint work with AT&T Research), rapid extraction of structured data from the web, efficiently storing semistructured data (i.e., XML without DTDs) in relational databases, and support for curated databases, annotation, and data provenance (which he called good applications for semistructured data). He brought an example of data that has definite structure but "will never fit in relational database." Other DB research at Penn includes complex object query languages (OQL-based), updates in complex object query languages, and constraints in optimization, updates, and distribution.

Victor Vianu, UC San Diego: Victor is interested in semistructured data, web queries, active DBs, e-commerce, and spatial DBs. Other UCSD DB research faculty include Yannis Papakonstantinou, Walter Burkhard, Rik Belew, Ramesh Jain, and Venkat Rangan. Some of Victor's current work addresses data integration and workflows; it is XML-based, taking advantage of DTDs (type information). It involves mediators/wrappers, and a concern is distributed, evolving, active data. A distributed scientific data corpus is an example of a data problem that this work should help address. Victor listed several other problems of interest: data ingestion (should you push changes to sources?), e-commerce, active features in XML documents, and workflow specification and verification.

Clement Yu, Univ. of Illinois at Chicago: His current interests are in text databases. Others at UIC include Ouri Wolfson, Prasad Sistla, Joge Lobo, Bob Grossman, and Aris Ouksel. He is interested in text queries that span multiple databases. Questions of interest include: Which DBs to search? Which documents to retrieve from chosen DBs? He seeks to "discover" similarity functions (of different search engines). Issues in this area include space/efficiency and flexible estimation. Clement is also working on efficient query processing for text data using logic-in-memory techniques. They have built a prototype chip (a modified DRAM chip). This work is anticipating 4K processors/16 MB chip, later 16K processors/64 MB chip. Currently, they are doing algorithm design and working on a 1MB prototype. Clement gave an example problem: given medical problems/tests/treatments, detect inconsistencies and suggest problems/tests for patient treatment. (Rules should be automatically captured.)

Meral Oszoyoglu, Case-Western Reserve Univ.: She is doing work on query languages, query processing, and access structures. Her co-researcher is Gultekin Oszoyoglu. One current project is graph query languages—e.g., for multimedia presentations, WWW, and OODBMS—that can support complex queries. They are also working on graph query optimization -- to handle paths, sequences, and other graph operations that tend to be very costly. On a related topic, they are working on access structures for graph databases with heterogeneous graph nodes (and may borrow techniques from temporal index structures). The current motivating application for this work is managing and querying multimedia presentations. The structure of the web is an obvious potential application as well.

Rick Snodgrass, University of Arizona: His primary research area is temporal databases. Rick stated that temporal databases is an area where the academic community has solutions and industry should adopt them. For many applications, use of TSQL would result in significantly shorter queries, partly addressing the query complexity problem addressed earlier in Todd Walters' talk. Rick has worked on SQL3 standards, specifically on the temporal part of the specification. Professionally, he is the current SIGMOD chair. Rick believes that the SQL standards group needs to get academic reviewing and input, and that the SQL standards process is closed/broken today. (This generated considerable discussion.)

Panos Chrysanthis, Univ. of Pittsburgh: He is interested in transactions and workflow. Modeling, correctness, managing/understanding, and analysis of transaction and workflow protocols is a general area of interest to him. Another is semantics-based concurrency control, caching/replication, recovery & commit protocols, transaction & data migration. Another area is designing TP systems for network-centric data-based servers (WANs and NUMA data servers, looking at flexibility/inconsistency management). Panos is studying the scalability of existing protocols and investigating extensions to improve scalability. Also, he is looking at network/DB issues related to quality of service (QoS) provisions in new networks. His current projects are PRO-MOTION (on mobile queries and transactions) and WANDS (wide area networked database systems). His end goal (or "killer app") is what he referred to as virtual enterprises, which demand support for multi-organizational workflows.

Daniel Barbara, George Mason Univ.: His research interests include data warehousing, data mining, and data characterization and modeling. He is working on "quasi-cubes." Today, industrial approaches to cubes are materialized (eager) or computed (lazy), whereas he is taking an approach based on "quasi-cube regions," where a given cube has both materialized regions (real/error-free) and modeled regions (estimated). His initial motivation was compression, but quasi-cubes have other uses as well, including online aggregation and data mining (e.g., exploratory analysis and Bayesian network learning from estimated values).

Vassilis Tsotras, UC Riverside: Current interests include access methods, especially for temporal data, and on wireless data dissemination. He has an upcoming survey paper (in ACM Computing Surveys) with Betty Salzberg. He works on indexing of spatiotemporal data (in cooperation with ESRI); applications for this work include navigation, monitoring, and planning. (This is joint work with D. Gunopoulos.) He is also working on the management of histories of changes in semistructured data (with C. Zaniolo). In addition, he is working on efficient "slicing" of complex spatial objects (with ESRI), handling various angles/positions, e.g., for airplanes, homes, and other sorts of design DBs.

Brad Adelberg, Northwestern Univ.: He works on real-time and main memory database systems. He believes in leveraging component-level interfaces (e.g., OLE/DB). Standard research (i.e., access methods) can be difficult, because you need to understand low level details of interfaces, and also vendors may not allow numbers to be published. Interfacing common applications with DBMSs is a problem we should work on to "escape the tyranny of the file system." This problem is more general than just the web. A problem today is that database systems don't allow you to develop and evolve data and applications over time very well. Brad believes this is part of our problem in terms of why we missed the boat on the web. Another of Brad's interests is the effect of computer architecture on DBMS performance.

Ken Ross, Columbia Univ.: Ken is working on extended languages for complex queries with repeated aggregates. He is also working on nested subqueries (with Sybase), joins with join indices, sparse datacubes, and generalized aggregates in datacubes. Ken's work on materialized views includes updates (especially across systems with diverse update policies) and deciding which views to materialize. Another area of interest is data reduction and visualization. Finally, Ken is doing work in main memory query processing, especially for OLAP.

Wesley Chu, UCLA: He is interested in knowledge-based information systems and multimedia and web information systems. Wesley has worked on intelligent query processing, including query relaxation, generalization, and approximate answers. He is currently working in the medical domain (with A. Cardenas and UCLA Radiology Dept) on multimedia queries; an example query is "find similar tumors" in an image database. Research problems being addressed include temporal aspects of query processing, content-based image retrieval, web information systems, incremental query answering, query dialogues, and flexible query interfaces (based on voice, visual, and natural language input modes).

Eric Hanson, Univ. of Florida. His research focus is on scalable trigger systems. He wants to support thousands to millions of triggers involving selection, join, aggregates, and temporal conditions; he sees a need to support transaction asynchrony for scalability (because he believes that the synchronous approach won't scale). He would like to see examples of customers' real (and/or desired) trigger applications. (He is interested in pointers to such customers who might be willing to share information.) His current focus is on the TriggerMan project, in which he and his students are building a trigger DataBlade using Informix. (Their approach heavily uses SQL callbacks, and only Informix has this support available right now.) Todd Walters commented that Teradata envisions a need for triggers with very complex conditions—e.g., "tell me when this stock falls below 40% of its 52 week average, and the industry is above its six month average."

Praveen Seshadri, Cornell Univ.: He is interested in complex data types and efficient, extensible databases. His view is that every DBMS is or soon will be extensible, and he wants to go beyond the "object-relational hype." The core idea underlying his work is enhanced ("smart") ADTs. His research style is oriented towards a "roll our own" DBMS world, and to support that, his group has produced Predator, a full-fledged OR-DBMS. Currently, one of his research themes is that since data moves, so should the DBMS. He is looking at portable query processing (and he says we shouldn't need 1M lines to run a query); he is also building a portable DBMS kernel (a database virtual machine - or DVM) in the context of his Jaguar project at Cornell. He is also working on mobile query processing (e.g., for smart cards, palmtops, and embedded devices) in the Cougar project.

Joe Hellerstein, UC Berkeley: He is interested in database systems, user interfaces, and statistics. A current interest is the CONTROL project, which aims to support users with continuous feedback and control for long jobs. CONTROL topics include online aggregation (OLAP), data visualization, data mining, and associated GUI widgets. In this work, they are building a "database-sized spreadsheet"—i. e., Excel for big data sets—and have submitted a paper to SIGMOD '99 about this work. Another interest of his is the GIST (Generalized Search Tree) project, where he and his students are developing extensible indices for objects and methods, including concurrency/recovery aspects. This project also includes work on indexability theory, an index analysis/debugging toolkit (amdb), and selectivity estimation for new data types (with student Paul Aoki). He is also involved in several collaborations at Berkeley: IDISK (w/Patterson, Yelick, Kubiatowicz, Kim Keeton), where a processor and memory are built into disk (they are looking at how databases might exploit such devices); River (w/Arpaci-Dusseau et al), which is a high-performance, self-tuning, shared-nothing workflow system; and, various work on digital libraries, economic models for computation, and consulting for Cohera (also known as Mariposa, Inc., and Data Everywhere).

Jeff Ullman, Stanford Univ.: Jeff presented information about the Stanford InfoLab. Participating faculty include Hector Garcia-Molina, Mike Genesereth, Rajeev Motwani, Jeff Ullman, Jennifer Widom, Gio Wiederhold, and Tery Winograd. Included among current InfoLab projects are C3 (on incremental change management), Lore (on semi-structured DBs), Tsimmis (on information source wrapping/mediation), Whips (on warehousing), Digital Library (on a wide range of issues), LIC (on large-scale interoperation), work by the Logic group (also on heterogeneous DB integration), MIDAS (on data mining), and various HCI projects.

Betty Salzberg, Northeastern Univ.: She is interested primarily in access methods and in spatiotemporal database systems. She has done work on spatial access methods, including concurrency/recovery algorithms for them. She has worked on access methods for supporting both spatial and temporal data as well. She has also done research on the problem of online database reorganization, including parallelization of this process.

4. Summary Session

In the closing session, Phil Bernstein moderated a discussion on recommendations for improved university/industrial cooperation. Some recommendations and summary points were:

- Encourage more visits of faculty to industry
- Encourage more visits of industry folks to universities
  - Via sabbaticals
  - Via video-taped presentations (maybe w/SIGMOD as a channel?)
- Several participants noted a need for a shared code base for DB research, probably based on a research

prototype, not a product (e.g., the Predator system from Cornell)

- Access is needed and should be provided to data sets, traces, tools, etc.
- More information is needed about which systems are available for academic use.
- University consortia could be set up to support kinds of industry research; the Stanford CIS model was given as a model/example.
- We need to compile lists of:
  - Suitable problems and DBMS usage scenarios. It was noted that SIGMOD Record "Industry Perspectives" papers provide a good, already existing forum for publishing these (submissions go to *seligman@mitre.org*).
  - People interested in collaborating - for university/industry matchmaking.
- Intellectual property issues are a barrier; CRA is working on this.
- More papers are needed about industrial products.
  - Will companies write these?
  - Will professors be willing to help?
- More industry input from university researchers could/should be obtained via:
  - Corporate technical advisory board membership
  - Input to standards activities
- Consider industry support of graduate students with promise of summer
- Issue: how to get more universities producing good database students?
- It is important to get customers/users at meetings like this
  - Tool/application developers would be interesting participants, too.
- Some questions related to this workshop:
  - Is this a model to be repeated more frequently? (perhaps sponsored by industry?)
  - What should we put on our web site right away? (e.g., research problems, jobs, etc.)

5. Closing Comments

We hope that this workshop and the webpage developed from it will serve to encourage more collaboration and exchange of information between reseachers and practitioners in industry and academics. Database systems research is not like mathematics; the existence of artifacts in industry constrains the discipline. Work with impact is hard to produce without interaction between the people designing commercial systems and the people designing research prototypes and algorithms.

Acknowlegments

(written by B. Salzberg)

This workshop was sponsored by the National Science Foundation. The organizing committee thanks Maria Zemankova for her encouragement and valuable suggestions at every stage in the planning process. We could not have done it without her. Thanks especially to Mike Carey and Len Seligman for this report and Phil Bernstein for his help with the webpage contents and the summary. Thanks to the organizing committee: Mike Carey, Joseph Hellerstein, and Ashish Gupta. Also, thanks to Matt Pietal for his continuing good work as webmaster.

The workshop was funded by the National Science Foundation, the Information and Data Management Program, under grant IIS-9815933. All opinions, findings, conclusions and recommendations in any material resulting from this workshop are those of the workshop participants, and do not necessarily reflect the views of the National Science Foundation.