Reminiscences on Influential Papers

Kenneth A. Ross, editor

See `http://www.acm.org/sigmod/record/author.html` for submission guidelines.

Rada Chirkova, North Carolina State University, `chirkova@csc.ncsu.edu`.

[Ron Avnur and Joseph M. Hellerstein. Eddies: Continuously Adaptive Query Processing. Proceedings of the ACM SIGMOD Conference, 2000, pages 261 272.]

The then newly published Eddies paper was on the reading list for the qualifying examination in databases that I took in my Ph.D. program. The wonderfully clean and beautiful scheme put on its head the world of query optimization I had assumed was the only one possible. In fact, this paper is all about questioning implicit assumptions behind classic query optimization. Is it always true that query-evaluation performance does not uctuate during query execution? Can we be sure that costs or selectivities do not change during execution time? Should we always strive for best-case performance? The authors argue that these assumptions do not necessarily hold in all environments where query processing takes place. Their philosophy of favoring adaptivity over best-case performance leads to a di erent approach of ne-grained, adaptive, online reoptimization, where performance comes from reoptimizing query execution plans regularly during the course of processing a single query. This way, the system is allowed to adapt dynamically to uctiations in computing resources, data characteristics, and user preferences.

In the classic optimization scheme, stored data in query execution are viewed as a monolithic set where individual tuples are, in a sense, indistinguishable. As a result of doing away with this assumption in the approach described in the paper, the eddy query-processing operator can focus on individual tuples coming from data sources, continuously reordering the application of pipelined operators in a query plan to each tuple. This idea is the basis of a radically di erent optimization/execution architecture. Interestingly, besides delivering runtime adaptivity and reduced code complexity in its target unpredictable environments, eddies can still give good results in tandem with traditional optimizers. The personal lesson I took away from this paper was that when approaching a problem, it is always good to start by questioning your assumptions. If you do that, then there's always hope that something useful and really beautiful will come out of your work.

Dimitrios Gunopulos, University of California at Riverside, `dg@cs.ucr.edu`.

[Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, A. Inkeri Verkamo. Fast Discovery of Association Rules. In Advances in Knowledge Discovery and Data Mining, edited by Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy. AAAI/MIT Press 1996: 307 328.]

This is among the rst papers on Data Mining that I have read. At the time I was a Post-doc in Max-Planck-Institut, having just nished my Ph.D. in Computational Geometry. I had asked Heikki Mannila for interesting problems to work on, and I have been working on Data Mining problems ever since. The ideas in this paper and the related papers that predate it are of course fundamental in the eld of Data Mining, have helped shape the eld, and have motivated and in uenced a tremendous body of research. The apriori approach for computing association rules is elegant, simple, yet a ords excellent performance. This paper had a signi cant in uence on work that I did and one of the main reasons was that it provided a solid theoretical foundation to address such a practical problem. Such subsequent work included studying the performance of the apriori algorithm, developing new algorithms, studying the complexity of the problem,

and identifying surprising connections between the data mining problem of nding association rules and seemingly unrelated topics such as the problem of subspace clustering or problems in computational learning theory.

Rachel Pottinger, University of British Columbia, rap@cs.ubc.ca.

[Je rey D. Ullman. Information Integration Using Logical Views. ICDT 1997, pages 19 40]

During my rst year of graduate school I took a course on Intelligent Information Systems, which was a joint class for the AI and the database groups. At the time, I had never taken a database course and had no intention of doing so; I took the class because I was interested in being an AI student. Reading this paper helped me to understand what database research is and why it is interesting.

The paper begins with a short synopsis of theory and algorithms for query containment and answering queries using views. Using this foundation, the paper describes the basics of a data integration system and then compares and contrasts the theory behind two data integration systems: the Information Manifold and Tsimmis. Ullman clearly delineates the two systems without getting bogged down by details. Ullman's writing is lucid as always, and since the paper provides just enough theory to understand the systems, it is a perfect primer for those who are new to the subject. To this day, if I want to give people a paper that will introduce them to query containment or database theory in general, I give them this paper.

Jun Yang, Duke University, junyang@cs.duke.edu.

[Shaul Dar, Michael J. Franklin, Bjorn Thor Jonsson, Divesh Srivastava, and Michael Tan. Semantic Data Caching and Replacement. In Proceedings of the 22nd International Conference on Very Large Data Bases (VLDB), September 3 6, 1996, Bombay, India.]

This seminal paper by Dar. et al. introduced the idea of *semantic caching*, an approach that combines materialized views and caching. Traditionally, caching is done at the object level, and hence only supports access to objects by identi ers. When the cache receives a declarative query, e.g., a selection involving non-id attributes, it is generally impossible to tell whether the cache provides a complete answer; we would always have to query the source data to ensure completeness. By remembering semantic descriptions of cached data (i.e., view de nitions), we can determine the completeness of query answers and only query the source data when necessary. This feature makes semantic caching perfect for applications that require declarative accesses to cached data. The approach has attracted a lot of attention in recent years because of applications such as caching for mobile data accesses and database-driven Web sites.

This paper had a great deal of in uence on the formation of my research agenda when I started as a faculty member at Duke University. I had worked on materialized views as a graduate student. The last problem I tackled in my dissertation was a class of views (temporal aggregates) that are too expensive to materialize because of updates; the solution was to forgo direct materialization of the view and instead materialize an index that supports e cient updates and accesses to the view. As I was looking for other ways to exploit the connection between views and indexes, I ran into the semantic caching paper by chance. It led me to realize something much more general and inspired me to consider the problem in a larger context. There exists a strong connection among caches, views, and indexes (and also replicas, continuous queries, synopses, etc.), because they are all *derived data* the result of applying some transformation, structural or computational, to *base data*. Indeed, the use of derived data to facilitate access to base data is a recurring technique in computer science. Although there has been a lot of work on derived data, most techniques were developed and applied separately for di erent forms of derived data. Newer and more complex data management tasks, however, call for creative combinations of traditionally separate ideas. Time and again, I nd myself using

semantic caching to help explain the theme underlying my recent research in combining and unifying derived data maintenance techniques, because semantic caching is such a compelling and inspiring example of doing so.

Jingren Zhou, Microsoft Research, jrzhou@microsoft.com.

[Goetz Graefe. Query Evaluation Techniques for Large Databases. ACM Computing Surveys 25(2), 1993, pages 73 170.]

I rst read this paper in early 2000 while I was a graduate student at Columbia University. This paper was extremely impressive with its deep and thorough review of query evaluation techniques for very large relational databases. It not only explains di erent query processing algorithms, but also provides the intuition behind these evaluation techniques, answering questions like *why* they are designed this way and *what* are the crucial design issues.

The Graefe paper is long and can take time to read and understand completely. I remember reading it many times. Each time, I was amazed more by its comprehensive and intuitive presentations and was inspired in one way or another. Personally, I particularly liked its crisp explanation of iterator models and its profound discussions of sorting and hashing techniques. It shaped my thinking and in uenced my research later on. During my PhD, I frequently came back to this paper and used it as my table reference. I still have my early copy of this paper, lled with colored highlights and marginal comments. I consider this paper a must-read for every database student.
