Reminiscences on Influential Papers

This column was established by Richard Snodgrass in 1998 and was continued by Ken Ross from 1999 to 2005. It celebrated one of the key aspects that makes us grow as a research community: *the papers that influence us*. At each issue, different members of the data management community wrote anecdotes about a paper that had a unique impact in their career. The anecdotes highlighted that *impact* can come in many forms. A paper's value is not only in its citation count, but also in the way it influences individuals who in turn influence other individuals that make up our community. Such impact is not countable.

When the SIGMOD Record's editor-in-chief Rada Chirkova approached me to revive this column last year, I was immediately excited. I would like to thank Rada Chirkova, Richard Snodgrass, and Ken Ross for this opportunity. I am delighted to present the three invited contributions for this issue. Hope you enjoy reading them as much as I did.

While I will keep inviting members of the data management community, and neighboring communities, to contribute to this column, I also welcome unsolicited contributions. Please contact me if you are interested.

Pınar Tözün, *editor*
IT University of Copenhagen, Denmark
pito@itu.dk

M. Tamer Özsu
University of Waterloo, Canada
tamer.ozsu@uwaterloo.ca

When you have been in business as long as I have, it is hard to select one paper that influenced me - so I cheat and select two: one that influenced how I approach new topics and the second that influenced my career in a fundamental way.

Jim Gray.
Notes on Data Base Operating Systems.
In Operating Systems, An Advanced Course, Lecture Notes in Computer Science, Vol. 60, pages 393-481, Springer, 1978.

This paper was first published as an IBM San Jose Laboratory Technical Report in 1977, which is the form I first read it. At that time, we used to subscribe to technical reports from a few database pioneer research places and IBM San Jose was one of them. This is an exceedingly well-constructed paper that properly frames the issues and walks through them systematically. At the time, I was doing my master's and transitioning to computer science from an undergrad degree in industrial engineering (with a whole bunch of undergrad CS courses to annoy advisors who insisted that I gain some breadth). But I did not have sufficient foundational knowledge. As I read materials, I kept trying to develop a reference model in my mind as to what the important pieces were and how they fit together. Jim's paper was eye opening to me in how systematically he framed the issues and developed the arguments. I have used it as a mental template when I tackle a new area and try to write a framework, at least for my own understanding.

Incidentally, the paper contains the following sentence: "If one tries to implement such an application on top of a general purpose operating system it quickly becomes clear that many necessary functions are absent from the operating system." After 45 years and tremendous development, it is interesting that DBMS-OS co-design people would probably still say the same thing.

Michael Stonebraker and Eric Neuhold.
A Distributed Database Version of INGRES.
In Proceedings of 2nd Berkeley Workshop on Distributed Data Management and Computer Networks, pages 9-36, 1977.

Berkeley Workshops, as they were known, ran for a few years and were sources of great insight into new directions. This paper originally appeared as a Berkeley technical report in 1976 and then was published in this workshop's proceedings which is where I read it. This was when INGRES and System R were being built as first relational DBMSs at Berkeley and IBM San Jose Laboratory, respectively. Thus, this paper appeared very early during INGRES development. It describes the extension of the system as one that runs on a single UNIX machine to run on multiple UNIX machines assuming "the existence of the UNIX to UNIX communication facility being constructed by the UNIX designers." This was early in the development of both database and computer network technologies, and I was new to both. I was desperately trying to wrap my head around how to reconcile the "integration" objective of database systems with "decentralization" objective of computer networks. The design of the multi-machine version of INGRES was eye opening to me and I decided to do my PhD in distributed databases. The rest is history.

Alberto Lerner
University of Fribourg, Switzerland
alberto.lerner@unifr.ch

Donovan A. Schneider and David J. DeWitt.
A Performance Evaluation of Four Parallel Join Algorithms in a Shared-Nothing Multiprocessor Environment.
In Proceedings of ACM SIGMOD, pages 110-121, 1989.

The paper dates back to 1989 and compares the performance of best-known hash-based join algorithms when executed in a parallel setting. We might call it an Experiment & Analysis paper nowadays, but it was much more than that. It proposed parallel formulations that did not exist before for centralized join algorithms and analyzed their performance over a rich set of scenarios. There were several important insights in the paper. It found that hash-based join algorithms were quite amenable to parallelization. It showed how the availability of main memory could be crucial to performance. It also called attention to the fact that skewed data affects the algorithms. The authors stopped short of suggesting skew handling techniques. This came at a later paper. Interestingly, the authors also performed experiments with *diskless* workstations in a setting that resembles what we call *serverless* today.

The paper represents the period when databases were essentially being redesigned for parallel execution. At the time, parallelism came from using a set of networked workstations. Little did we know that ten years or so after the paper was published, we would be ushered into the multicore era. Many of the paper's techniques are still applicable today, and the paper (and a cluster of related ones) have been regularly cited ever since.

I came upon the topic of parallel databases when I was looking for problems to work on for my master's degree. The paper was part of a compilation book much like the "Red Book," only this time it focused specifically on parallel query processing papers, and it was green. The compilation appeared in 1994, edited by Hongjun Lu, Beng-Chin Ooi, and Kian-Lee Tan. Its opening paper is Gray's and DeWitt's presentation of parallel database systems as the future of high performance. When I found the paper, clusters of ethernet-connected workstations were quite common, which allowed me to reproduce the results. I still remember the thrill of writing parallel software and engaging multiple machines to attack increasingly large data sets. The experience sparked an interest in query execution techniques that I keep to this day.

Tianzheng Wang
Simon Fraser University, Canada
tzwang@sfu.ca

Ryan Johnson, Ippokratis Pandis, Radu Stoica, Manos Athanassoulis, and Anastasia Ailamaki.
Aether: A Scalable Approach to Logging.
In Proceedings of the VLDB Endowment, Vol. 3, Issue 1-2, pages 681-692, 2010.

In fall 2012 I arrived at the University of Toronto as a new grad student hoping to work on *systems*. Database systems, of course, is one of these areas, yet I wasn't quite sure about pursuing it coming from a storage and embedded systems background. It was the Aether paper that settled it. Although the title says it's about logging, there you find all kinds of delicate interactions and design issues touching various components inside and beyond the DBMS kernel to make logging (and hence the whole system) scalable: concurrency control, synchronization, OS CPU and I/O scheduling and more! I was in particular fascinated by the ideas of

commit pipelining and early lock release, where one can get throughput like asynchronous commit yet without losing correctness. Of course, the *fineprint* was that we are trading latency for throughput, a classic tradeoff in high-performance storage engines. It was also fun to think about the twist on *group* vs. *pipelined* commit, although in many cases we take pipelining for granted when we say *group commit*.

The bulk of my PhD thesis got inspirations from this paper when I explored persistent memory and RDMA for scalability and reliability at even bigger scales than Aether could handle. To this day, the fundamental principles in this paper still find their way into my own research group's latest work. This paper remains my favourite recommendation for anyone who would like to get a taste on what and how to think about when building a transactional storage manager.