Reminiscences on Influential Papers

Kenneth A. Ross, editor

I continue to invite unsolicited contributions. See `http://www.acm.org/sigmod/record/author.html` for submission guidelines.

Mitch Cherniak, Brandeis University, `mfc@cs.brandeis.edu`.

[John Backus. Function Level Programs as Mathematical Objects. Proceedings of the 1981 conference on Functional Programming Languages and Computer Architecture, 1981, Portsmouth, New Hampshire, October 18–22, 1981.]

I must admit certain misgivings about citing a paper from outside the database community as my choice for the "Most Influential Paper" column appearing in SIGMOD Record! But no paper had greater influence on my Ph.D research than Backus' seminal work in Functional Programming. When I was a second year Ph.D student, I was handed the task of using a theorem-prover to verify a set of query optimization rules expressed over a high-level query algebra. After spending many frustrating hours failing to express the identities using the theorem prover tool, I turned to some papers from the Programming Language community to see if the problem might lie within the query algebra itself. This paper introduced me to the simplicity and elegance of combinators as a means of reasoning about programs (and by extension, queries) as first-class objects subject to algebraic transformations. Not only did this influence the direction I took in my Ph.D dissertation (which was about using a combinator-based query algebra as the basis for expressing provably correct query rewrites), but it also instilled in me an ethos for programming language design that influences still, my sensibilities concerning query languages and algebras.

Gottfried Vossen, University of Muenster, `vossen@uni-muenster.de`.

[P.A. Bernstein and N. Goodman. A Sophisticate's Introduction to Distributed Database Concurrency Control. Proc. 8th Int. Conf. on Very Large Data Bases (VLDB), Mexico City, September 1982, 62–76.]

Database transactions had long been invented in 1982, and developers were already busy implementing this concept in commercial database management systems; in fact, that was the time when everybody thought that "distributed" databases would come in the next system generation. The literature already had quite a number of concurrency control and recovery protocols, and at the time people were eagerly extending algorithms for centralized databases to the distributed case. However, not too much had been done from a formal point of view, and there was no textbook or monography on the topic yet. I was a young graduate student looking for a neat exposition to the field when I came across this paper. Bernstein and Goodman made it clear that (in a distributed scenario, where certain activities may happen in parallel at distinct sites) transactions and schedules (which were called transaction logs and DDBS logs, resp.) were partial orders of steps, they formalized view-serializability through reads-from relations and final writes, and stated a serializability theorem saying that schedules with an acyclic conflict graph were view-serializable. Moreover, they surveyed protocols such as 2PL, TO, SGT, and optimistic concurrency control ("certifiers"), extended all of this to the distributed scenario, and completed their survey with mentioning the possibility of heterogeneous protocols (where separate schedulers take care of rw and ww conflicts) as well as of multi-version concurrency control.

The paper was intended as a unification of a number of algorithms that had been published until then on the topic of distributed concurrency control, but it served much more than just this purpose. I was fascinated by the simplicity and elegance that the formal foundations of the field were all of a sudden beginning to get. For the first time, I got a clear picture of transaction fundamentals, and for a long time I used the formal framework for classes I had to give on the topic. For me, it was a breakthrough paper that even influenced my own writings on the topic many years later.
