Paper: The End of an Architectural Era (It's Time for a Complete Rewrite)

Thursday, April 16, 2009 at 1:16AM

Todd Hoff in Database, Paper

Update 3: A Comparison of Approaches to Large-Scale Data Analysis: MapReduce vs. DBMS Benchmarks. *Although the process to load data into and tune the execution of parallel DBMSs took much longer than the MR system, the observed performance of these DBMSs was strikingly better.*
Update 2: H-Store: A Next Generation OLTP DBMS is the project implementing the ideas in this paper: *The goal of the H-Store project is to investigate how these architectural and application shifts affect the performance of OLTP databases, and to study what performance benefits would be possible with a complete redesign of OLTP systems in light of these trends. Our early results show that a simple prototype built from scratch using modern assumptions can outperform current commercial DBMS offerings by around a factor of 80 on OLTP workloads.*
Update: interesting related thread on Lamda the Ultimate.

A really fascinating paper bolstering many of the anti-RDBMS threads the have popped up on the intertube lately. The spirit of the paper is found in the following excerpt:

In summary, the current RDBMSs were architected for the business data processing market in a time of different user interfaces and different hardware characteristics. Hence, they all include the following System R architectural features:
** Disk oriented storage and indexing structures*
** Multithreading to hide latency*

** Locking-based concurrency control mechanisms*
** Log-based recovery*

Of course, there have been some extensions over the years, including support for compression, shared-disk architectures, bitmap indexes, support for user-defined data types and operators, etc. However, no system has had a complete redesign since its inception. This paper argues that the time has come for a complete rewrite.

Of particular interest the discussion of H-store, which seems like a nice database for the data center.

H-Store runs on a grid of computers. All objects are partitioned over the nodes of the grid. Like C-Store [SAB+05], the user can specify the level of K-safety that he wishes to have.
At each site in the grid, rows of tables are placed contiguously in main memory, with conventional B-tree indexing. B-tree block size is tuned to the width of an L2 cache line on the machine being used. Although conventional B-trees can be beaten by cache conscious variations [RR99, RR00], we feel that this is an optimization to be performed only if indexing code ends up being a significant performance bottleneck.
Every H-Store site is single threaded, and performs incoming SQL commands to completion, without interruption. Each site is decomposed into a number of logical sites, one for each available core. Each logical site is considered an independent physical site, with its own indexes and tuple storage. Main memory on the physical site is partitioned among the logical sites. In this way, every logical site has a dedicated CPU and is single threaded.

The paper goes through how databases should be written with modern CPU, memory, and network resources. It's a fun an interesting read. Well worth your time.
