Letter from the Special Issue Editor

SQL Strikes Back! (a.k.a. The SEQUEL to NO-SQL)

Extremely large amounts of data are generated routinely today, by a number of sources such as Web sites, genome sequence and microarray data, network monitoring data, sensor data, and many other kinds of sources; such data is commonly referred to as "Big Data". The map-reduce paradigm has proven particularly successful for processing such Big Data. The availability of open-source map-reduce implementations such as Hadoop has lead to widespread use of the map-reduce paradigm.

However, many data processing needs that can be expressed very concisely in SQL, need an inordinate amount of coding effort in a map-reduce system, since the programmer has to express the logic of the computation imperatively, and further has to make choices about the implementation. The above problems with processing of Big Data have been addressed using two complementary approaches, which are represented by different papers in this special issue.

The first approach continues to use the map-reduce programming paradigm, but seeks to automate the task of optimizing map-reduce programs. For example, the paper by Herodotou and Babu discusses how to automate the choice of parameters for the map-reduce system, which has been shown to significantly improve performance. The paper by Dittrich et al. shows how to exploit the redundancy already present in the HDFS (and other similar) storage system to create and store different indices and different physical layouts in the different copies, which can significantly speed up map-reduce jobs. The paper by Kwon et al. addresses the problem of execution skew in Hadoop, showing how to reduce skew by more sophisticated initial partitioning, and by run-time adaptation through dynamic repartitioning.

The second approach provides programmers a declarative means of specifying their requirements, allowing the implementation to choose the best way of executing the declarative specification. Three systems that follow this approach are represented in this special issue: Apache Pig, SCOPE from Microsoft, and Hyracks from UC Irvine. The use of such declarative specifications has already overtaken low-level map-reduce specifications for many applications; for example, it has been reported that a large fraction of the map-reduce applications at Facebook have been replaced by queries in Hive, which supports an SQL-like declarative language.

The paper by Gates et al. from Hortonworks describes several optimizations that have been implemented in the Pig system, as well as some that are being considered for implementation. The SCOPE system from Microsoft allows programmers to integrate declarative specifications in an SQL-like language with imperative code written using a generalized map-reduce-combine framework. The declarative specification enables cost-based optimization, but a particular problem with optimization for many Big Data applications is the lack of statistics as well as the lack of detailed cost models for user-defined operations. The paper by Bruno et al. describes how information from earlier runs can be used to better estimate costs of alternative plans, and thus speed up future runs of a job. Since many jobs are executed repeatedly, their approach is particularly useful for such recurring jobs. Finally, the paper by Borkar and Carey describes a common algebraic framework which can support multiple Big Data languages such as SQL, Pig, or HiveQL. The algebraic framework implemented in the Alegbricks platform, gives users a choice of language appropriate for their needs, while allowing a single underlying implementation layer. The algebraic nature of the Algebrics platform lends itself to cost-based optimization (an area of future work for the Algebricks project).

With support for declarative querying, and query optimization, combined with transaction support in a new generation of massively parallel data management systems such as Google's Megastore and Spanner, it is clear that database technologies are back in the Big Data world. The articles in this issue provide an excellent insight into recent developments in query optimization for Big Data systems, and I'm sure you will find them as exciting a read as I did.

S. Sudarshan
IIT Bombay