BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data

(Extended Abstract)*

Sameer Agarwal+, Aurojit Panda+, Barzan Mozafari*, Samuel Madden*, Ion Stoica+

+UC Berkeley *MIT CSAIL

Companies increasingly derive value from analyzing large volumes of collected data. There are cases where analysts benefit from the ability to run short exploratory queries on this data. Examples of these exploratory queries include root-cause analysis, problem diagnosis on logs (such as identifying the cause of long start-up times on a video streaming website), and in analyzing the effectiveness of an ad campaign in real time. For many such queries, timeliness is more important than perfect accuracy, the queries are ad-hoc (i.e., they are not known in advance), and they involve processing large volumes of data.

Achieving short, bounded response times for queries on large volumes of data has remained a challenge due to limited disk bandwidth, inability to fit these datasets in memory, considerable network communication overhead of large data shuffles, and variability in task completion times. For instance, executing a simple aggregate query on a few terabytes of data spread across hundreds of machines may take tens of minutes when run on top of these distributed frameworks. This time is further exacerbated by unpredictable delays due to stragglers or network congestion during large data shuffles. Such delays severely impact the analyst's ability to carry out interactive exploratory analysis on data.

In this poster, we introduce BlinkDB, a query engine running on top of a variety of distributed frameworks, intended to provide support for ad-hoc, low-latency queries. BlinkDB accepts SQL-like queries and adds extensions which allow aggregation queries to be annotated with either error or maximum execution time constraints, which the system uses for satisfying the user's requirements as well as for optimizing query execution. For instance, the following query in BlinkDB can either be augmented to return the fastest possible answer within a relative error of $\pm 10\%$ with 95% confidence or can be made to return the most accurate answer within 5 seconds, both augmented by appropriate error bars and confidence intervals.

```
SELECT COUNT(*)
FROM Sessions
WHERE Genre = 'western'
GROUP BY OS
ERROR 0.1 CONFIDENCE 95% -OR-
WITHIN 5 SECONDS
```

As shown in Fig. 1, BlinkDB accomplishes the real-time interactiveness by pre-computing and maintaining a carefully-chosen set of samples from the data, and executing queries on an appropriate sample that meets the error and/or time constraints of the query. The system uses a variety of statistical techniques to provide error bounds for queries executed on these small samples. For common aggregation functions such as average, sum and count, the system relies on known closed-form equations. We are also investigating

*The first and the second author are students.

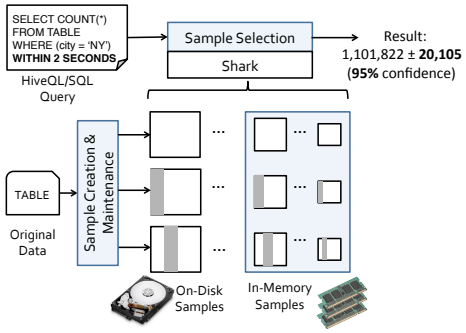

Figure 1: BlinkDB Architecture

the use of bootstrap, in particular Bag of Little Bootstraps [1], for providing error estimates on arbitrary user-defined functions. Bag of Little Bootstraps is a version of bootstrap better suited for distributed implementations.

To be able to handle queries that require samples from relatively infrequent sub-groups, BlinkDB maintains a set of *uniform* samples as well as several sets of *biased* samples, each *stratified* over a subset of columns. Maintaining *biased* samples for the superset of columns requires an exponential amount of storage and is hence, impractical. On the other hand, only *stratifying* on those subsets of columns that have appeared in past queries will limit the applicability of BlinkDB for ad-hoc queries. Therefore, we rely on an optimization framework to determine the set of columns on which to bias our samples. The optimization formulation takes into account the data distribution, past queries, storage constraints and several other system-related factors. Using these inputs, BlinkDB determines a set of samples which would help answer subsequent queries, while limiting additional storage used to a user configurable quantity. The samples themselves are both *multi-dimensional* (*i.e.,* biased over different columns), and *multi-resolution* (*i.e.,* of a variety of sizes) and are stored on disk or cached in memory across hundreds of nodes. This enables BlinkDB to efficiently answer queries with varying accuracy (or time) constraints, while minimizing response time (or error).

BlinkDB is open-sourced[1] and is under active development. Our initial set of experiments show that BlinkDB can execute a range of queries from a real-world query trace on up to 17 TB of data and 100 nodes in 2 seconds, within an error bound of $2 - 10\%$.

1. REFERENCES

[1] A. Kleiner, A. Talwalkar, P. Sarkar, and M. I. Jordan. Bootstrapping big data. In *Big Learn*, 2011.

[1] http://blinkdb.cs.berkeley.edu