**Cartilage:** Flexible Hadoop Skeleton

Alekh Jindal
CSAIL, MIT

Jorge Quiané
Data Analytics Group, QCRI

Samuel Madden
CSAIL, MIT

### Current Practice: modify the Hadoop code

**Issues**
1. Improved performance only for specific workloads
2. Still a hard-coded upload plan
3. Deep changes to Hadoop
4. Hard to use

**Research Challenges**
1. How to adapt storage to a large variety of workloads?
2. How to provide flexibility without code changes?
3. Flexibility vs Ease-of-Use vs Efficiency
4. How to preserve fault-tolerance?

**Cartilage**

**Idea**
1. Introduce a declarative upload plan
2. Decouple users datasets from physical files
3. Allow for flexible query processing

**Benefits**
1. Flexible Upload Plan
2. Easy implementation of any data layout
3. Enable storage heterogeneity
4. Allow for new applications

### Large Variety of Upload Plans

(a) Emulating HDFS

(b) Flexible data replication

(c) Heterogenous storage

(d) Heterogenous partitioning

**Improved Performance**

- **HDFS**
  - **Projection**
  - **Selection**
  - **Aggregation**
  - **Join**

- **DBMS-X**
  - **Projection**
  - **Selection**
  - **Aggregation**
  - **Join**

**DBMS-X**

- **Projection**
- **Selection**
- **Aggregation**
- **Join**

**DBMS-X** is able to outperform Hadoop by a rather significant factor across all cluster scaling levels. Although the relative percentage gain decreases with increasing cluster size, the absolute performance is still quite impressive. For example, with a 6000-node cluster, **DBMS-X** is able to achieve a performance gain of approximately 36,000 records per second, compared to 15,000 records per second for Hadoop. This is in spite of the fact that each node still executes the query in a reasonable amount of time, with the Hadoop program taking around 24.3 seconds and **DBMS-X** taking around 12.7 seconds. In contrast, **DBMS-X** is able to handle the query much more quickly, with an execution time of around 0.4 seconds per worker.

**Vertica**

- **Projection**
- **Selection**
- **Aggregation**
- **Join**

**Vertica** is another parallel database system that can be used to implement the MR program. However, it is not as well-suited to the task at hand as **DBMS-X**. **Vertica** is designed to be a columnar database, which makes it well-suited to handling large data sets. However, it is not as well-suited to handling small data sets, which is the case for the MR program. As a result, **Vertica** is not able to outperform Hadoop in this case. Instead, it is able to outperform Hadoop by a relatively small factor, with an absolute performance gain of around 10,000 records per second. This is in spite of the fact that each node still executes the query in a reasonable amount of time, with the **Vertica** program taking around 30 seconds to execute the query.

**DBMS-X** and **Vertica** are both well-suited to handling large data sets. However, **DBMS-X** is better suited to handling small data sets. As a result, **DBMS-X** is able to outperform Hadoop by a significant factor across all cluster scaling levels. This is in spite of the fact that each node still executes the query in a reasonable amount of time, with the **DBMS-X** program taking around 24.3 seconds and the Hadoop program taking around 12.7 seconds. In contrast, **DBMS-X** is able to handle the query much more quickly, with an execution time of around 0.4 seconds per worker.