HyperPartitioning
Anil Shanbhag (anils@mit.edu), Yi Lu (yil@mit.edu)

Introduction
- Data partitioning significantly improves the query performance in distributed database systems.
- Existing techniques focus on finding best data partitioning given a workload.
- Many modern analytics applications involve ad-hoc/ exploratory analysis.
- Workloads might change over time.
- Static workload-based partitioning not suitable.

HyperPartitioning is a multi-attribute data partitioning approach which does not require an upfront workload and adapts to the user queries.

Our Approach
- Many data warehouses use a block-based storage system like HDFS to store their data. Instead of creating the blocks based on size, we create blocks based on a partitioning tree.
- During data upload time, we use the upfront partitioning algorithm to generate a balanced partitioning tree. The tree is created such that:
  1. Average number of ways of partitioning of each attribute is the same.
  2. Each leaf node (~ block) has same size.
- At runtime, when a user submits a query, the optimizer is run. The optimizer tries to improve the partitioning by replacing / reorganizing the partitioning tree based on query predicates.
- The re-partitioning interleaves with query execution there-by sharing scan. Also, we never re-partition data not accessed by the query.

Experimental Results
- Upfront data partitioning leads to 2x overhead.
- Optimization time is very small (1-5 seconds).
- First experiment is on system behaviour for different query patterns:
- When applied to a real-world workload: