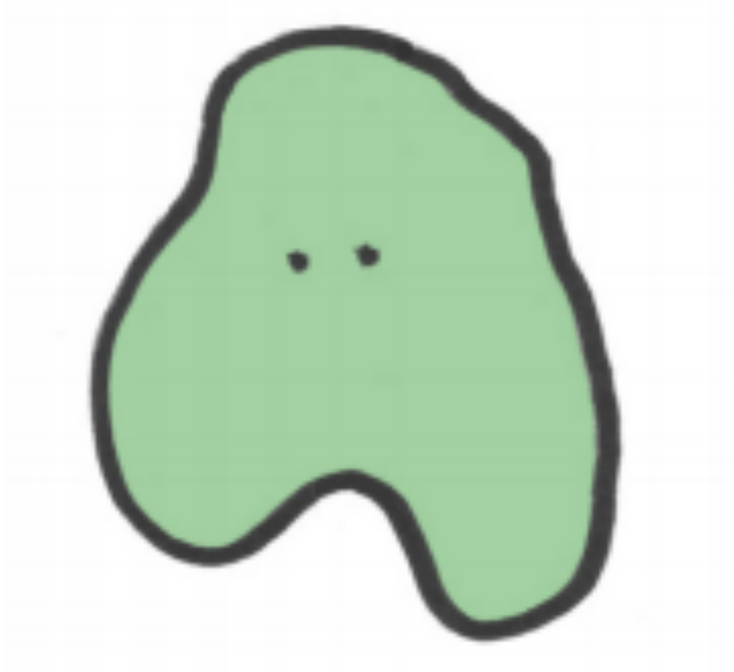


Amoeba A Shape changing Storage System for Big Data

Anil Shanbhag, Alekh Jindal, Yi Lu, Samuel Madden

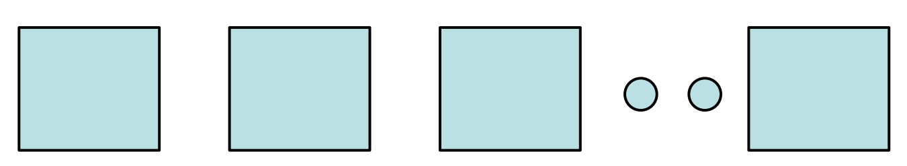
The Problem !

- Data partitioning is important !
- Modern analytic applications involve ad-hoc/ exploratory analysis. There is no fixed workload or it change over time.
- Static workload-based partitioning fails
- Enter Amoeba !

Our Approach

Amoeba is a relational storage system on top of HDFS (like Hive / Parquet) for the Hadoop ecosystem

It uses an adaptive data partitioning approach which does not require an upfront workload and adapts to the user queries.

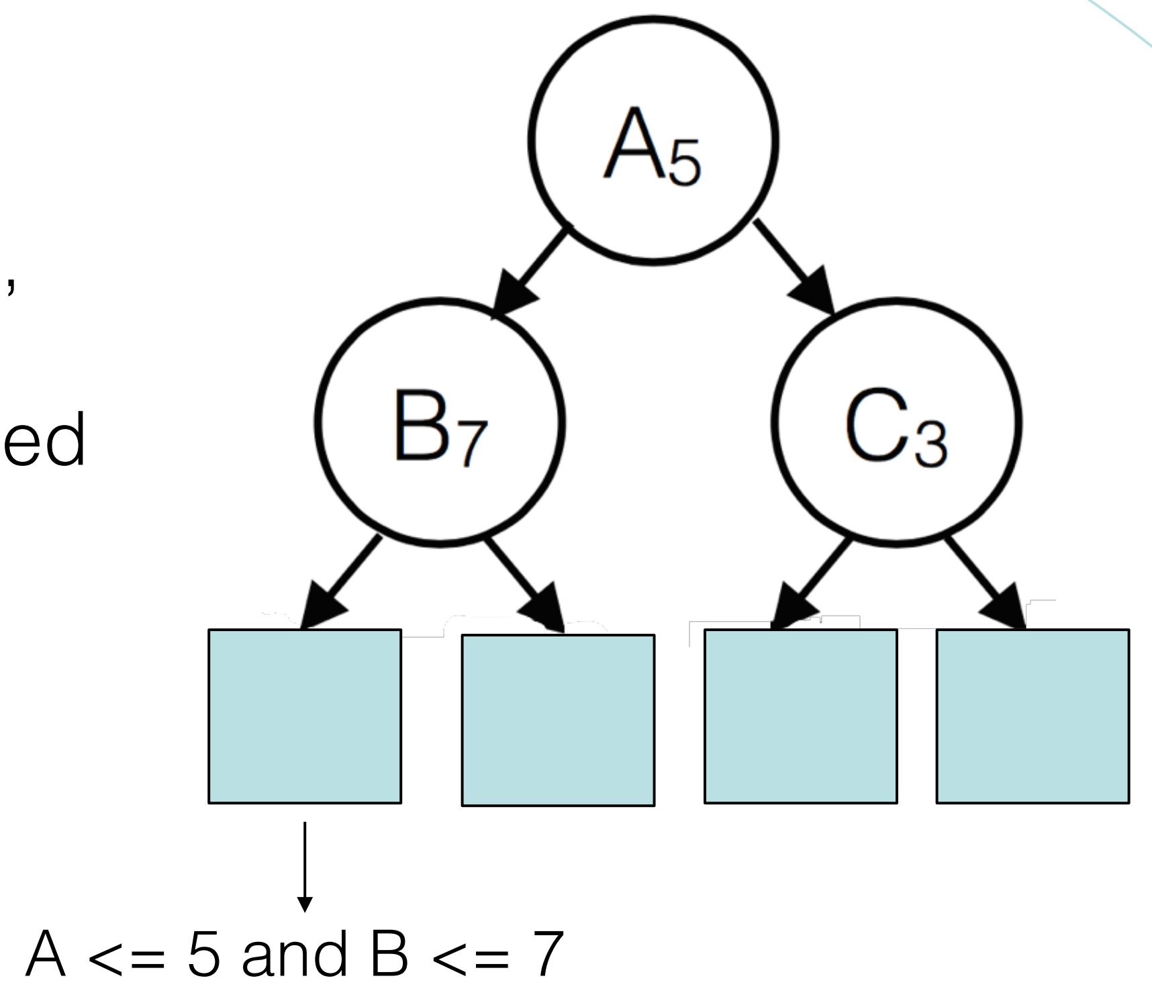


In block-based systems like HDFS, files broken into blocks (128 MB chunks)

Upfront Partitioning

Instead of partitioning by size, partition by attributes.
Same number of blocks created as in HDFS. Each block now has additional metadata

Distribute partitioning effort across attributes based on **Allocation**



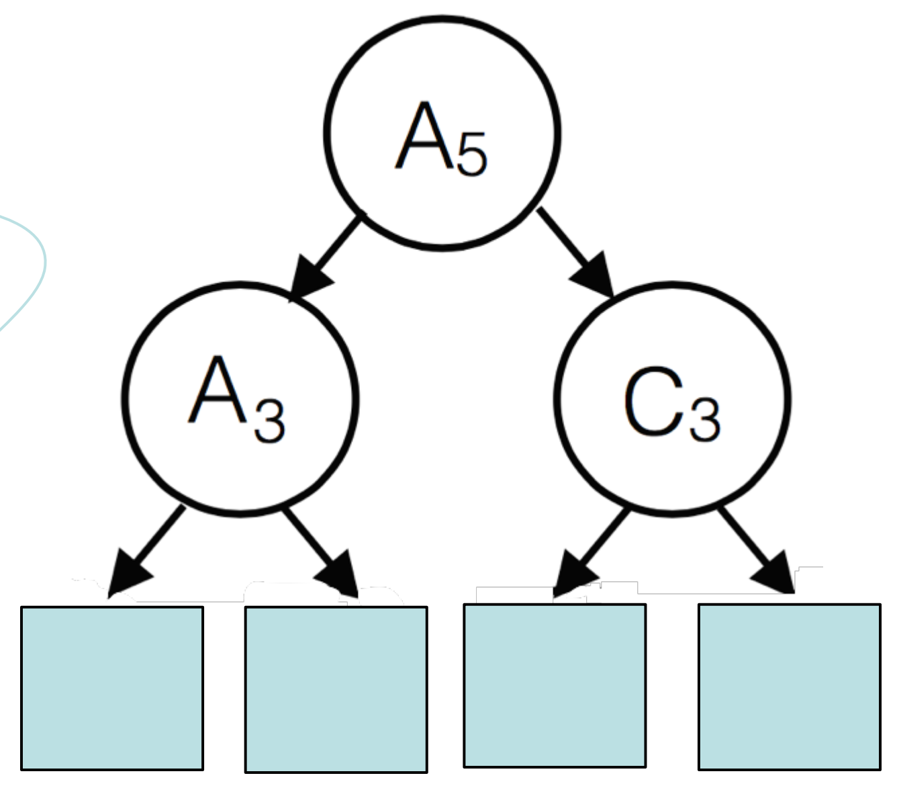
Allocation$_j$ (average partitioning of an attribute j) $= \Sigma\, n_{ij}\, c_{ij}$

n_{ij} is number of ways node i partitioned on attr j

c_{ij} is the fraction of data this applies to

Done on datasets which are O(TB) with ~ > 8000 node partition trees.

Adaptive Re-Partitioning



When user submits a query, optimizer tries to improve the partitioning by reorganizing the tree partitioning
Here if queries ask A <= 3 many times, replace B_7 by A_3

Cost Model

$$\text{Cost}(T, q) = \sum n_b$$

$$\text{RepartitioningCost}(T, q) = \sum_{b \in B} c \cdot n_b$$

Repartitioning ONLY happens when reduction in the total cost of the query workload is greater than re-partitioning cost.

Put on your Data Analyst Hat

Using a stream of ad-hoc queries on an Internet-of-Things dataset, examine the trade-offs involved with using Amoeba vs a static storage system

(1) At what point should we re-partition the data ?

Use the robust/reactive knob to control reactiveness to changes in workload

(2) See improved time-to-first-query

With no information, why does a ad-hoc query like trip length ⊂ (1,2) run 2x faster with Amoeba

(3) Runtime gains over sequence of queries

See how multi-dimensional adaptivity matches against static-partitioning schemes