CLARINET: WAN-Aware Optimization for Analytics Queries

Presented By Robert Claus

Agenda

1. The Problem
2. Clarinet
3. Optimizing WAN Queries
4. Results

Agenda

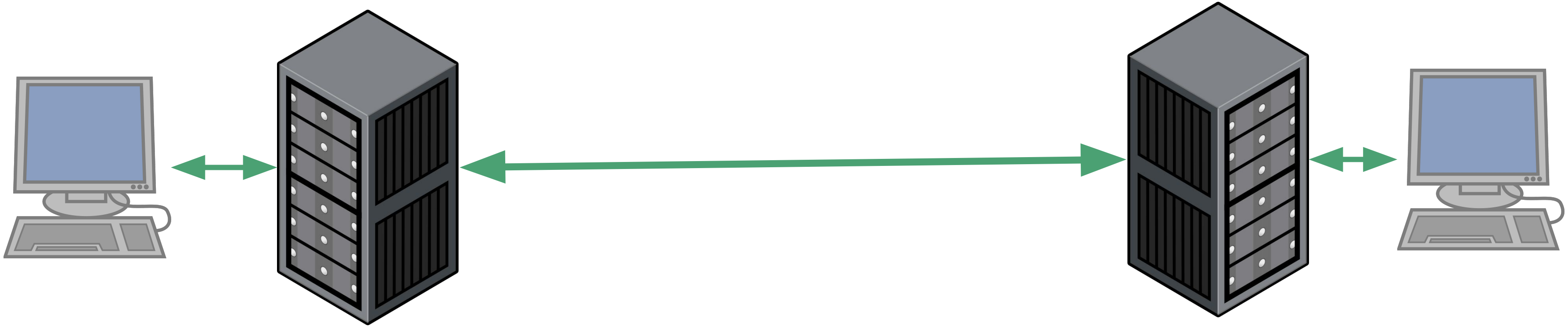
1. **The Problem**
2. Clarinet
3. Optimizing WAN Queries
4. Results

Low Application Latency Requires Localized Servers

Servers must be close to clients for latency.

Wide Area Networks (WANs) are necessary.

Collecting data into a central datastore for analytics is costly and slow.



Geode Focused On Execution

Previous work focused on *executing* queries smartly.

- Caching / Sending Deltas
- Choosing efficient distributed join algorithms
- Minimizing bandwidth rather than optimizing performance
- Allowing servers to adjust their sub-query execution plans

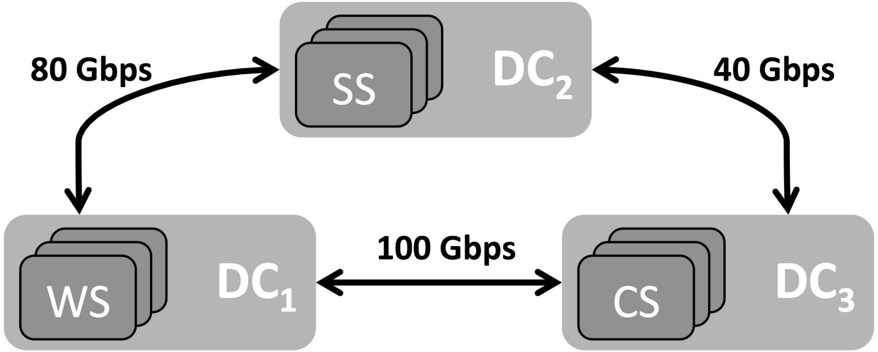
Wide Area Networks Are Heterogeneous

Sites may have different data available.

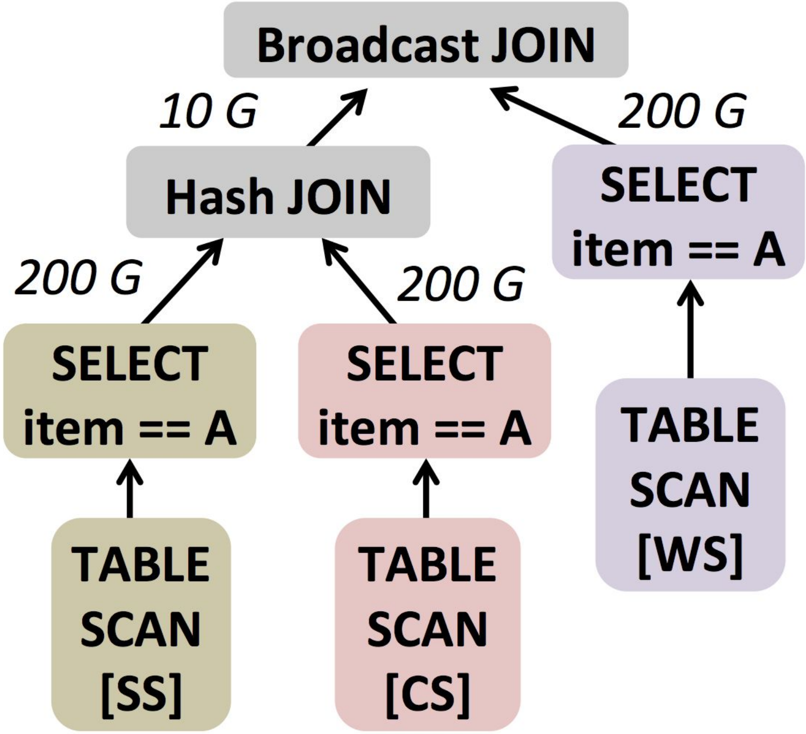
Links vary by 20x in latency.

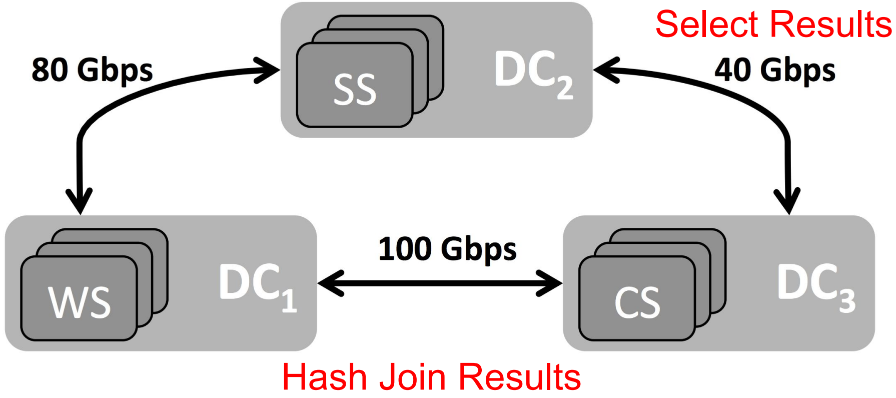
Link properties are relatively constant.

Bandwidth is finite.



Example Query Planned Sub-optimally

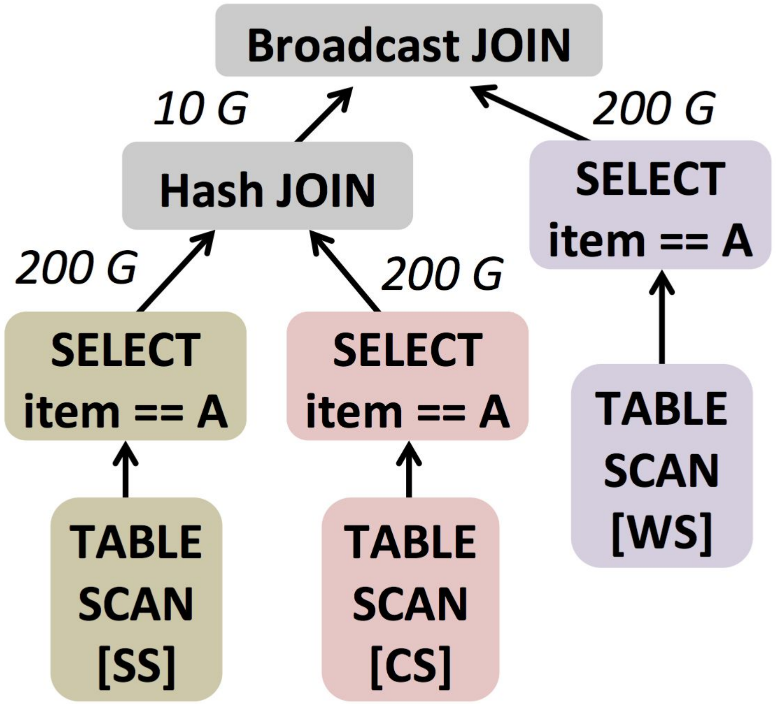




Central Planning Is Necessary

Execution plans limit flexibility during execution.

Need to consider the network before the execution plan.



Agenda

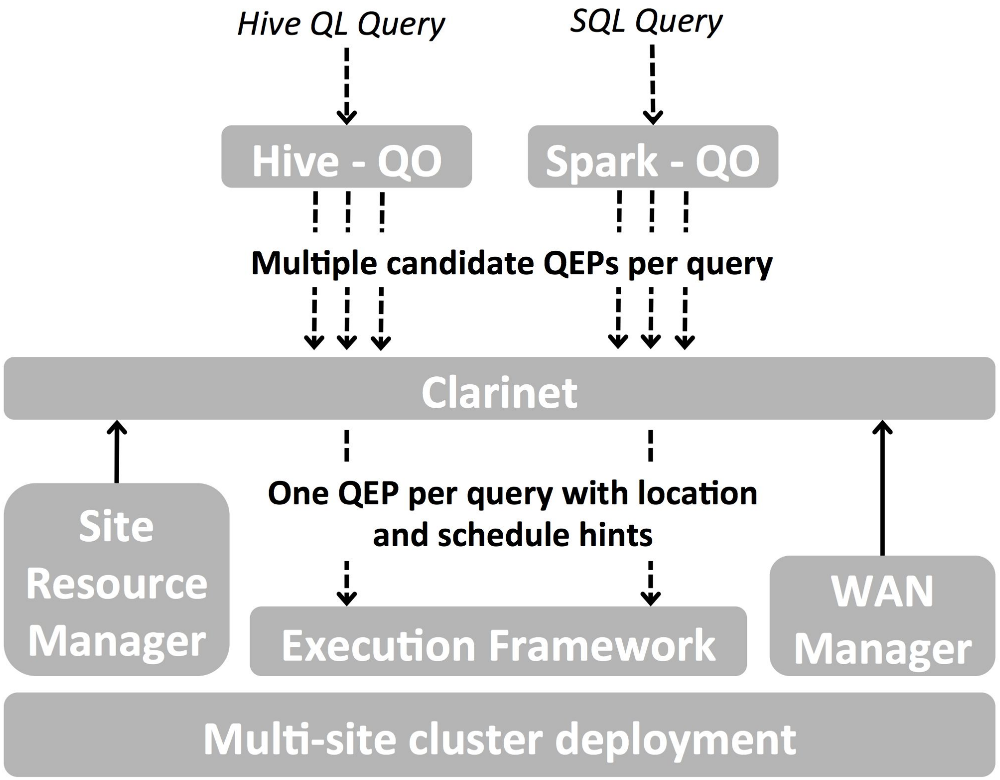
1. The Problem
2. **Clarinet**
3. Optimizing WAN Queries
4. Results

Clarinet Focuses on Planning

Clarinet adds network considerations into *logical query plan optimization*.

- Allows global optimization across queries.
- Introduces optimizations not possible at execution stage.
- Optimize execution time rather than resource usage.

Combining Optimization and Scheduling



Agenda

1. The Problem
2. Clarinet
3. **Optimizing WAN Queries**
4. Results

Optimizing WAN Queries Is Hard

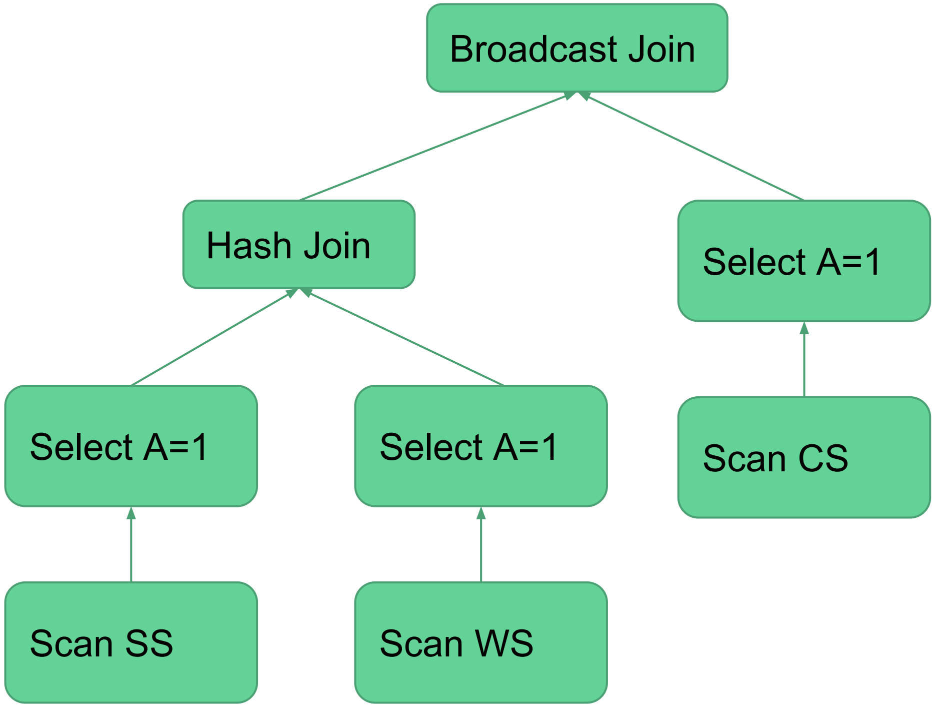
There are too many options to optimize in absolute terms

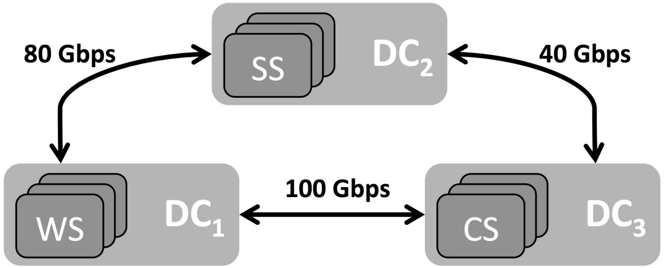
- Breaking queries into sub-queries
- *Where* each subquery will be run
- *How* each subquery will be run
- Network properties are a *shared resource* across all queries

Heuristic Optimization Algorithm

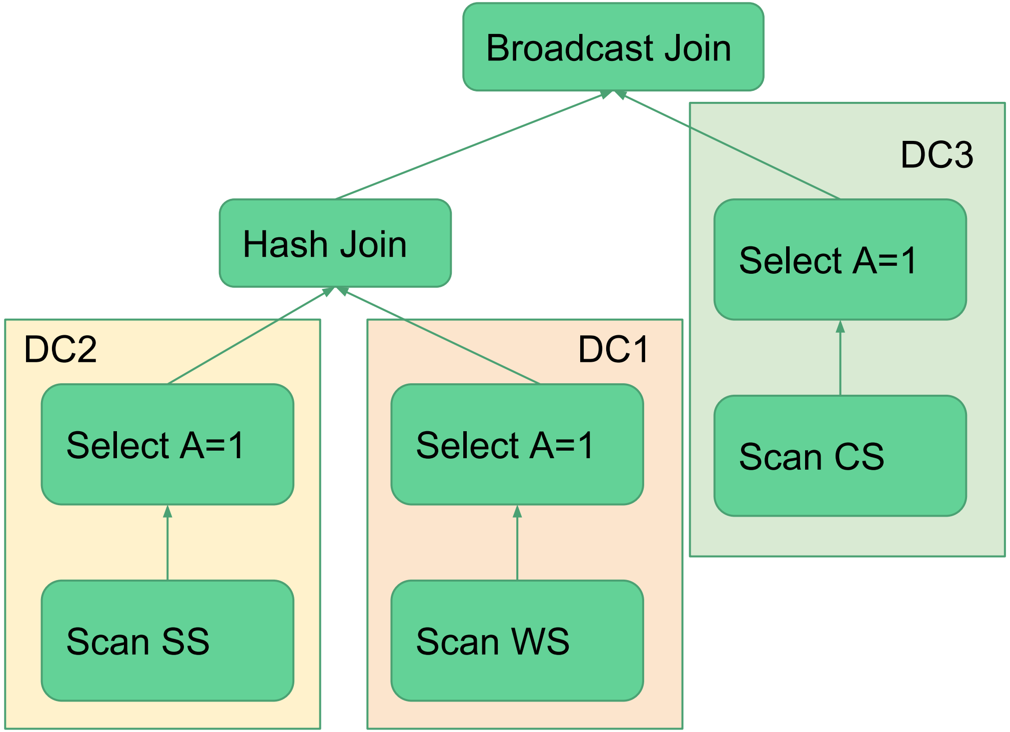
1. Assign **where** tasks run first:
   a. Place tasks with no dependencies (Mappers) where the data is.
   b. Just optimize where dependant tasks (Reducers) run based on network capacity.
      i. Also consider just putting all reducers on the node with the most mappers.
2. Estimate **how long** each DAG should take:
   a. Insert “shuffle” nodes into the DAG whenever data is moved over the network.
      i. Network properties
      ii. Currently running tasks
   b. Calculate the total length the DAG will take using a LP.

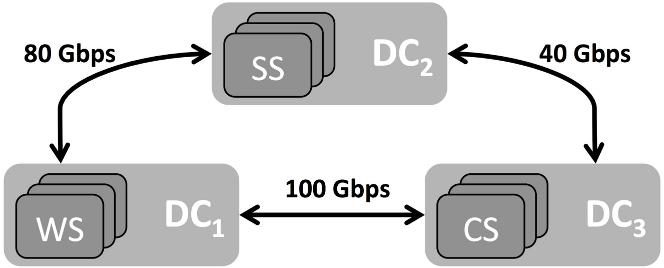
Example Query Planning



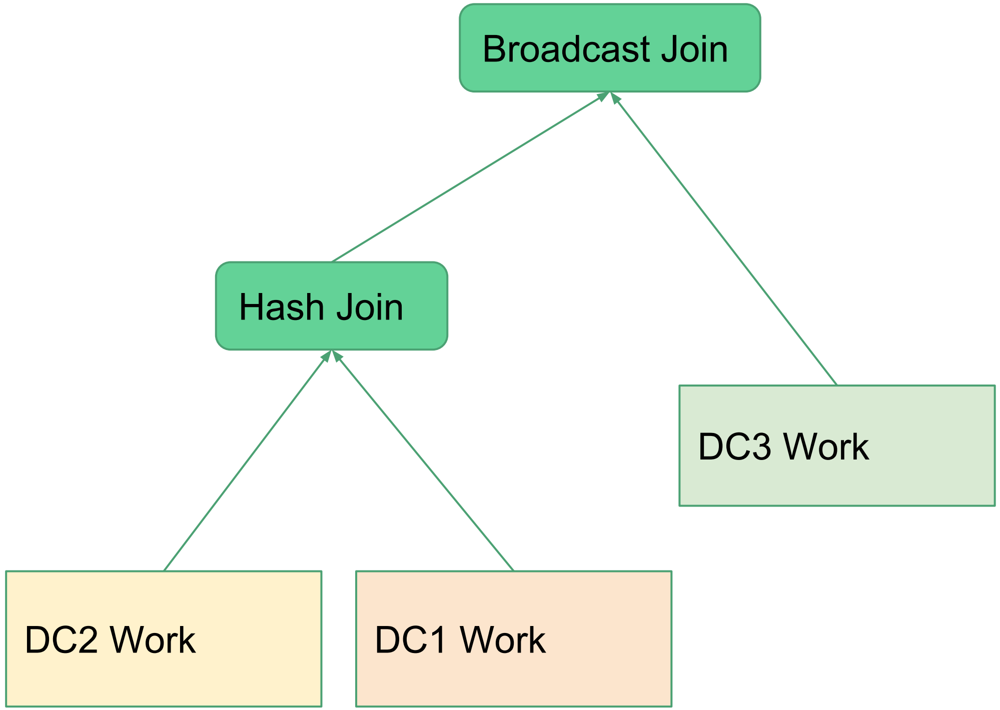


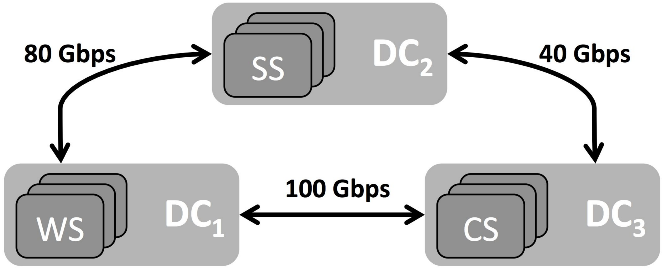
Assign Mappers





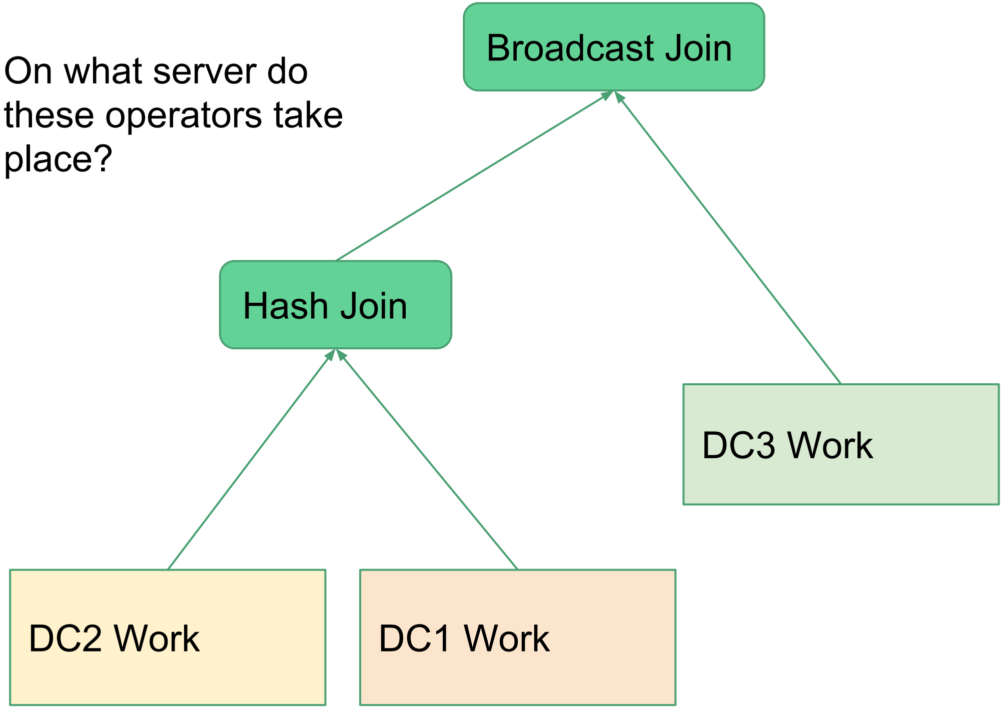
Compress Compute Operators

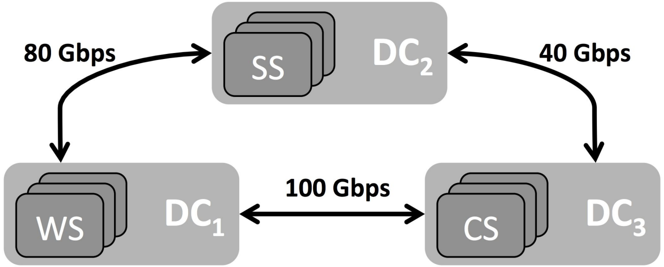




Compress Compute Operators

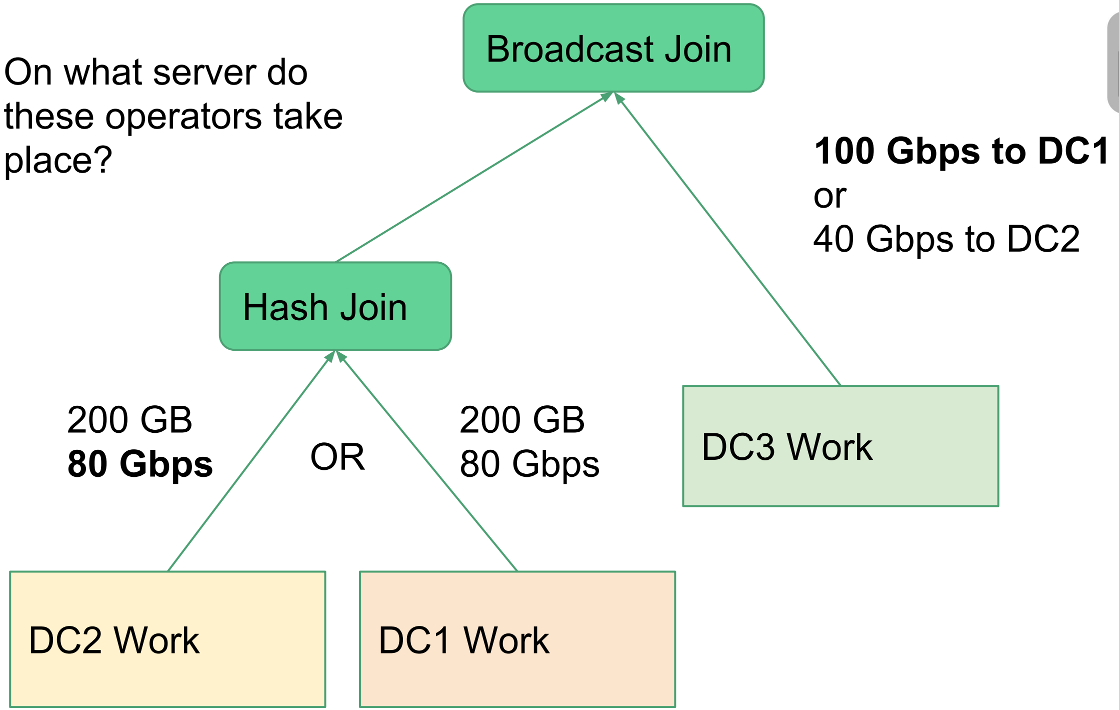
On what server do these operators take place?

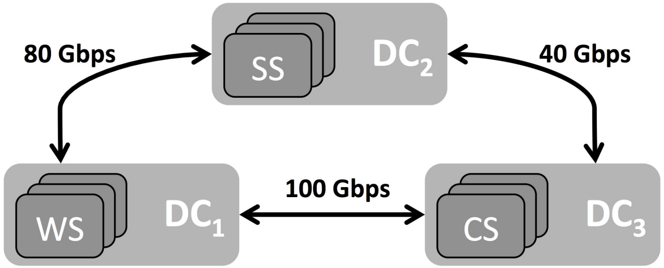




Compress Compute Operators

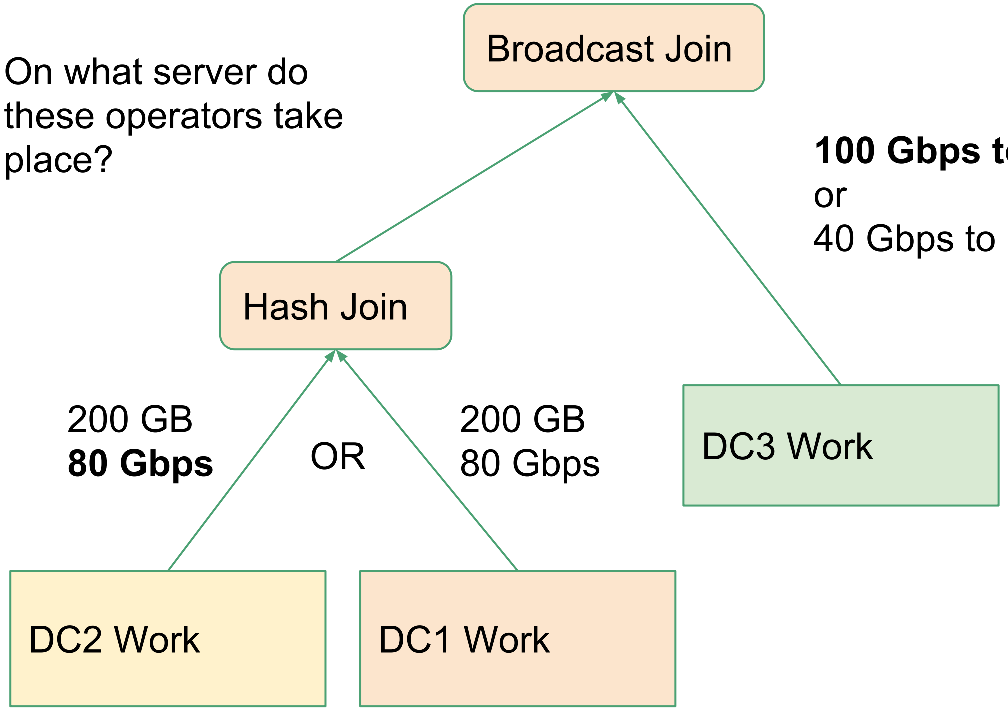
On what server do these operators take place?



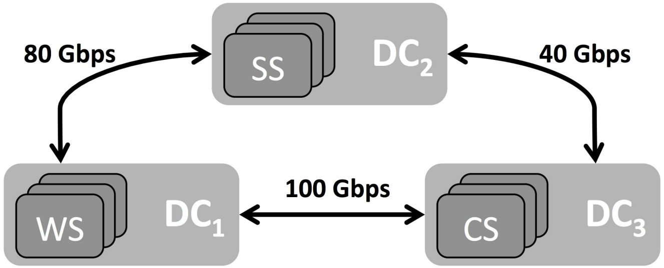


Compress Compute Operators

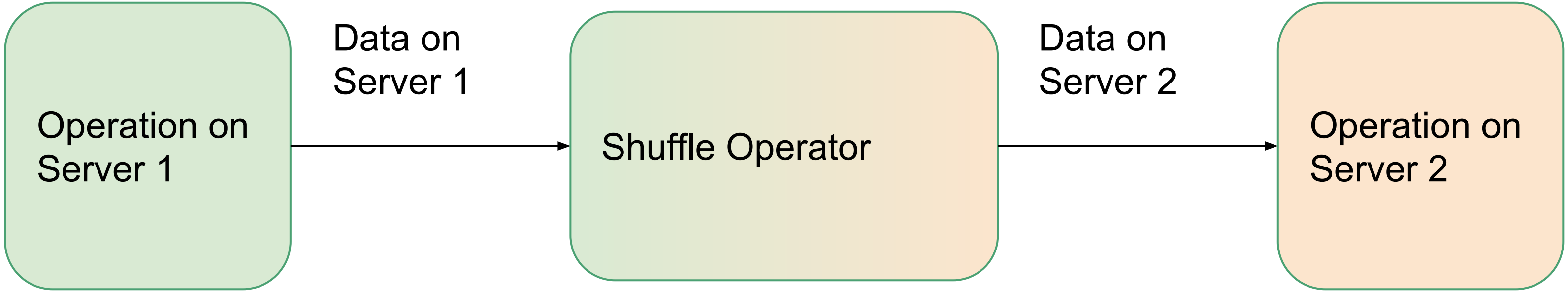
On what server do these operators take place?



100 Gbps to DC1
or
40 Gbps to DC2

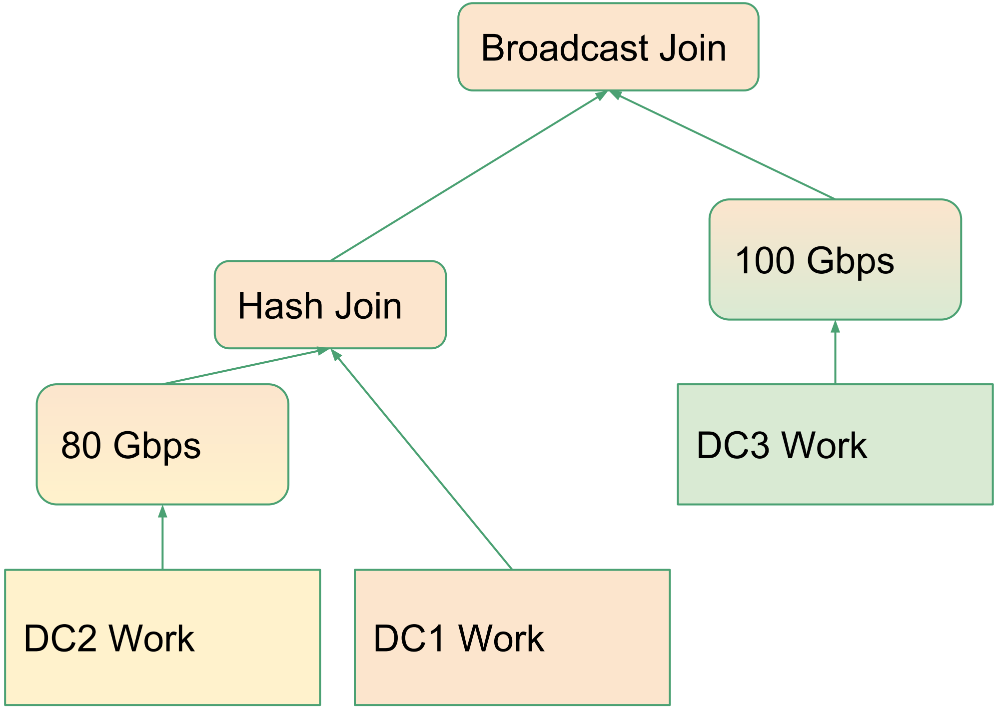


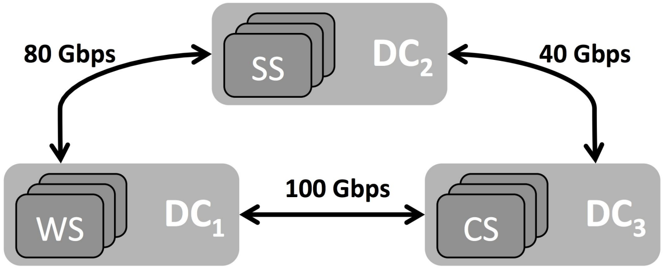
Shuffle Operators



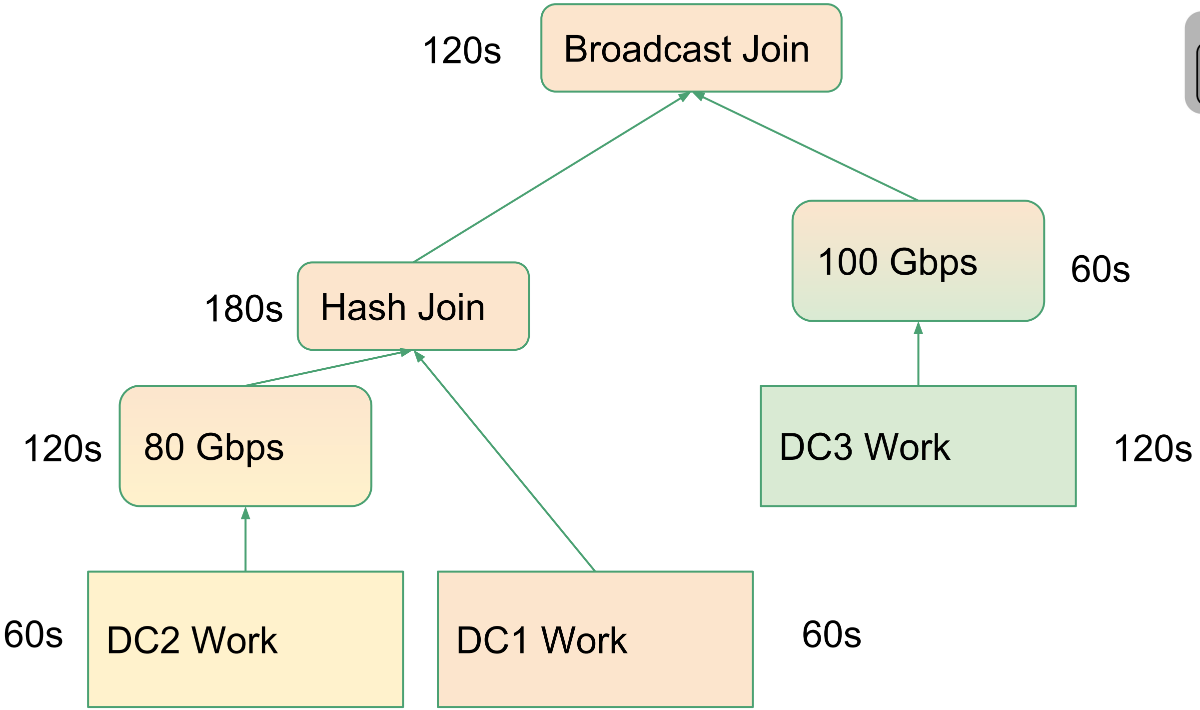
This operation's cost can be estimated from the volume of data and network bandwidth.

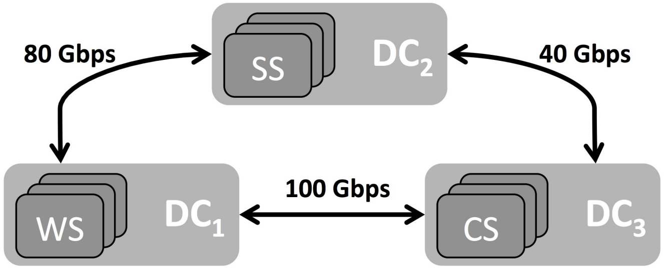
Introduce “Shuffle” Operators





Compute Cost Estimate





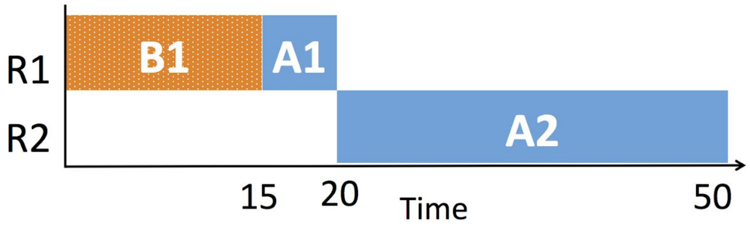
Dynamically Scheduling Resources

Allow scheduling tasks from any of the next *k* queries if resources available.

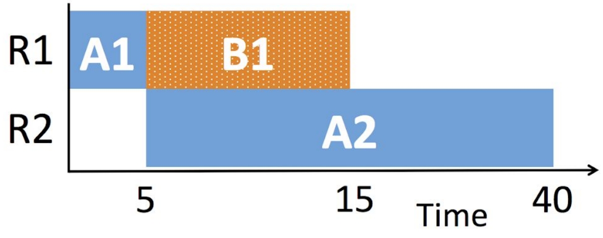
Efficiently uses available resources.

k must be tuned to avoid over-scheduling tasks with no dependencies.

Queries selected based on relative deadline proximity.



(a) SJF schedule

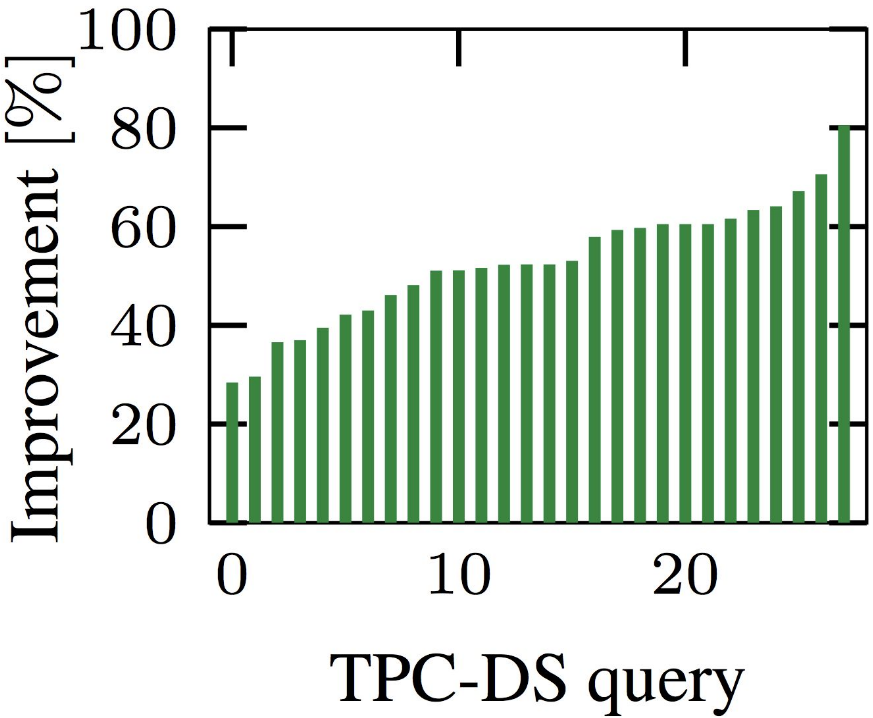


(b) Better Schedule

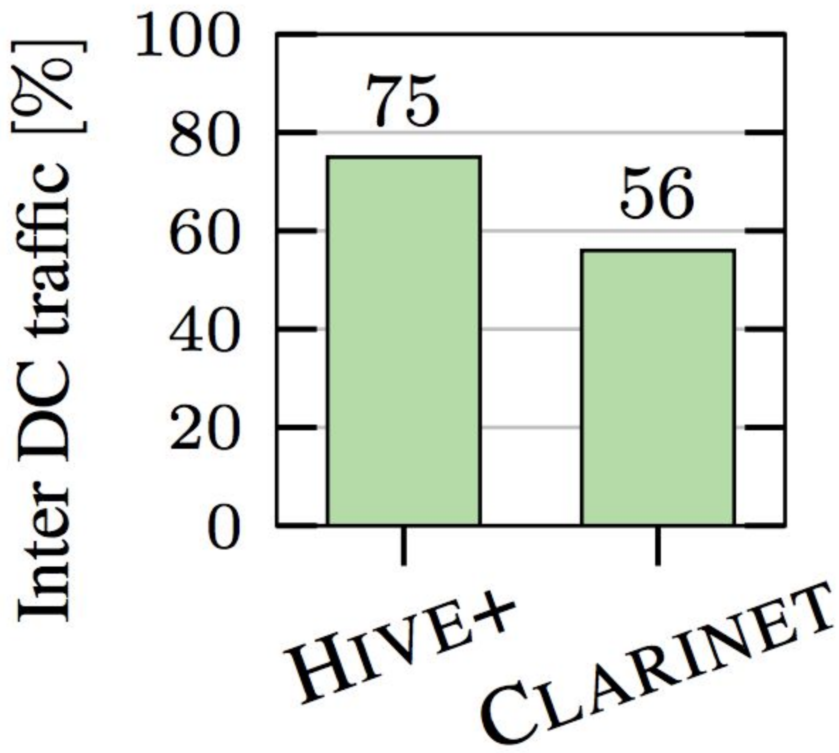
Agenda

1. The Problem
2. Clarinet
3. Optimizing WAN Queries
4. **Results**

Running Time Improved



Network Usage Improved



Other Performance Features

Multi Query Optimization

60% of queries run in batches ended up with different plans.

Resource Fragmentation

Network links are fallow less than 3% of the time.

Optimization Time

Approximately 10 seconds

Questions?