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

**Broadcast JOIN**

- **Hash JOIN**
  - 10 G
  - 200 G

- **SELECT item == A**
  - 200 G

- **TABLE SCAN [SS]**
- **TABLE SCAN [WS]**

**Select Results**

- **SS**
- **DC2**
- **WS**
- **DC1**
- **CS**
- **DC3**

- 80 Gbps
- 40 Gbps
- 100 Gbps

Hash Join Results
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

Hive QL Query

Hive - QO

Multiple candidate QEPs per query

SQL Query

Spark - QO

Clarinet

One QEP per query with location and schedule hints

Site Resource Manager

Execution Framework

WAN Manager

Multi-site cluster deployment
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

- **Hash Join**
  - **Select A=1**
    - **Scan SS**
  - **Select A=1**
    - **Scan WS**

- **Broadcast Join**
  - **Select A=1**
    - **Scan CS**

Network Diagram:
- DC_1 is connected to SS and WS with 100 Gbps.
- DC_2 is connected to SS and DC_1 with 80 Gbps.
- DC_3 is connected to CS and DC_1 with 40 Gbps.
Assign Mappers

- **Broadcast Join**
  - **Hash Join**
    - **Select A=1**
      - **Scan SS**
    - **Select A=1**
      - **Scan WS**
  - **Select A=1**
    - **Scan CS**

Network Connections:
- 80 Gbps: SS to DC2
- 40 Gbps: DC2 to DC3
- 100 Gbps: DC1 to DC3

DC2:
- **Select A=1**
  - **Scan SS**

DC1:
- **Select A=1**
  - **Scan WS**

DC3:
- **Select A=1**
  - **Scan CS**
Compress Compute Operators

- **Broadcast Join**
  - **Hash Join**
  - **DC3 Work**
    - **DC2 Work**
    - **DC1 Work**

Diagram:
- SS -> DC2 (80 Gbps)
- WS -> DC1 (100 Gbps)
- DC1 -> CS (40 Gbps)
- DC3 -> SS (40 Gbps)

Compress Compute Operators

On what server do these operators take place?

- Hash Join
- Broadcast Join
- DC3 Work
- DC1 Work
- DC2 Work
On what server do these operators take place?

- Hash Join
- Broadcast Join

200 GB
80 Gbps
OR
200 GB
80 Gbps

100 Gbps to DC1
or
40 Gbps to DC2

DC3 Work

DC2 Work
DC1 Work
Compress Compute Operators

On what server do these operators take place?

- **Broadcast Join**
  - **Hash Join**
    - **OR**
      - **200 GB 80 Gbps**
      - **200 GB 80 Gbps**

  - **DC3 Work**
    - **OR**
      - **200 GB 80 Gbps**

  - **100 Gbps to DC1**
  - **or**
  - **40 Gbps to DC2**

- **80 Gbps**
- **40 Gbps**
- **100 Gbps**

- **DC2 Work**
- **DC1 Work**
- **DC3 Work**
Shuffle Operators

This operation's cost can be estimated from the volume of data and network bandwidth.
Introduce “Shuffle” Operators

- Broadcast Join
  - Hash Join
    - 80 Gbps
      - DC2 Work
    - 100 Gbps
      - DC1 Work
      - DC3 Work

- 100 Gbps
- 80 Gbps
- 40 Gbps
- 100 Gbps
Compute Cost Estimate

- **Hash Join**
  - 180s
  - 80 Gbps
  - DC2 Work
  - 120s
  - DC1 Work
  - 60s

- **Broadcast Join**
  - 120s
  - 100 Gbps
  - DC3 Work
  - 120s

- Connections:
  - 80 Gbps from DC2 to DC2
  - 100 Gbps from DC1 to DC3
  - 40 Gbps from SS to DC2
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

<table>
<thead>
<tr>
<th>Inter DC traffic [%]</th>
<th>HIVE+</th>
<th>CLARINET</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>75</td>
<td>56</td>
</tr>
</tbody>
</table>
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?