Network Aware Operator Placement for Stream Processing Systems

Proposed by
Pietzuch, P., Ledlie, J., Shneidman, J., Roussopoulos, M., Welsh, M., Seltzer, M
Harvard University

Presented by
Mohammed Shaik Hussain Ali
University of Pennsylvania

Current Situation
Stream Processing Systems (DSPS) push query operators to nodes within the network

Examples of such a system: Aurora, Borealis

Resulting Challenges
Placement of operators must result in good query performance - low delay

Use the Network efficiently - reduce global impact

Reuse existing operators

Optimization Technique

Cost Space
- Multidimensional Metric
- Scalable
- Decentralized
- Adaptive

Working
- Every node has a Cost Space
- Distance between two nodes is the overhead of moving data
- Determine the placement by Spring Relaxation Algorithm
- Then map it back to Physical Network

Algorithm
- Optimize Network usage of the query
- Keep the query delay low
- Find nodes with adequate bandwidth
- Dynamic evaluation of operator placement and reuse of operators

A Descriptive Example
Hurricane Warning System

A collection of large set of sensors spread out over the entire region

Network of nodes
Information about changes in the environment is collected

Producers
Information passed onto the systems working for indications

Consumers
Systems that perform the necessary operations over specific regions

Operators

Stream Based Overlay Network

Improves network utilization
Provides low stream latency
Enables dynamic optimization
Manages operator placement
Increases performance
Built over the existing infrastructure

Analytical Modeling
Network Usage

\[ u(q) = \text{SUM}( \text{DR}(L) \times \text{Lat}(L) ) \]

- \( u(q) \) = Network usage for query \( q \)
- \( \text{DR}(L) \) = Data rates over link \( L \)
- \( \text{Lat}(L) \) = Latency over link \( L \)

\[ U = \text{SUM}( u(q) ) \]
SBON can be applied to any Data Stream Processing System.

SBON interacts with DPS in:
- Establishing and destroying
- Connecting/disconnecting I/O links of a node
- Migration of operators

Important takeaway: Use of Relaxation Algorithms

Features:
- Necessary
- Scalable
- Independent of global information
- Low communication overhead
- Adaptive
- Low latency
- Proper network usage
- To changing latency
- To changing data routes
- To changing load

SBON achieves efficient, decentralised in network operator placement and optimization through:

Cost Space
Relaxation Placement Algorithm

- Optimal cost for running data between nodes
- Operates in a distributed fashion
- The Relaxation iterative in the cost of running data
- Efficiency (number of nodes) vs. Space (nodes)
- Operator high/low in latency/immobile of Virtual Cost Space
- Operator in move to the closest physical SBON node in the Virtual Cost Space

Advantages:
- Adapt to changing network and query conditions
- Decentralised and no coordination is required
- Support cross-query optimisation

Full Placement Optimizer
- DPS passes query plan to SBON
- PFO finds initial placement using relaxation
- PFO runs once and has full information of query plan
- When physical nodes are identified, SBON informs DPS
- DPS creates stream and instantiates operators
**Placement Optimization**

*Local Placement Optimizer*
- Runs periodically
- Iterates over all operators & modifies latency space
- Maps virtual nodes to physical nodes ONLY when new is better
- Calculates saving in network usage
- Migration occurs ONLY if its more than migration threshold

**Evaluation**

Minimize the network usage and cause low delay

Optimize the placement of operators by migration and reusing

Link SBON with DPS

**Management**

- Before migration, DSFS is notified to stop data streaming
- With no streaming, other optimizers stop optimizing
- After migration is accomplished, streaming is restarted
- Inherent advantage is no data loss in migration

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Network Usage Penalty</th>
<th>Delay Penalty</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIMAL</td>
<td>0%</td>
<td>13%</td>
</tr>
<tr>
<td>RELAXATION</td>
<td>15%</td>
<td>24%</td>
</tr>
<tr>
<td>IP MULTICAST</td>
<td>27%</td>
<td>0%</td>
</tr>
<tr>
<td>PRODUCER</td>
<td>43%</td>
<td>75%</td>
</tr>
<tr>
<td>CONSUMER</td>
<td>60%</td>
<td>0%</td>
</tr>
<tr>
<td>RANDOM</td>
<td>81%</td>
<td>76%</td>
</tr>
</tbody>
</table>

**Algorithm Notes**
- **OPTIMAL**: Best possible placement (not feasible)
- **RELAXATION**: Global knowledge
- **IP MULTICAST**: Nodes are either off-chip or for a producer
- **PRODUCER**: Any producer is randomly selected
- **CONSUMER**: Operator placed at consumer nodes
- **RANDOM**: Any randomly picked node

![Graph 1](image1.png)

![Graph 2](image2.png)
Summary

SBON
- Is compatible with existing infrastructure
- Uses relaxation algorithm to arrive at an optimum placement technique
- Uses cost space metric to consider all aspects of network
- Is scalable and decentralized
- Achieves network usage and low delay

Steps of Implementation
- For every new user query, Birelle creates a query plan
- Query plan is sent to SBON
- Full query optimizer gives an initial starting node
- This initial starting node is sent back to Birelle
- SBON keeps track of network and node parameters
- SBON sends migration information to Birelle

Figure 13. Reuse Topology