An Optimizer for Distributed Intra-Node-Parallel Streaming Systems

Matthias J. Sax1, Malu Castellanos2, Qiming Chen2, Meichun Hsu2

Streaming and Parallel Dataflow Execution

- Data Intensive Computing.
- Requires low response time, up to (near) real-time analysis.
- MapReduce does not fit (batch system).
- New class of intra-node parallel streaming systems address this problem:
  e.g., Storm, S4, Muppet.

Input is a dataflow specified as directed acyclic graph (DAG):

Dataflow (called Topology in Storm) is executed in a parallel manner:

Batching in Streaming Systems

- Sending data tuple-by-tuple results in high network overhead.
- Tuple batching can increase throughput.

Key-based data distribution (w/o batching):

Novel batching schemas for intra-node parallelism:

Cost Model for Batch Size and Degree of Parallelism

Optimizing Batch Size:
• Batching reduces network overhead n
• n is shared over multiple tuples, each with payload s

Optimizing Degree of Parallelism (dop):
• Increasing dop reduces load on single node

Topo-logy Optimization Algorithm
1: P ← all source nodes
2: while P is not empty do
3:   for all p ∈ P do
4:     if input latency is smaller than act. proc. time then
5:       b ← calc batch size to reduce ship. time
6:     if b > B_{max} then b = B_{max}
7:      increase dop of p to increase l_i
8:   end if
9: end if
10: if calculate output rate r_o
11:   end for
12: C ← all unprocessed nodes with known input rate r_i
13: for all c ∈ C do
14:   dop_c ← calculate dop such that l_i > ypt
15: end for
16: P ← all c ∈ C with outgoing edges
17: end while

Evaluation

Processing Times for Fixed-Size Data Sets

Tuple Latency for Selected Bolts