Controlling Memory Footprint of Stateful Streaming Graph Processing

Pourya Vaziri

Keval Vora

Simon Fraser University

USENIX ATC 2021
July 14, 2021
Graph Analytics
Streaming Graph Processing

- Graph structure keeps on changing via a stream of edge/vertex updates
Streaming Graph Processing

• Graph structure keeps on changing via a stream of edge/vertex updates

• **Incremental Processing**
  • Adjust results incrementally
  • Reuse work that has already been done
Stateful Iterative Processing Model

• Track intermediate state as iterations progress

Supersteps (a.k.a Iterations)

Initial State → Intermediate State → Intermediate State → Intermediate State → Final State
Streaming Graph Systems: GraphBolt & DZiG

• Dependency-Driven Incremental Refinement
• Generates same results at every iteration as a BSP execution from scratch
• Lightweight dependence tracking at vertex-level
• Dependency-aware value refinement upon graph mutation

Memory-Efficient Stateful Iterative Models

• Limit memory footprint of stateful incremental processing
• Retain performance benefits of incremental processing

• Selective Stateful Iterative Model
• Minimal Stateful Iterative Model
Selective Stateful Iterative Model

- Vertex computations are independent of each other (regardless of whether intermediate value is tracked or not)
- Usefulness of tracked intermediate values is different across vertices
Selective Stateful Iterative Model: Challenges

Which intermediate state should be tracked, and which skipped?

How to perform incremental computation with missing state?
Selectively Tracking Intermediate State

• How much intermediate state to track?

$$\arg\min_k |\text{mem\_budget} - (k \times \text{state\_size} \times t + \text{base\_size})|$$

• What intermediate state should be tracked?
  • Top-k highest in-degree vertices
Selective Incremental Processing

\[
\text{diff}(a_{\text{new}}, a_{\text{old}}) \quad \text{diff}(b_{\text{new}}, b_{\text{old}})
\]

Untracked Vertex

Tracked Vertex

Propagate Old Value & New Value

Propagate Difference in Values
Minimal Stateful Iterative Model

- Aggressively reduce the memory footprint
- Specialize incremental processing for purely distributive operations.

- Distributive Update Property
Distributive Update Property

- Computation distributed into sub-computations on subsets of inputs
- Change in a vertex's output value can be **directly computed using only incoming changes** it gets, **without tracking intermediate states**

\[
\text{Val}(a) + \text{change}_a = \bigoplus (\text{S(change}_a), \text{S(change}_b))
\]

\[
\text{Val}(b) + \text{change}_b = \gamma(x)
\]

**PageRank**

**Katz Centrality**

Co-Training Expectation Maximization

Path-based / Monotonic Algorithms:

- Breadth First Search
- Shortest Paths
- Connected Components
Minimal Stateful Iterative Model

• Aggressively prune out intermediate state tracking
• Track values only for the earliest points affected by mutation

• Earliest mutation points can be identified on application-basis
  • What-if analysis around an event or bug/leak location
  • Replaying historical mutation streams

• Incremental refinement: directly propagate changes
Other Details

• Dynamically maintain **useful intermediate state** as graph changes
  • Incrementally maintain top-k highest in-degree vertices

• **Eliminate expensive branches** from critical path

• **Optimized graph layout** for direct parallel operations of different types

• Dynamically **switching** between minimal incremental processing and selective incremental processing
Experimental Setup

• Graph algorithms
  • PageRank (PR), Co-Training Expectation Maximization (CoEM), Collaborative Filtering (CF), Label Propagation (LP), Circuit Simulation (CS), Multi-Manifold Ranking (MMR), Multi-Modality Learning (MML)

• 24-core (48 threads) – 320 GB

<table>
<thead>
<tr>
<th>Graphs</th>
<th>Vertices</th>
<th>Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>TwitterMPI(TT)</td>
<td>52.6M</td>
<td>2.2B</td>
</tr>
<tr>
<td>Friendster(FT)</td>
<td>68.3M</td>
<td>2.5B</td>
</tr>
<tr>
<td>UK-2007-05(UK)</td>
<td>105M</td>
<td>3.7B</td>
</tr>
<tr>
<td>UK-union(UN)</td>
<td>133M</td>
<td>5.5B</td>
</tr>
<tr>
<td>Clueweb(cwb)</td>
<td>978.4M</td>
<td>42.5B</td>
</tr>
</tbody>
</table>
Selective Stateful Iterative Model

- Selective-20%
  - Up to 70% less memory than GraphBolt
  - Up to 80% performance gained by GraphBolt

- Selective model successfully delivers high performance for cases where GraphBolt runs out of memory
Selective Stateful Iterative Model

- Amount of computation get tuned based on the tracked intermediate state
Minimal Stateful Iterative Model

- Only ~30% more memory consumption compared to stateless execution
- Up to 8.2x faster than stateless execution
Other Experiments

CoEM on UK

LP on UK

CoEM on TT

Execution Time (sec)

Memory Footprint (GB)

Other Experiments
Conclusion

• Dependency-driven incremental processing important to maintain low latency and high performance
  • Enabled by systems like GraphBolt & DZiG
• Limiting memory footprint is crucial to scale on larger datasets
• Memory-Efficient Stateful Iterative Models
  • Selectively track intermediate values for only a subset of vertices
  • Distributive Update Property to aggressively reduce memory footprint
• Generic models that can be incorporated in other dynamic (graph) processing system