Paper Review - Controlling Memory Footprint of Stateful Streaming Graph Processing

Sai Venkat Malreddy

12 november 2024

1 Idea and Contribution

This paper addresses the high memory overhead problem in stateful streaming graph processing systems by introducing two memory-efficient models: (1) a Selective Stateful Iterative Model that controls memory by maintaining intermediate state for only selected high-impact vertices, and (2) a Minimal Stateful Iterative Model that further reduces memory footprint by exploiting the distributive update propertyöf certain graph algorithms. The key contributions include a technique for selecting high-impact vertices based on in-degree, an optimized graph layout and incremental processing strategy for handling tracked/untracked vertices, formalization of the distributive update property, and an implementation demonstrating significant memory reduction (35-70% less) while retaining 15-83% of performance benefits compared to full stateful systems. The solutions are general enough to be incorporated into existing streaming graph processing systems while providing clear memory-performance tradeoffs, allowing users to scale to larger graphs that previously couldn't be processed due to memory constraints.

2 Positive Comments

- The paper addresses a significant practical problem - memory overhead limiting scalability of streaming graph processing systems to large graphs
- The solutions are general and can be incorporated into existing systems while providing clear memory-performance tradeoffs
- Comprehensive evaluation across multiple real-world graphs and algorithms demonstrates practical benefits and scalability improvements

3 Negative Comments

- The selective model still requires maintaining both old and new values for untracked vertices, leading to some memory overhead compared to purely stateless execution
- The minimal stateful model is limited to algorithms satisfying the distributive update property, which excludes several important graph algorithms like Collaborative Filtering and Label Propagation
- The performance benefits of selective tracking decrease as the number of tracked vertices increases, suggesting diminishing returns for memory-performance tradeoffs at higher tracking percentages

4 Questions unanswered about the paper

- Dynamic Vertex Selection Strategy: How often should the set of tracked vertices be updated as the graph evolves? What are the overhead costs of maintaining and updating the degree-ordered max-heap? Are there better selection strategies beyond just using vertex degree?