Department of Computer Science

Flexograph: Storing and Analyzing Large Graphs

Puneet Mehrotra, Haotian Gong, Margo Seltzer

Problem

- Most graph processing systems(GPS):
  - Do expensive preprocessing steps
  - Are ETL heavy
- In-memory GPS need expensive clusters for large graphs
- Out-of-core GPS are slower
- Neither provide data management and redo preprocessing on each new snapshot of the graph
- Runtime performance depends on graph representation

Using Databases for Analytics

- No expensive ETL pipelines
  - Graph is a materialized view
- SQL is hard for iterative graph analytics
- Graph Databases are slow and do not scale

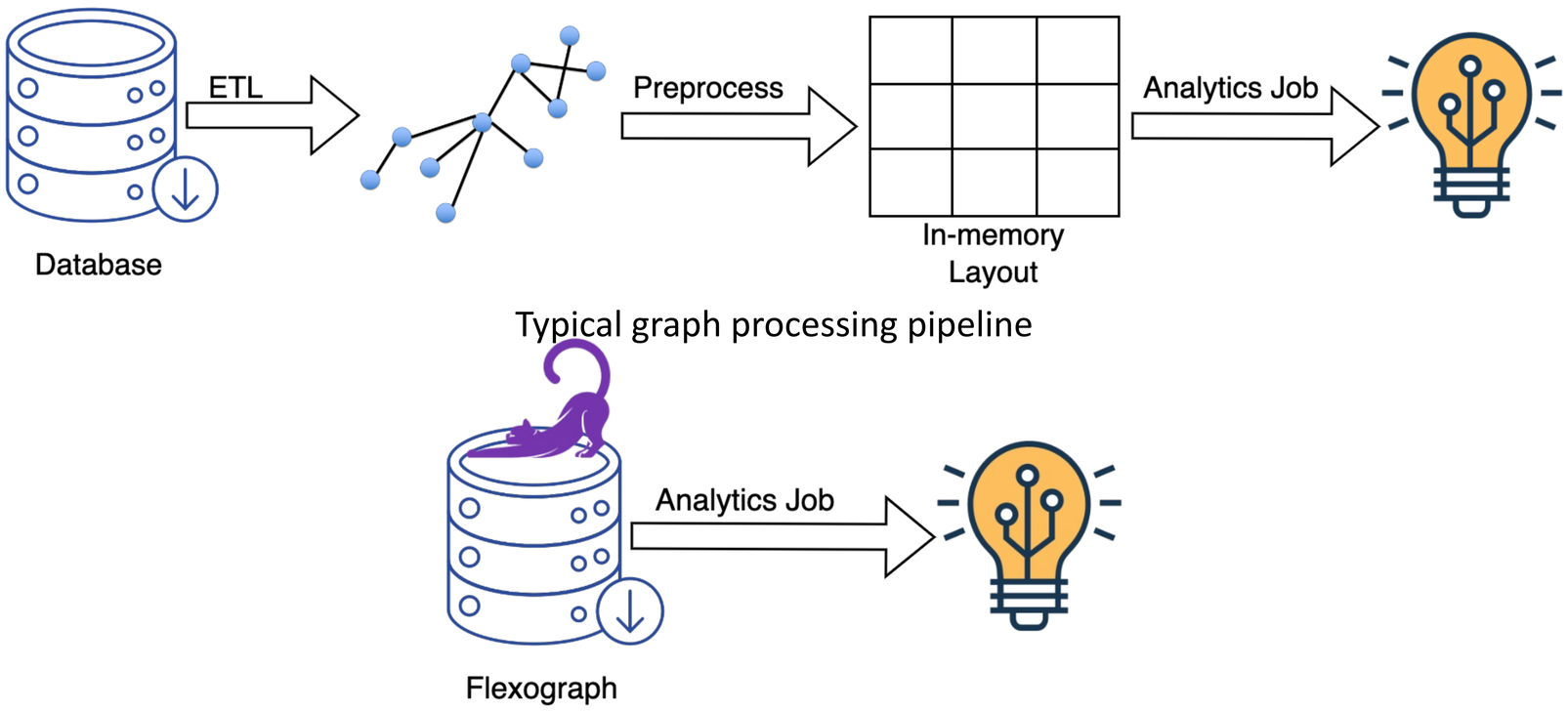
Graph Layouts

- Popular **in-memory** graph layouts:
  - **Compressed Sparse Rows:** Space efficient, immutable
  - **Adjacency List:** often as Linked Lists
  - **Edge Logs:** preserves temporal order

Representation	Insertion	Edge Scan
CSR	Immutable	Sequential
Adjacency List	*depends on implementation*	
Edge Log	O(1)	Sequential

- Representation **on disk** does not match the in-memory representation
  - Preprocessing is needed to gain sequential access

RQ: **Can we achieve the performance comparable to conventional GPS while persisting the data in a graph-friendly form?**



Typical graph processing pipeline

Flexograph : persistent data layout matches expected in-memory layout

Our Idea: Flexograph

A **Semi-External** graph processing and storage system that uses a mature KV store to persistent graph representations

- No need to extract or preprocess data
- No re-inventing storage engines for GPS
- API provides near-sequential access to neighborhood info
  - Express algorithm in edge- or vertex-centric fashion
- Semi-External model allows both synchronous and asynchronous algos
- Transient state is persisted for future use
- Prototype with 3 representations: RDBMS-like, CSR-like, Adjacency List

Preliminary Results

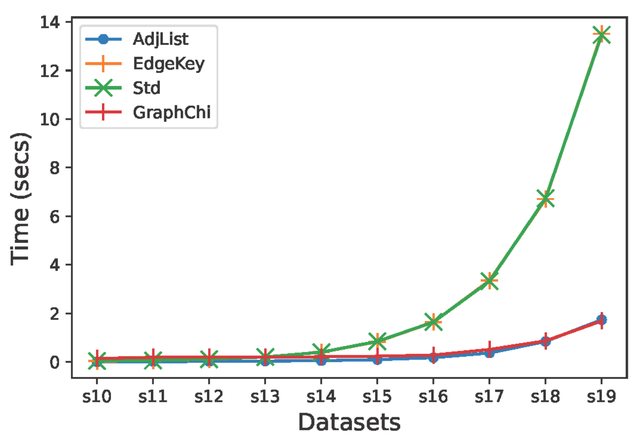
- We are faster than Neo4j and ArangoDB on standard graph benchmarks
- Flexograph is faster than GraphChi
- We can sustain a peak insertion rate of 1.9 million edges per second

Conclusions

- Graph Databases can be made fast for analytics without needing preprocessing

Next Steps

- Compare against other graph processing systems
- Evaluate effect of graph properties on runtime



THE UNIVERSITY OF BRITISH COLUMBIA

Systopia Lab
Researching Systems, construed Broadly

Partners
HUAWEI
NSERC CRSNG