Janus: Transactional Processing of Navigational and Analytical Graph Queries on Many-core Servers

Hideaki Kimura
(speaker)

Alkis Simitsis

Kevin Wilkinson

Hewlett Packard Labs

Take-away

- *Graph Engine on modern servers for both navigational and analytic queries.*

- *Leverages Transaction Processing.*

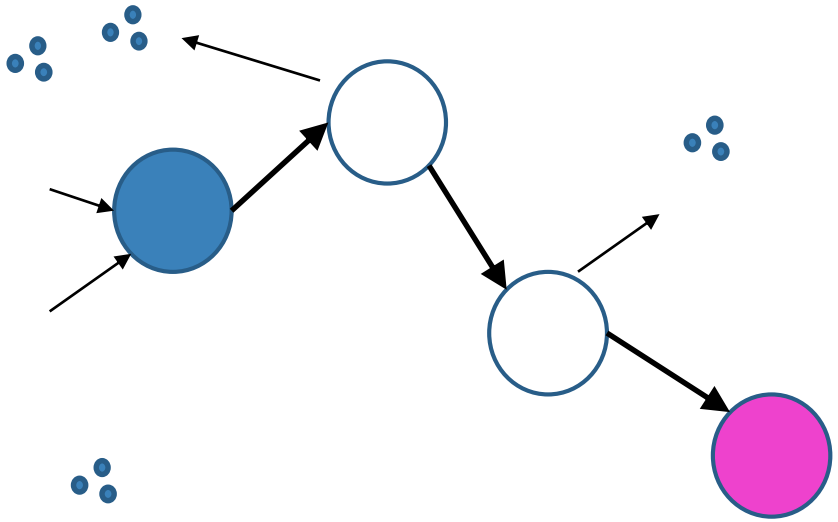
Navigational vs Analytic Graph Queries

Navigational

High-throughput

Accesses few vertices/edges

e.g., Pair-wise shortest path,
"Can he see my LinkedIn prof."

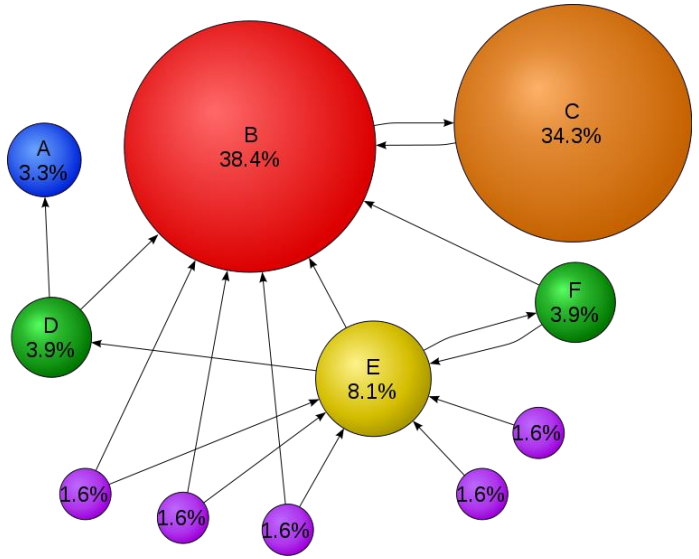


Analytic

Resource-intensive

Accesses a large fraction of graph

e.g., PageRank,
Graph clustering



Existing Graph Engines

- Optimized *either* for navigational (e.g., *Neo4j),* or for analytic queries (e.g., *GraphLab)*
- Limited scalability on *many-core*
- Poorly leverages *large, NUMA Memory*
- No fast, *concurrent updates*

Janus

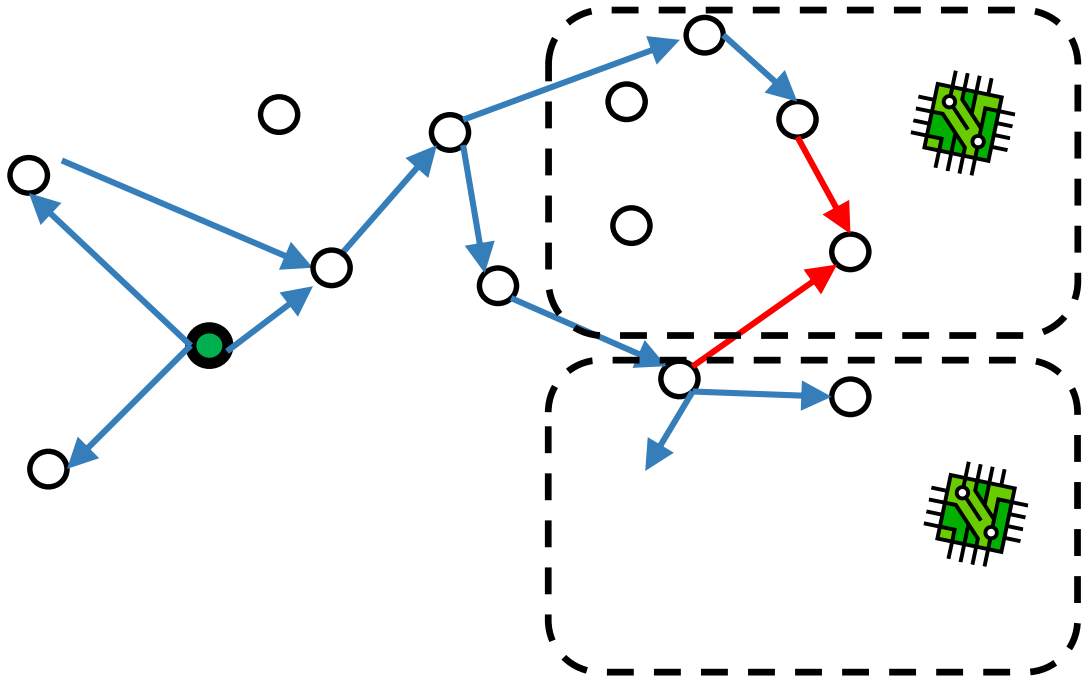
- Runs both type of queries as well as concurrent updates
- Exploits emerging server hardware; many-cores, large DRAM/NVM.
- Built on Transaction Processing engine (FOEDUS)

Reason 1 : Concurrent/serializable update. Obvious.

Reason 2 : *Scalability*. To parallelize a query.

Parallelizing a Graph Query as Transactions

Single-Source Shortest-Path (SSSP)

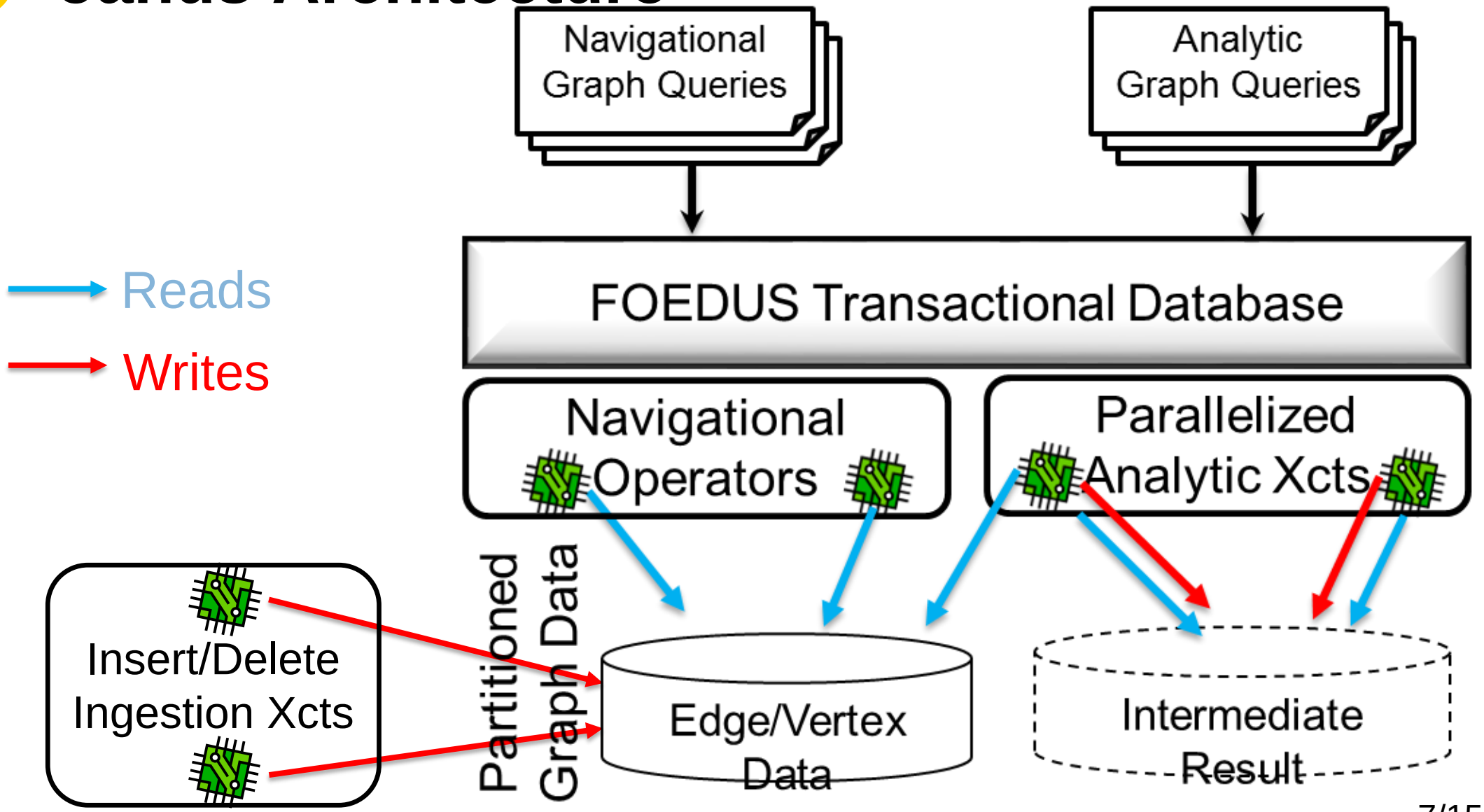


"Distributed GraphLab"
[Low et al, VLDB'12]

Parallel workers issue *millions of concurrent transactions*.

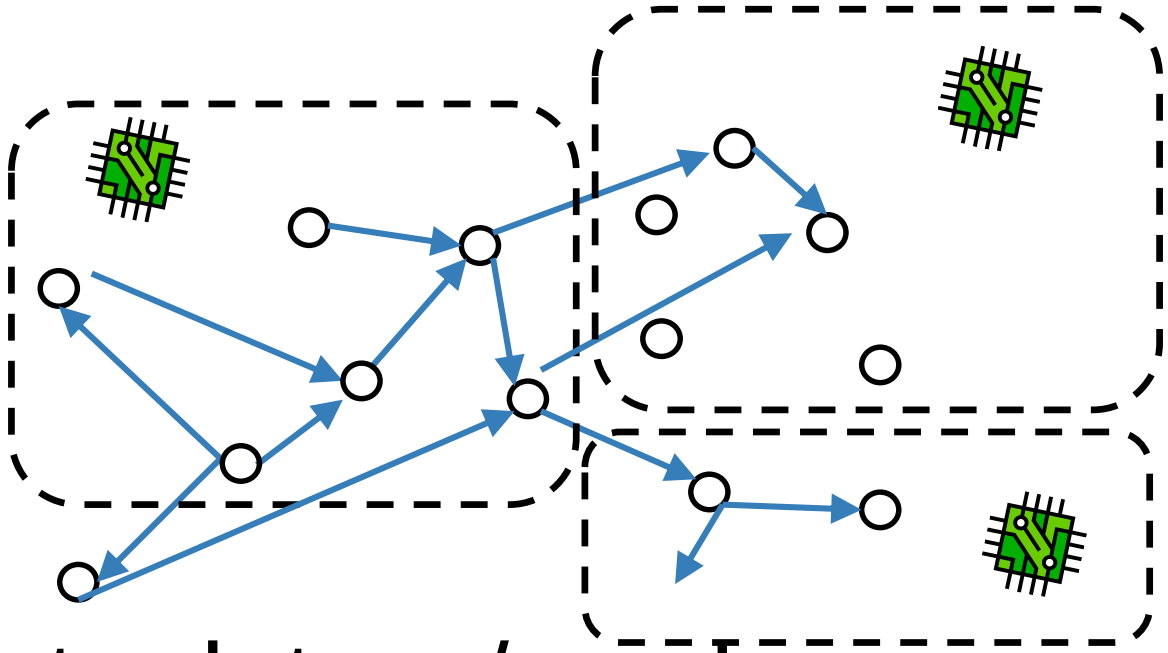
- Serializability is must; otherwise loop forever.
- Scalability is must; many-cores, large NUMA.

Janus Architecture



Partitioning Graph and Workers

- NUMA-aware partition for permanent graph, intermediate data, and workers.

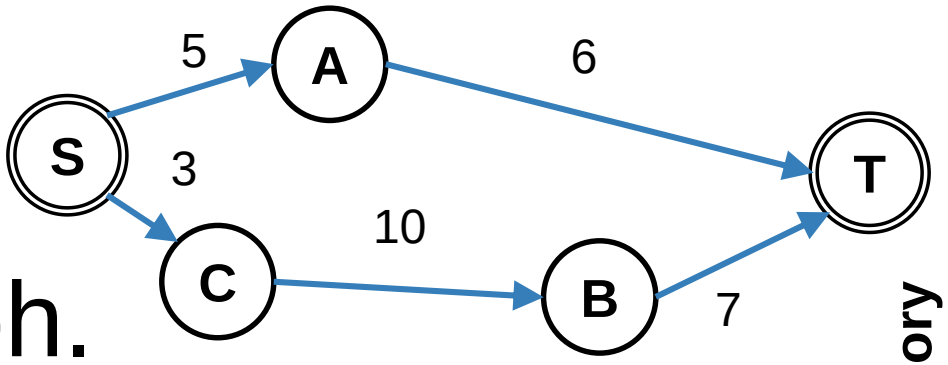


- Locality matters. Co-locate data w/ workers.
- Needs a database that supports flexible partitioning and data-worker co-location.

Pair-wise shortest-path Impl. in Janus

- Good-old Dijkstra.
- A NUMA-aware worker.
- Serializable Reads on Graph.

Graph Data



Global Memory

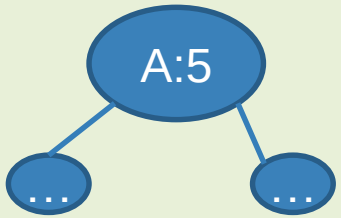
From	Edges
S	A:5, C:3..
A	T:6
…	…

Intermediate Data on worker-local memory

Distance hashtable

Node	Dist.
A	5
B	13
C	3

Relaxation min-heap



Navigational Worker

Serializable Reads

(Snapshot Reads from NVM as of same epoch)

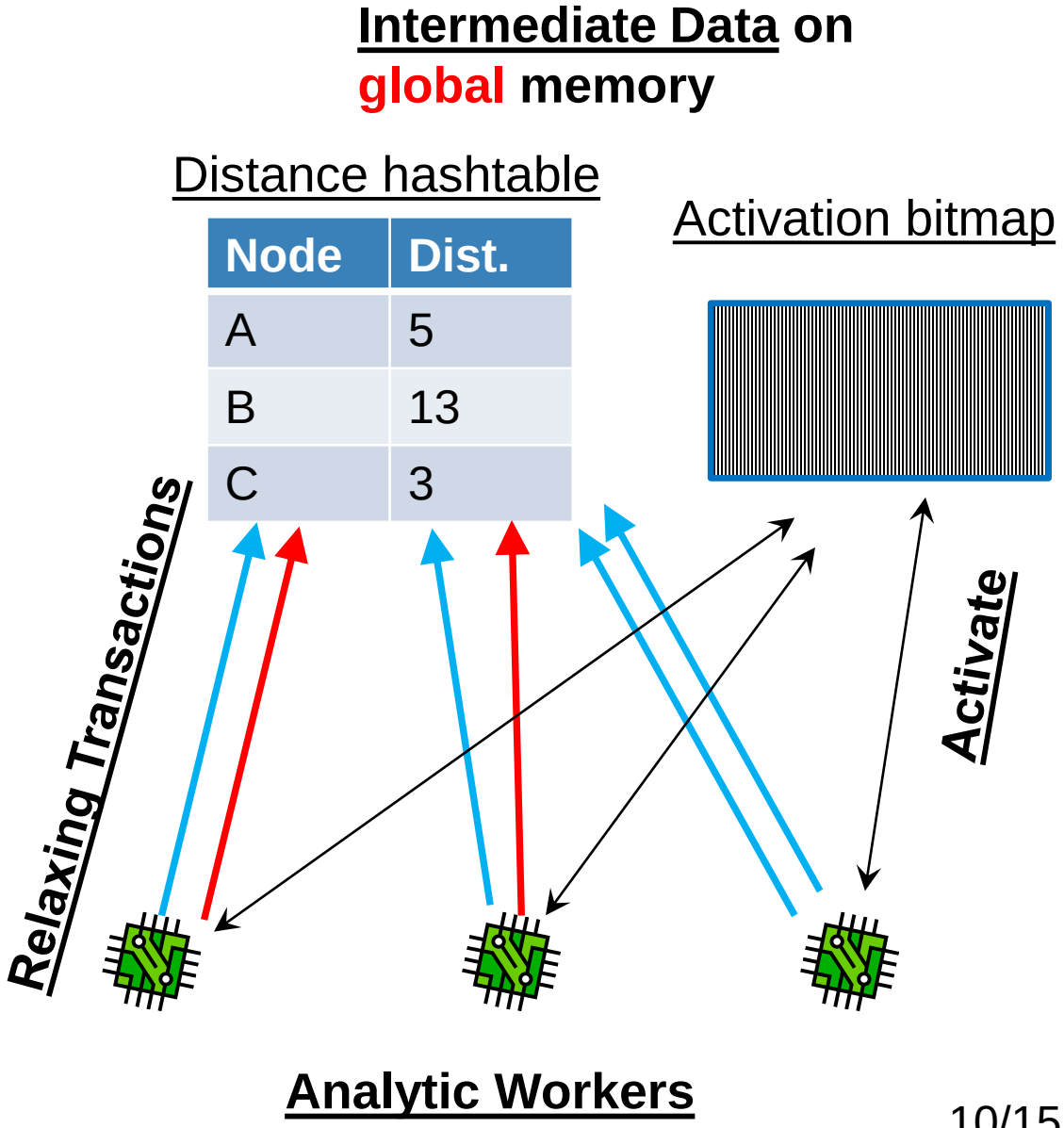
9/15

"FOEDUS", [SIGMOD'15]

SSSP Impl. In Janus

- Distributed Bellman-Ford
- Analytic-workers cooperatively maintain global memory.
- Processes billions of highly contended Xcts on Intermediate Data

"Mostly Optimistic Concurrency Control" [VLDB'17]



Experiments

- Shortest-Path

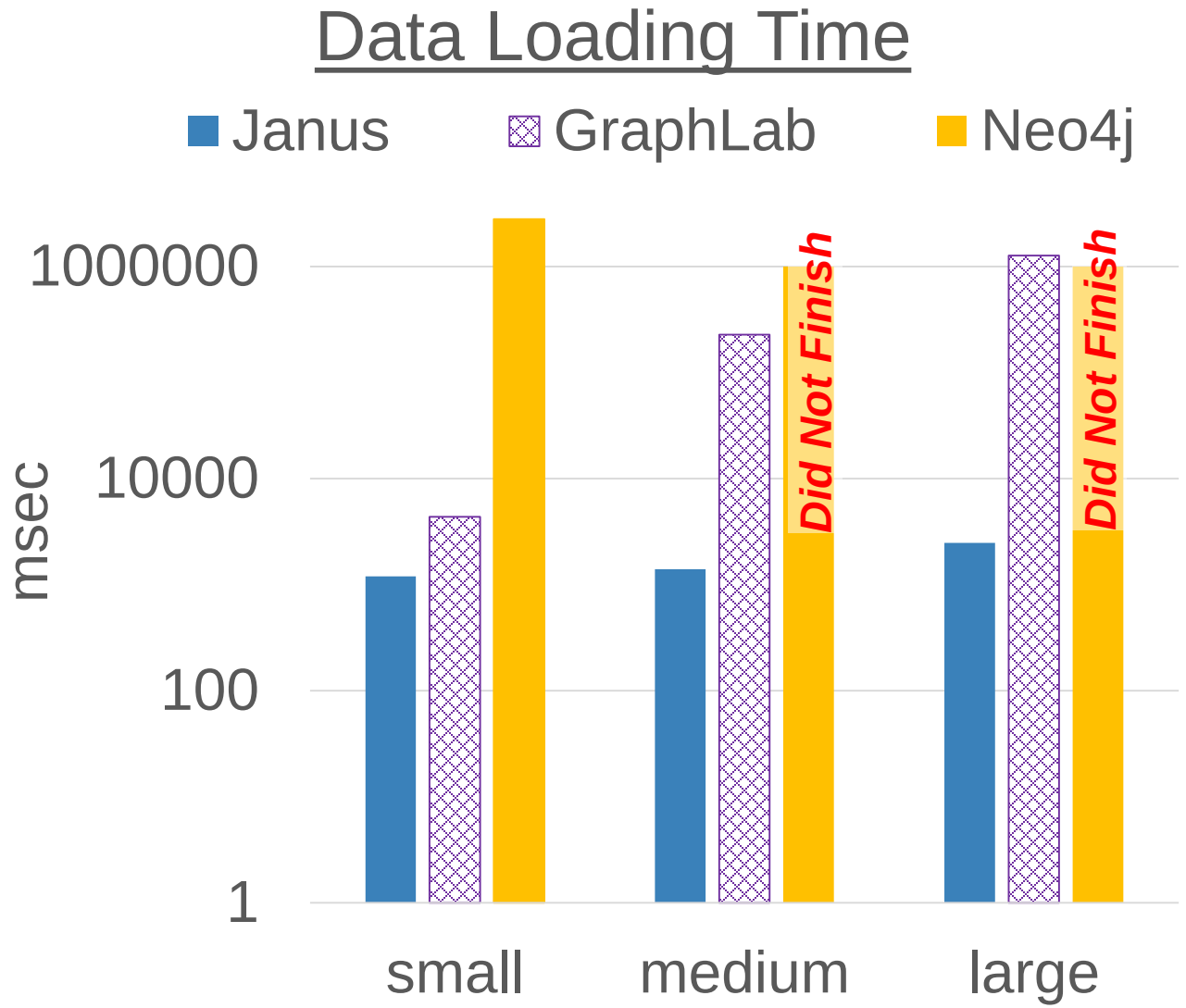
Navigational : Pair-wise

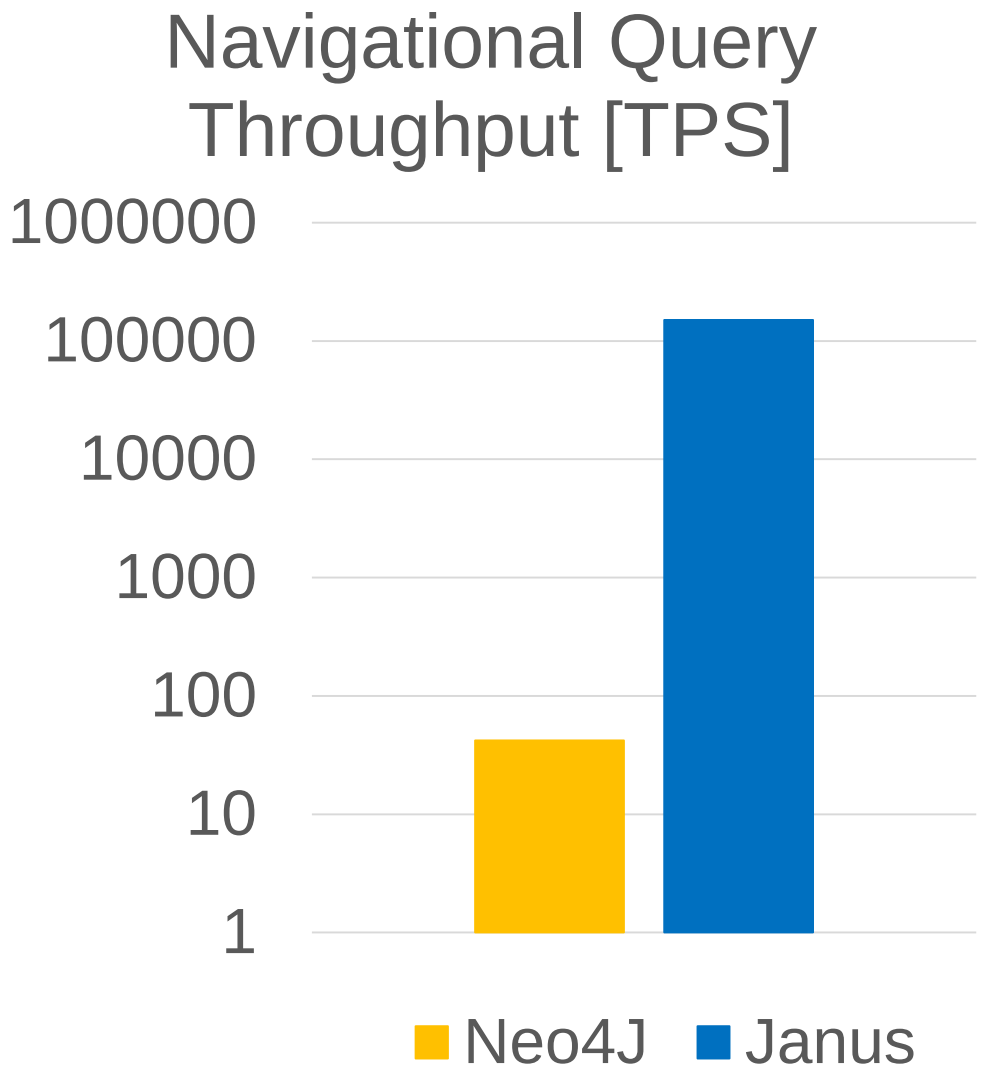
Analytic : SSSP

	# Nodes	# Edges
SMALL	2 M	37 M
MEDIUM	97 M	1600 M
LARGE	403 M	6500 M

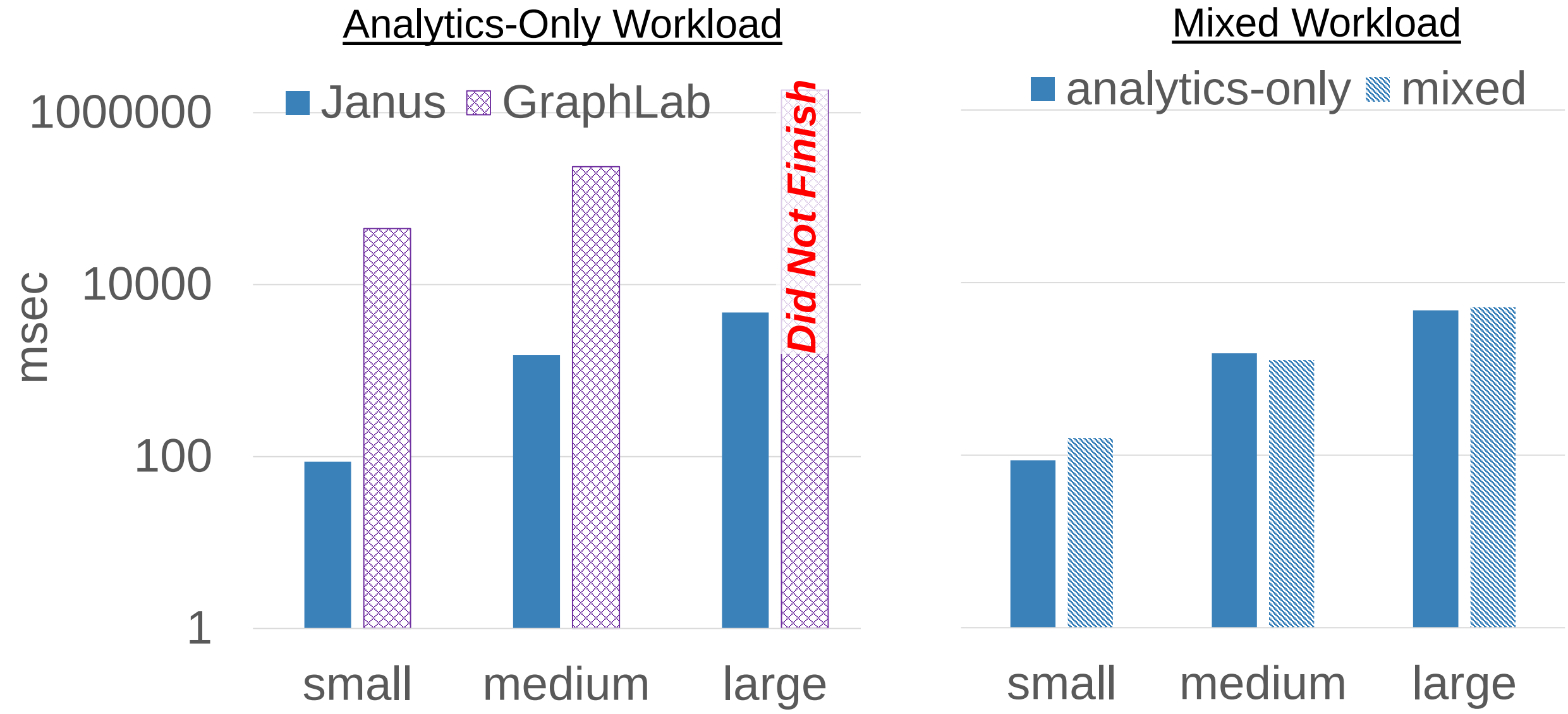
- Compared with *Neo4J* (navigational) and *Distributed GraphLab*
- H/W: HP DragonHawk, 240-Cores and 12 TB DRAM (not yet NVRAM) on 16-Sockets

Loading and Navigational Throughput





Analytic Query Runtime



Conclusions

- Janus : graph engine on future servers for navigational/analytic queries.
- Transaction is the key, breeding edge to massively parallelize big-data analytics.

Open Questions

- Not a panacea! *e.g., Topic Modeling*

Where's good fit?

- Autonomous Partition/Query Optimization e.g., when to activate/propagate nodes in what order
- Fast resume/failover with NVM