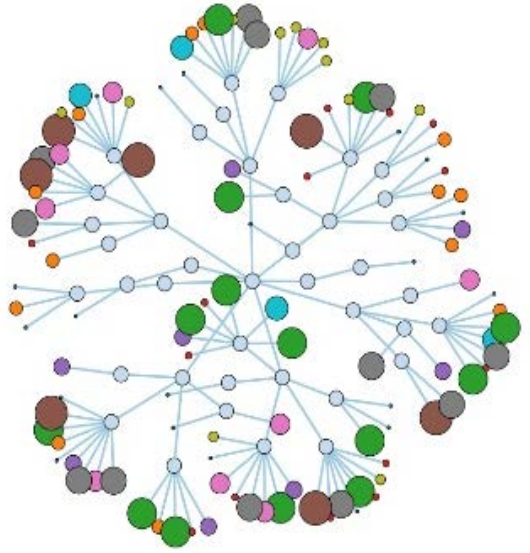
A Hybrid Solution for Mixed Workloads on Dynamic Graphs

Mahashweta Das, Alkis Simitsis, Kevin Wilkinson

Hewlett Packard
Labs

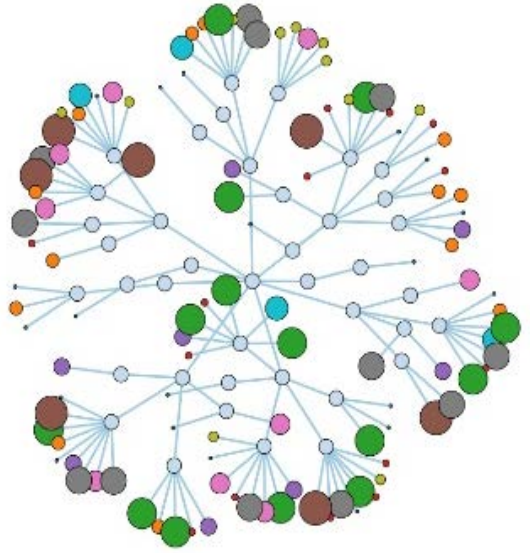
Background

- Graphs are everywhere!
  - social network, bioinformatics applications, transportation network, workforce management in business organizations ….
- Emergence of many new specialized graph management systems
  - storing, querying, processing, and analyzing graphs ….
  - tailored optimizations for different kinds of workloads, algorithms, and execution



Background

- Graphs are everywhere!
  - social network, bioinformatics applications, transportation network, workforce management in business organizations ….
- Emergence of many new specialized graph management systems
  - storing, querying, processing, and analyzing graphs ….
  - tailored optimizations for different kinds of workloads, algorithms, and execution



- Existing graph systems popularly classified into two categories:
  - **(i) navigation or online**: support high throughput and low latency for short requests that access relatively few graph vertices and edges (Example: Graph database Neo4j, RDF Store Jena, etc.)
  - **(ii) analytic or offline**: support long, resource-intensive, analytical computations and iterative batch processing that access a significant fraction of a graph (Example: GraphLab, Pregel, etc.)

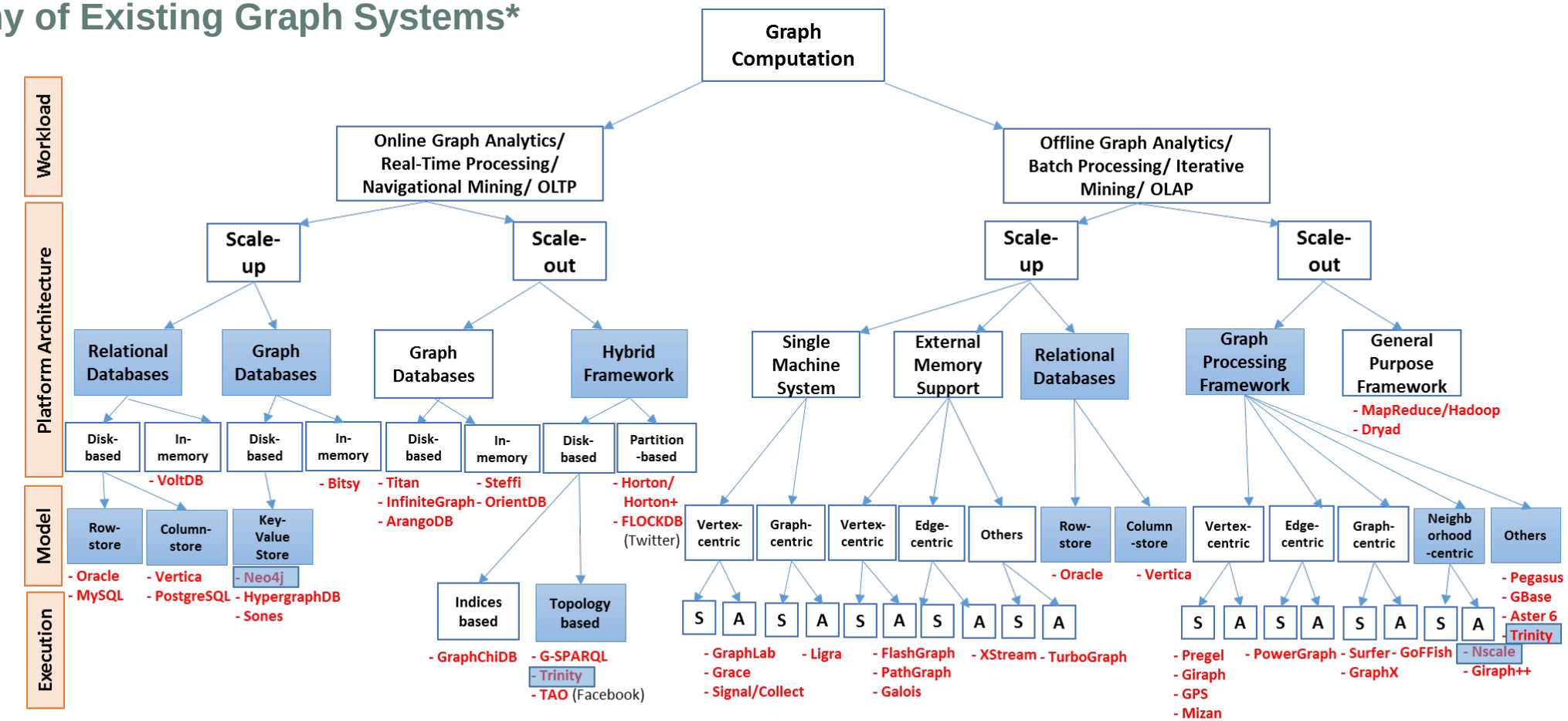
Background

Operational Analytics: Capture, analyze and react to events in real-time to improve business operations

- **Example: IT security analytics**
  - **capture DNS, proxy, netflow, syslog events to looking for attacks, intrusions, unusual behavior**
  - **IT assets (PCs, servers, printers, routers) come and go or are modified**
  - **security threat patterns come and go and black/white lists are modified**
- **Example: oil-gas production (and related IoT scenarios)**
  - **capture temperature, pressure, flow at drills to anticipate and avoid slowdowns or failures**
  - **drilling equipment status constantly changes, equipment added, moved or retired**
- **Example: national security tracking suspected terrorists**
  - **analytics run over snapshot of graph data as well as real-time graph**

Background

Taxonomy of Existing Graph Systems*



Systems than handle navig. and mining both workload S: Bulk Synchronous Parallel A: Asynchronous Parallel As of August 2014

Hewlett Packard Enterprise

Our Focus

- A **general** purpose graph data management system that
  - provides **efficient** and **concurrent** processing of graph navigation and graph analytic queries, i.e., **mixed workloads** for enterprise applications
  - enables enterprises to manage **real-time** graph, **dynamic** graphs, **historical** graph, and their **derived** graphs (views, i.e., application-specific models) in a single framework

- We call it **MAGS**: A **Ma**chine for **G**raph**s**

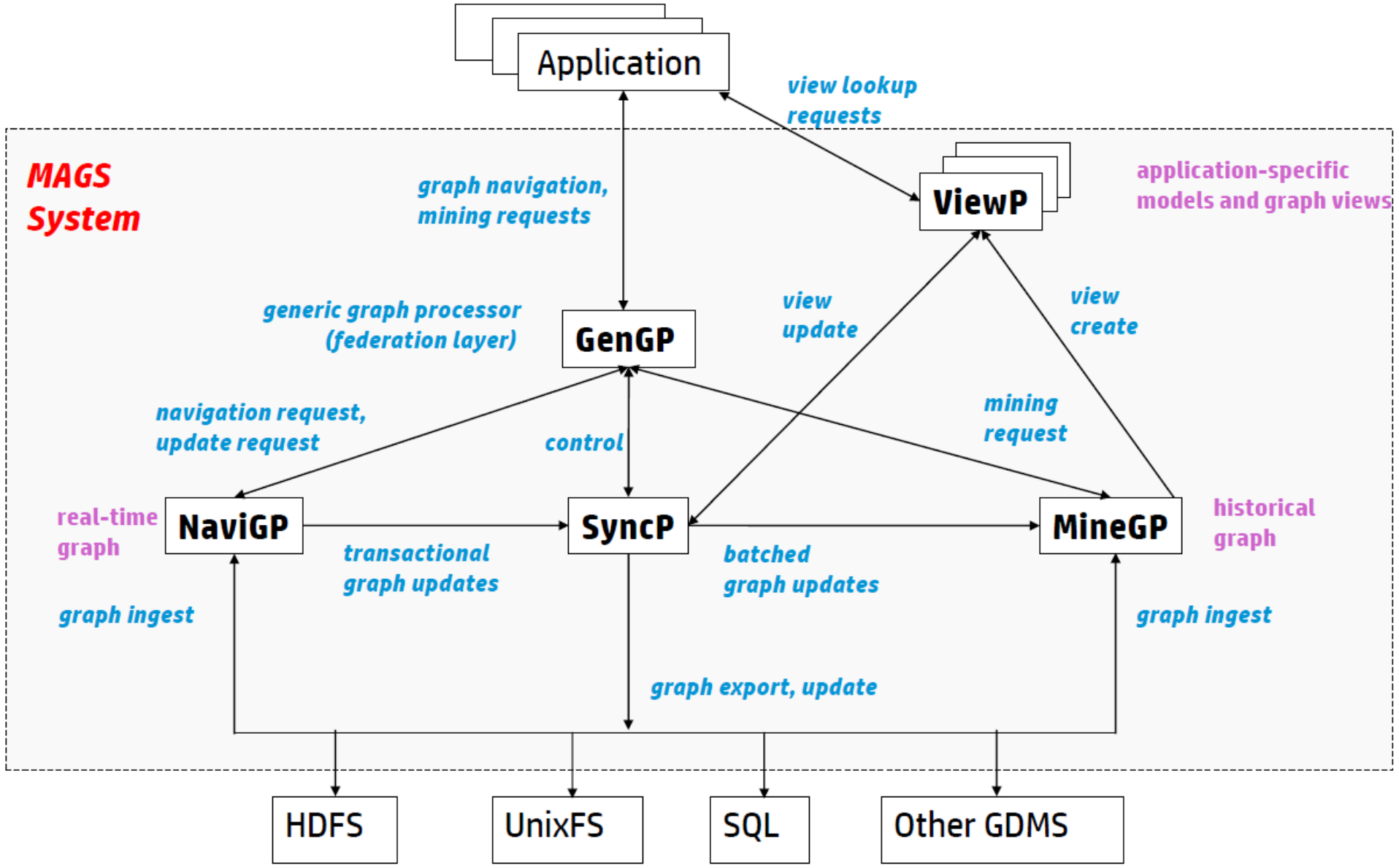
- We designed a flexible hybrid architecture that utilizes existing graph systems
- We developed a proof-of-concept
- We conducted experiments using the LDBC SNB workload to demonstrate its potential

Solution

- A **hybrid architecture** comprising two existing **graph systems** (one for each workload) with a **synchronization unit** to manage updates and a **federation layer** to present the hybrid system as a single API to graph applications.

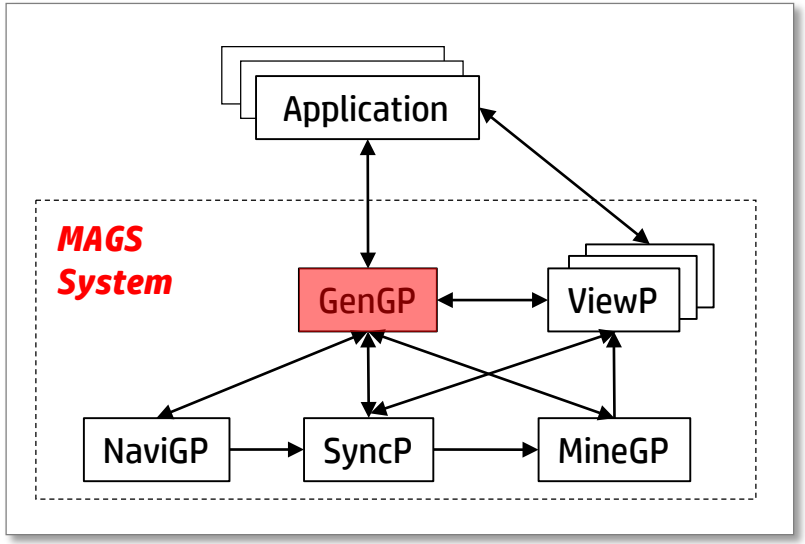
  - Key idea: segregate short navigation requests and updates on real-time graph from long analytic requests on historical graph
  - Key idea: separately tune the two graph systems to provide best performance for each workload
  - Key idea: prevent updates from interfering with analytic operations

Hybrid Architecture



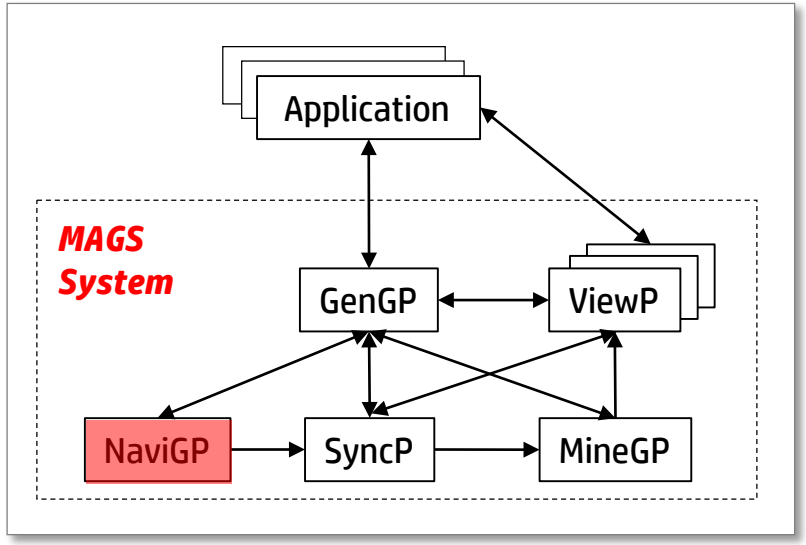
Hybrid Architecture: GenGP

- Application Interface
  - Provides a single unifying API for all graph applications
    - Currently Java based RESTFUL web service
  - Redirects graph requests to appropriate engines, i.e., query classification
    - Simple method: tags all requests from a particular application or user as one type or the other
    - Advanced method 1: classifier that compares features of an input query against a set of rules derived from previously executed queries in order to identify its class
    - Advanced method 2: simulating input query on a small synthetic graph to assess the proportion of nodes/edges accessed
  - Accepts graph queries in a wide variety of languages
    - Currently supports SQL
  - Other system management tasks!



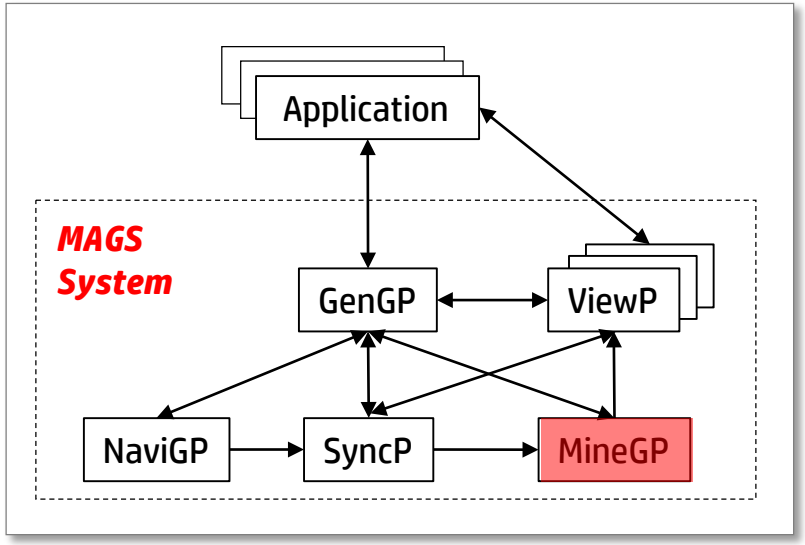
Hybrid Architecture: NaviGP

- Navigation Requests Processor
  - **Processes short graph requests** (Example: nearest neighbor, reachability query, etc.)
  - **Processes all update requests**
    - Real-time active graph
  - **Tuned for low-latency and high throughput**
  - **Potential choices: graph databases like Neo4j and OrientDB**



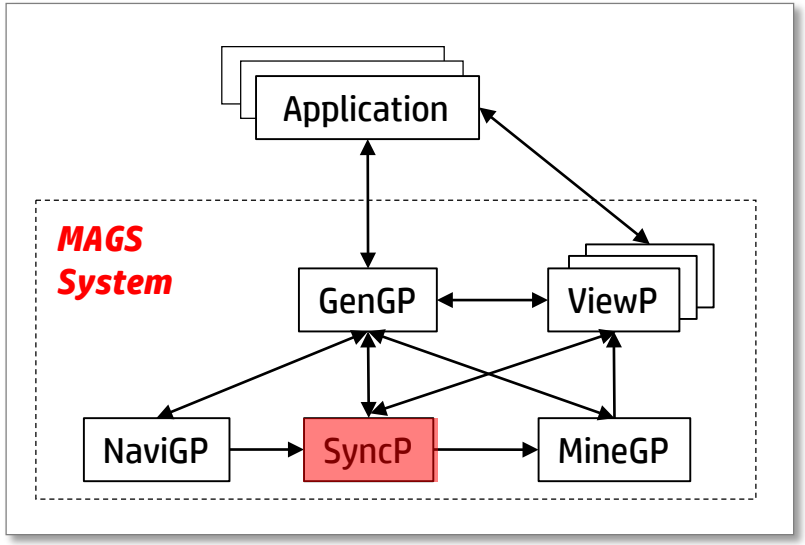
Hybrid Architecture: MineGP

- **Analytic Requests Processor**
  - **Processes all graph requests that are not classified as short or update** (Example: PageRank, social network analysis, etc.)
  - **Processes long, possibly iterative and batch requests**
    - **Historical graph**
  - **Potential choices: GraphLab, Pregel and Giraph**



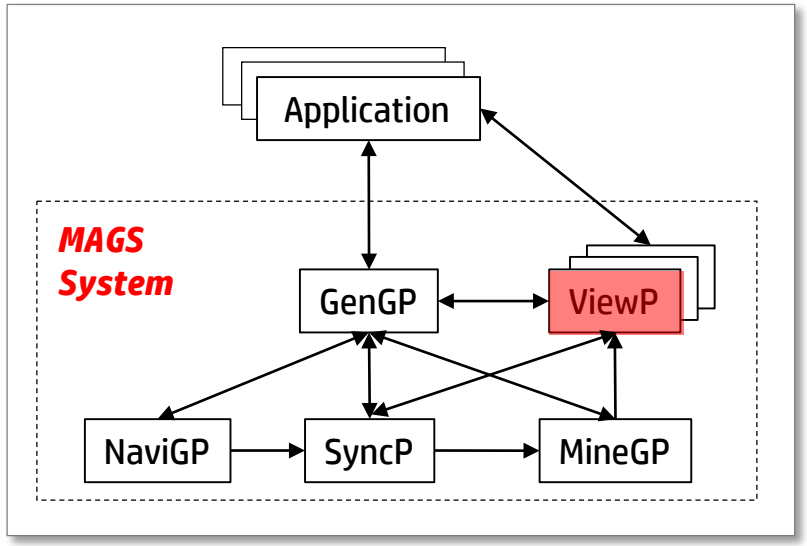
Hybrid Architecture: SyncP

- Synchronization Processor
  - Periodically collects the latest updates in the real-time graph in NaviGP, assembles them into a batch, and bulk loads the changes into MineGP
    - NaviGP changes collection using log-sniffing
    - Transactional bulk load using versioned tables in MineGP
  - Can tune the delay between historical graph and real-time graph
    - Typically in the order of 5-10 seconds
  - Sends transactionally consistent batched updates to application-specific derived views of the graph (in ViewP)



Hybrid Architecture: ViewP

- View Processor
  - Creates instances of application-specific models or views
    - Application probes model directly rather than graph
  - Updates or regenerates view when notified of changes made to the underlying graph in MineGP
  - Potential choice: GraphLab

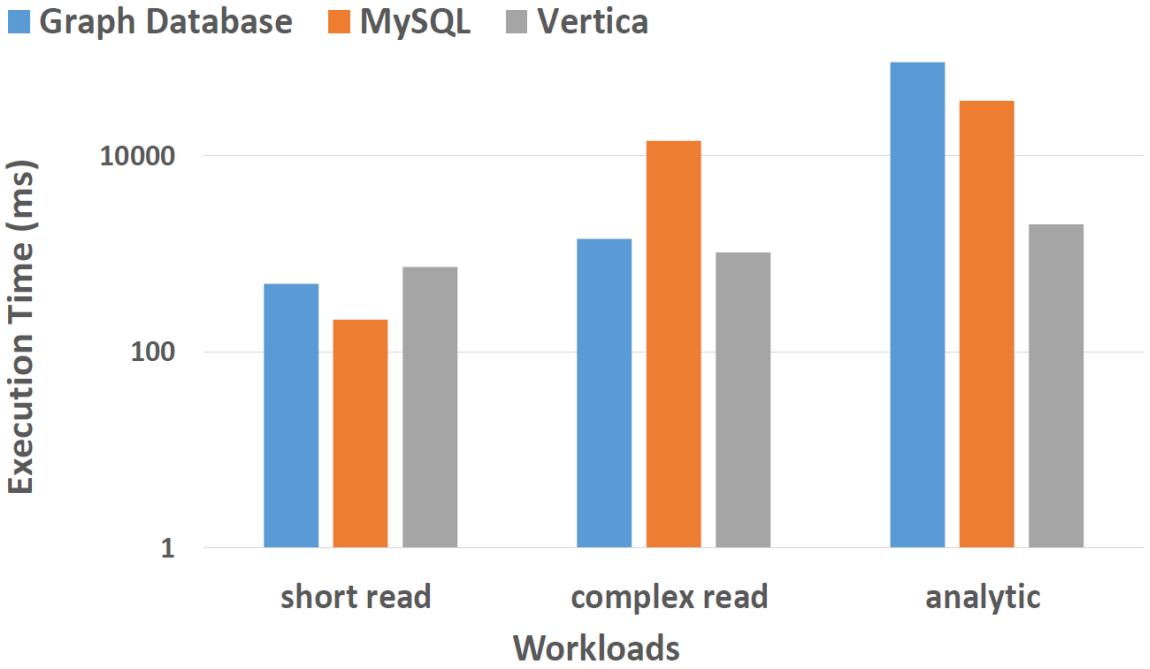


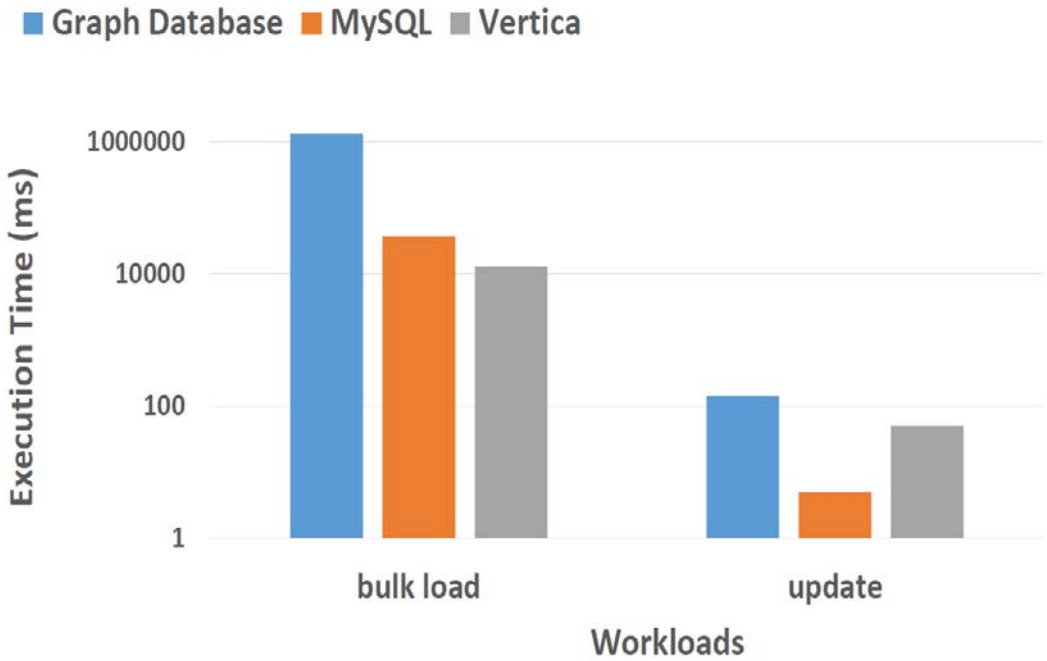
Proof-of-Concept

- Choice of engines:
  - Used off-the-shelf engines for NaviGP and MineGP for rapid prototyping
  - Performed a bake-off to select candidate engine comparing
    - Bulk load performance, update performance, short read performance (LDBC Social Network Benchmark interactive workload), complex read performance (LDBC Social Network Benchmark interactive workload), analytic (PageRank) performance

Proof-of-Concept

- Choice of engines:
  - Used off-the-shelf engines for NaviGP and MineGP for rapid prototyping
  - Performed a bake-off to select candidate engine comparing
    - Bulk load performance, update performance, short read performance (LDBC Social Network Benchmark interactive workload), complex read performance (LDBC Social Network Benchmark interactive workload), analytic (PageRank) performance

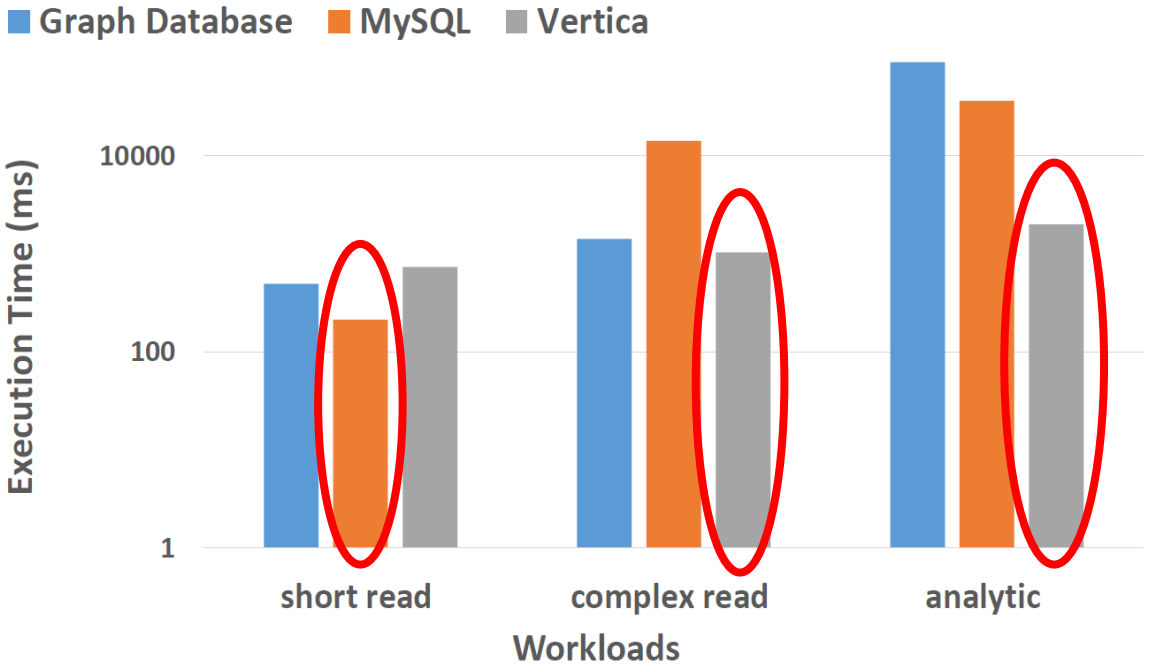


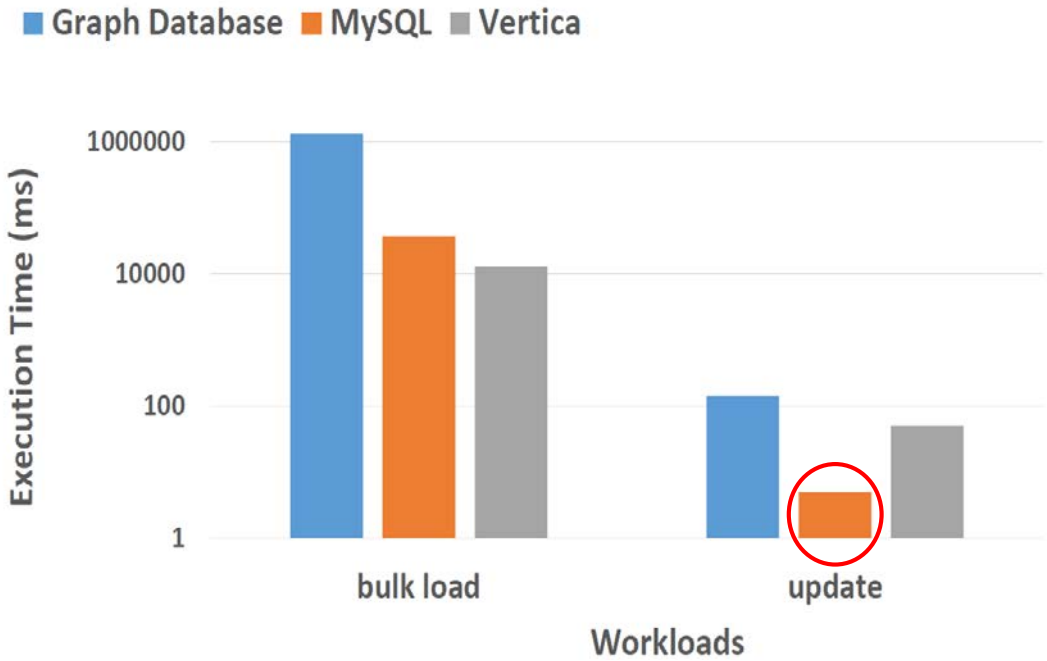


* Single machine with Intel Xeon E5-2660v2 (40 cores) and 128GB memory
* LDBC SNB graph at scale factor 1, i.e., 3M nodes, 20M edges for 10K persons and 1GB size.

Proof-of-Concept

- Choice of engines:
  - Used off-the-shelf engines for NaviGP and MineGP for rapid prototyping
  - Performed a bake-off to select candidate engine comparing
    - Bulk load performance, update performance, short read performance (LDBC Social Network Benchmark interactive workload), complex read performance (LDBC Social Network Benchmark interactive workload), analytic (PageRank) performance





* Single machine with Intel Xeon E5-2660v2 (40 cores) and 128GB memory
* LDBC SNB graph at scale factor 1, i.e., 3M nodes, 20M edges for 10K persons and 1GB size.

Proof-of-Concept

Implementation:

– **NaviGP:** MySQL

– **MineGP:** Vertica

– **SyncP:**

We modified LDBC SNB interactive workload to include inserts + deletes and demonstrated that synchronization has low impact on performance (Presented at LDBC TUC meeting on June 23)

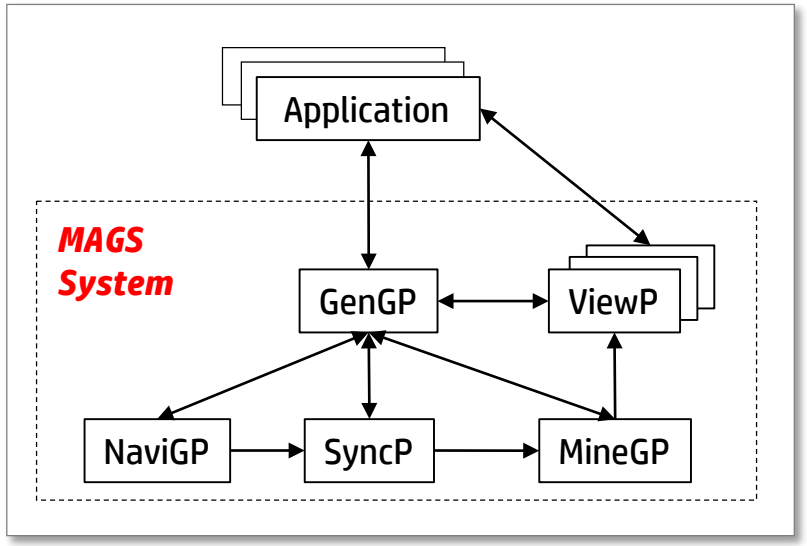
– **ViewP:** GraphLab

We implemented a Vertica-GraphLab bidirectional connector that uses shared memory to reduce data and function shipping overhead between two engines (Not the focus of this talk)

– **GenGP** query language: SQL

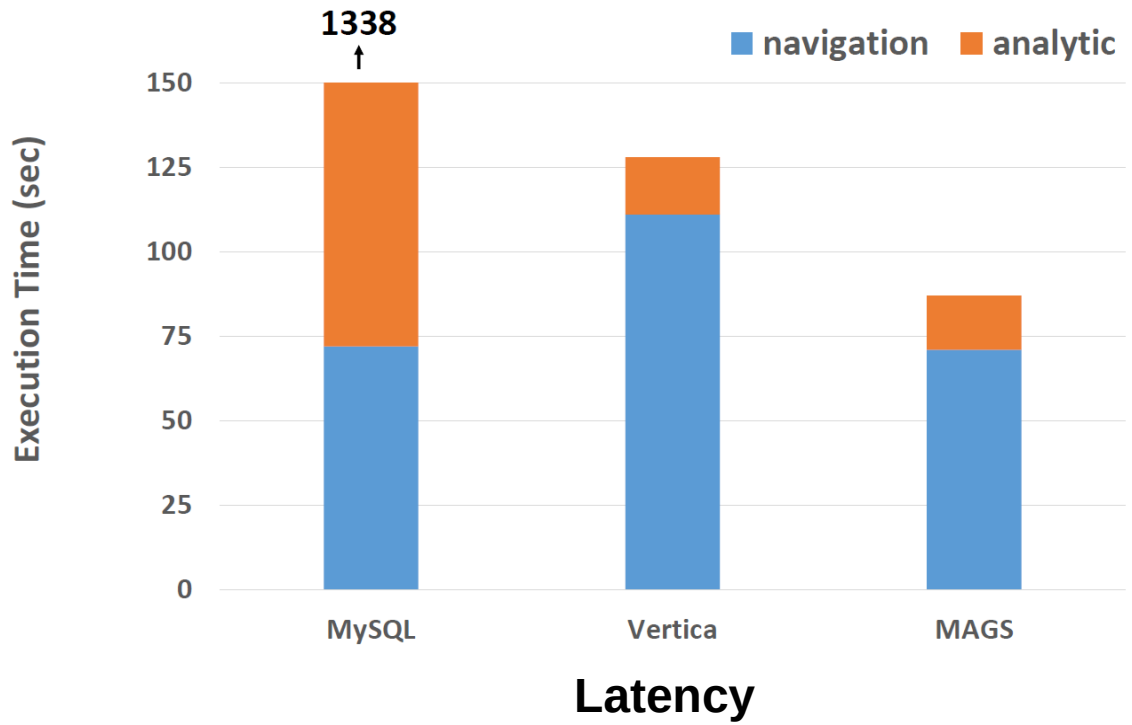
Workload:

- **LDBC Social Network Benchmark (SNB) interactive workload:**
  - short read, complex read
- **Additional queries:** Analytic (Page Rank), LDBC SNB inserts + deletes

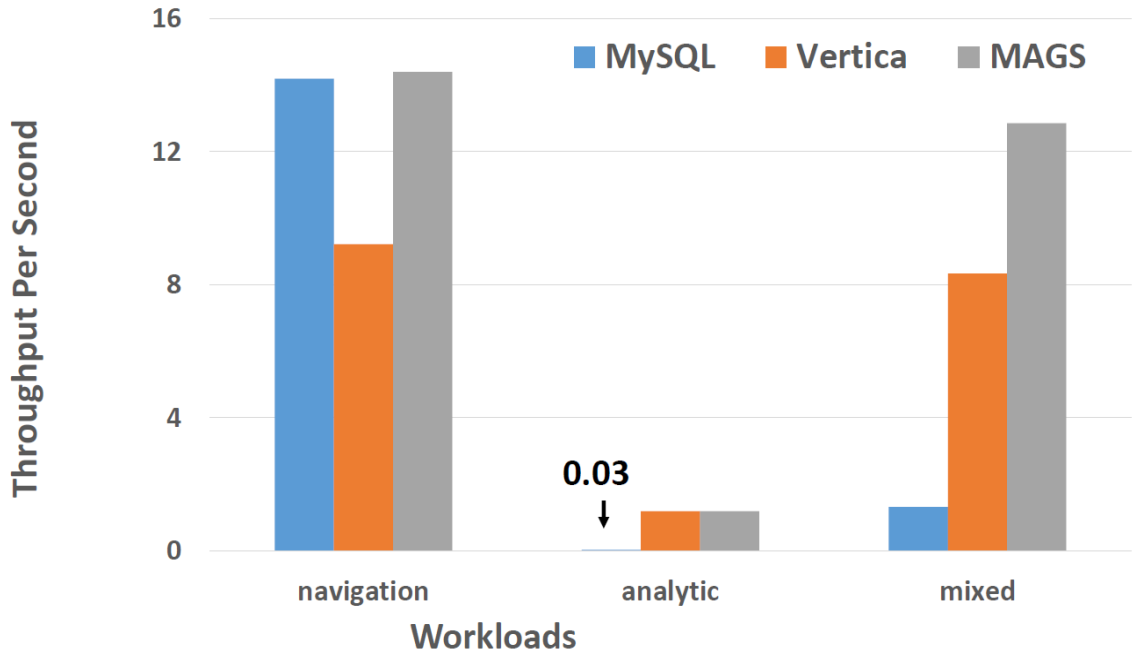


Experimental Validation

– LDBC SNB interactive workload complemented with additional analytic queries
  – 1041 queries: 1022 short requests (short read in LDBC SNB interactive workload)
    23 long requests (complex read in LDBC SNB interactive workload + PageRank)



Latency

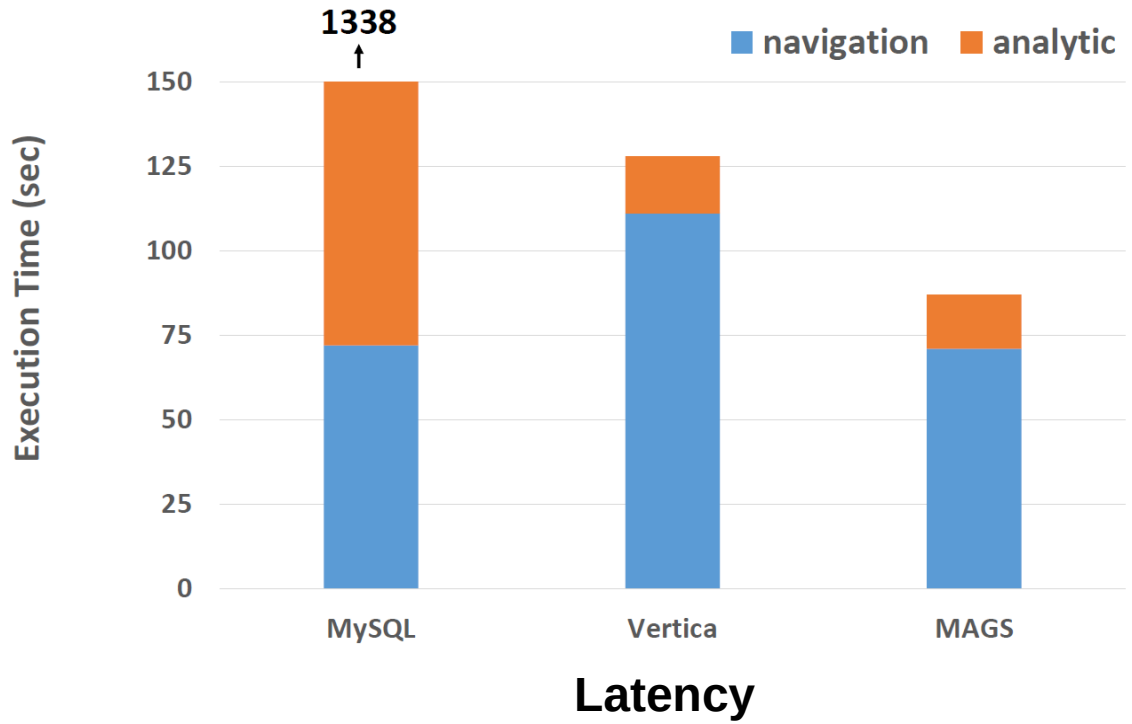


Throughput

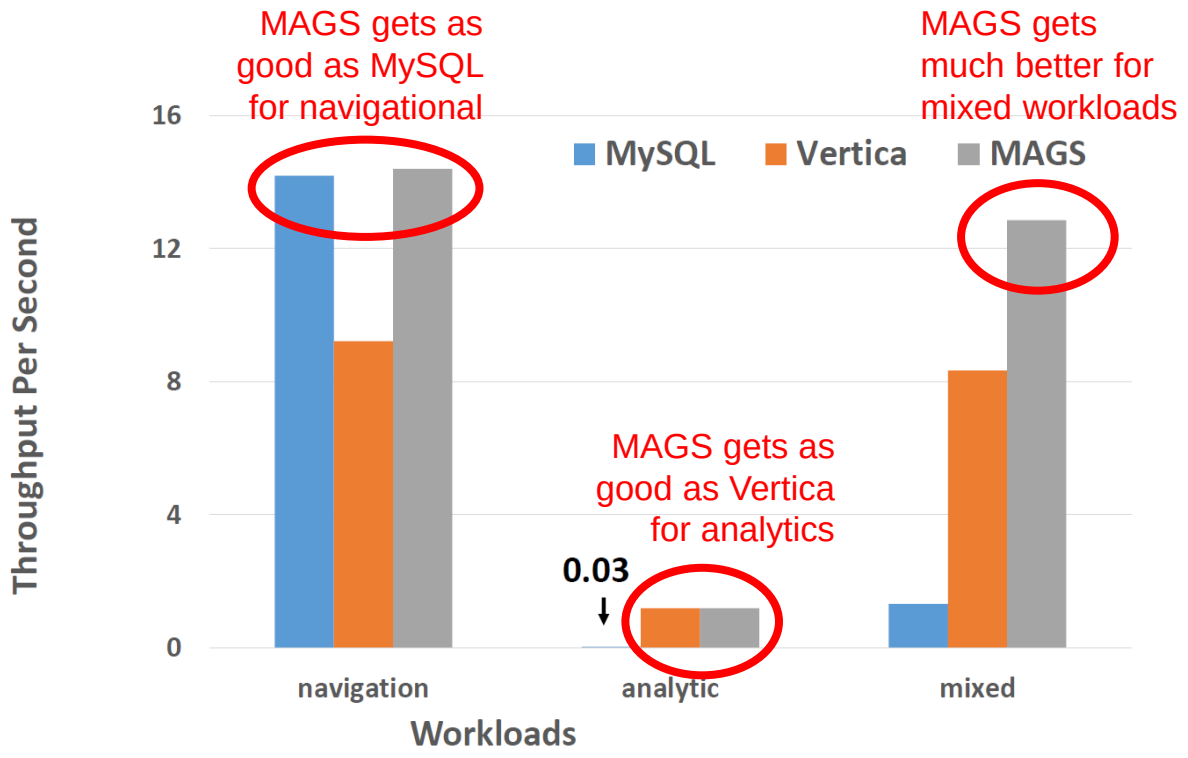
* Single machine with Intel Xeon E5-2660v2 (40 cores) and 128GB memory
* LDBC SNB graph at scale factor 1, i.e., 3M nodes, 20M edges for 10K persons and 1GB size.

Experimental Validation

– LDBC SNB interactive workload complemented with additional analytic queries
  – 1041 queries: 1022 short requests (short read in LDBC SNB interactive workload)
    23 long requests (complex read in LDBC SNB interactive workload + PageRank)



Latency



Throughput

* Single machine with Intel Xeon E5-2660v2 (40 cores) and 128GB memory
* LDBC SNB graph at scale factor 1, i.e., 3M nodes, 20M edges for 10K persons and 1GB size.

Hewlett Packard Enterprise

Summary and Future Work

- A flexible hybrid architecture that utilizes existing graph navigation engines and graph analytic engines for executing mixed workload efficiently and concurrently

- Explore scale-out architecture for the hybrid graph data management system

- Exploit next-generation hardware for improved latency and throughput

- Extend the hybrid graph data management system to handle multiple workloads coming from multiple graph applications

Thank You!

Questions?

Back up Slide: Synchronization