Graphalytics = From Benchmarking to Performance Engineering, leading to Massivizing Graph-Processing Systems

Tim Hegeman, Wing-Lung Ngai, and Stijn Heldens.

Presentation developed jointly with Ana Lucia Varbanescu.

Several slides developed jointly with Yong Guo.

dr. ir. Alexandru Iosup
Distributed Systems Group
(TU) Delft – the Netherlands – Europe

- Delft: founded 13th century, pop: 100,000
- Barcelona: pop: 16.5 M

Delft: founded 1842, pop: 15,000
Graphs Are at the Core of Our Society: The LinkedIn Example

The State of LinkedIn

400 million Q3 2015

A very good resource for matchmaking workforce and prospective employers

Vital for your company’s life, as your Head of HR would tell you

Vital for the prospective employees

Tens of “specialized LinkedIns”: medical, mil, edu, gov, ...

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
LinkedIn’s Service/Ops Analytics

The State of LinkedIn

3-4 new users every second

By processing the graph: opinion mining, hub detection, etc. Always new questions about whole dataset.

100+ million questions of customer retention, of (lost) customer influence, of ... Plus new hypotheses to test on whole dataset.

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
Why Analytics?

The State of LinkedIn

3-4 new users every second

but fewer visitors (and page views)

Periodic and/or continuous full-graph analytics

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
How to do Analytics? Graph Processing @large

A Graph Processing Platform

ETL (Extraction, Transformation, Loading)

Active Storage (filtering, compression, replication, caching)

Distribution to processing platform

Algorithm

Interactive processing not considered in this presentation.
Streaming not considered in this presentation.
Graph-processing is at the core of our society

The Data Deluge vs. Analytics

Graph processing @large
Which to select? What to tune? What to re-design?
A performance comparison of graph-processing systems
Take-home message
The data deluge: large-scale graphs

Social network
~1 billion vertices
~100 billion connections

Web graph
~50 billion pages
~1 trillion hyperlinks

Brain network
~100 billion neurons
~100 trillion connections

Source: Smith, CHI’10; Blog webpage; Gigandet et al., PLoS ONE 3(12)]
The data deluge: graphs everywhere!

LinkedIn

- 400M users
- ??? edges

Twitter

- 270M MAU
- 200+ avg followers
- >54B edges

Facebook

- 1.2B MAU
- 0.8B DAU
- 200+ avg followers
- >240B edges

Yahoo!

Friendster

XFire
The data deluge: graphs everywhere!

- LinkedIn: 1.2M followers, 132k employees
- Oracle: 1.2M followers, 132k employees
- Yahoo!: company/day: 40-60 posts, 500-700 comments
- Twitter: 270M MAU, 200+ avg followers, >54B edges
- Facebook: 1.2B MAU, 0.8B DAU, 200+ avg followers, >240B edges
- Friendster: XFire:
The data deluge vs. Analytics

Data-intensive workload
10x graph size ➔ 100x—1,000x slower
The data deluge vs. Analytics

LinkedIn

270M MAU
200+ avg followers
>54B edges
1.2B MAU 0.8B DAU
200+ avg followers
>240B edges

Oracle 1.2M followers, 132k employees

Data-intensive workload
10x graph size ➞ 100x—1,000x slower

Compute-intensive workload
more complex analysis ➞ ?x slower

friendster

TU Delft
The data deluge vs. Analytics

**Data-intensive workload**
10x graph size ➞ 100x—1,000x slower

**Compute-intensive workload**
more complex analysis ➞ ?x slower

**Dataset-dependent workload**
unfriendly graphs ➞ ??x slower
Your network is so large...

Sorry, but your network is too large to be computed, we are working to increase the limit, stay tuned!
The “sorry, but…” moment

Supporting multiple users
10x number of users ➔ ???x slower
What would **you** do to solve this?

**Data-intensive workload**
10x graph size → 100x—1,000x slower

**Compute-intensive workload**
more complex analysis → ??x slower

**Dataset-dependent workload**
unfriendly graphs → ??x slower

**Supporting multiple users**
10x number of users → ?????x slower
Graph-processing is at the core of our society
The data deluge vs. Analytics

**Graph Processing @Large**

Which to select? What to tune? What to re-design?
A performance comparison of graph-processing systems
Take-home message
Graph Processing @large

A Graph Processing Platform

Ideally, $N$ cores/disks $\rightarrow$ $N_x$ faster

(Distributing/compression, replication, caching)

Distribution to processing platform

Ideally, Parallel/Distributed/Heterogeneous

Ideally, $N$ cores/disks $\rightarrow$ $N_x$ faster

Interactive processing not considered in this presentation.
Streaming not considered in this presentation.
Graph processing systems

- **Dedicated Systems**
  - Systems for graph processing
  - Separate users from backends
  - Think Giraph

- **Custom Systems**
  - Specify application
  - Choose the hardware
  - Implement & optimize
  - Think Graph500 performers

- **Generic Systems**
  - Use existing distributed systems
  - Mapping is difficult
  - Parallelism is “free”
  - Think Hadoop/Spark

- **Performance**

- **Development Effort**
System diversity (CPU-based)

Intel Graphmat

ORACLE PGX

Neo4j the graph database

Trinity

YARN

Dedicated Systems

Generic
System diversity (GPU-enabled)

- **medusa-gpu**: Medusa: Simplified Graph Processing on GPUs
- **mapgraph**: Massively Parallel Graph processing on GPUs
- **Gunrock**: High-performance Graph Primitives on GPU
- **VertexAPI2**:
- **NetSysLab**:
- **TOTEM**:

**Dedicated Systems**

**Generic**
Graph-Processing Platforms

Platform: the combined hardware, software, and programming system that is being used to complete a graph processing task

Which to choose?
What to tune?
What to re-design?
Graph-processing is at the core of our society
The data deluge vs. Analytics
Graph processing @large

Graphalytics:
Which system to select?
What to tune? What to re-design?

A performance comparison of graph-processing systems
Take-home message
Benchmarking, but … What Is a Benchmark?

Benchmark definition must include:

- **Data schema**: data representation
- **Workloads**: formalize datasets + algorithms
- **Performance metrics**: from performance to non-traditional to cost-related
- **Execution rules**: how to run the benchmark tests, parameter values, etc.

Desirable support for stakeholders:

- Live addition of results
- Curation of added results
- Auditing results
What is the performance of graph-processing platforms?

- **Graph500**
  - Single application (BFS), Single class of synthetic datasets. @ISC16: future diversification.

- Few existing platform-centric comparative studies
  - Prove the superiority of a given system, limited set of metrics

- **GreenGraph500, GraphBench, XGDBench**
  - Issues with representativeness, systems covered, metrics, ...
What is the performance of graph-processing platforms?

- **Metrics Diversity**
- **Graph Diversity**
- **Algorithm Diversity**

Graphalytics = comprehensive benchmarking suite for graph processing across many platforms

Why is “LDBC” here?
BENCHMARKS

Here you may find the results for different benchmarks, i.e. the Social Network Benchmark (SNB) and the Semantic Publishing Benchmark (SPB), their definitions and best practices, the repositories where to find the data generators and the query implementations, an access to the intranet for the LDBC industry partners and a list of the LDBC member vendors.

READ MORE

LDLC official benchmarks for industry
Semantic Publishing Benchmark (SPB)

Why LDBC?
What are Graph Database systems?
What are RDF Database systems?
Why is benchmarking valuable?
What is the mission of LDBC?

The benchmarking community
Test the SPB and/or contribute to it
Test the SNB and/or contribute to it
Provide feedback on the P145
Graphalytics, in a nutshell

- An LDBC benchmark*
- Advanced benchmarking harness
- Many classes of algorithms used in practice
- Diverse real and synthetic datasets
- Diverse set of experiments representative for practice
- Granula for manual choke-point analysis
- Modern software engineering practices
- Supports many platforms
- Enables comparison of community-driven and industrial systems

http://graphalytics.ewi.tudelft.nl
https://github.com/tudelft-atlarge/graphalytics/
Iosup et al. LDBC Graphalytics: A Benchmark for Large Scale Graph Analysis on Parallel and Distributed Platform, VLDB’16.
### Graphalytics = Representative Classes of Algorithms and Datasets

- **2-stage selection process of algorithms and datasets**

<table>
<thead>
<tr>
<th>Class</th>
<th>Examples</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Graph Statistics</strong></td>
<td>Diameter, Local Clust. Coeff, PageRank</td>
<td>20</td>
</tr>
<tr>
<td><strong>Graph Traversal</strong></td>
<td>BFS, SSSP, DFS</td>
<td>50</td>
</tr>
<tr>
<td><strong>Connected Comp.</strong></td>
<td>Reachability, BiCC, Weakly CC</td>
<td>10</td>
</tr>
<tr>
<td><strong>Community Detection</strong></td>
<td>Clustering, Nearest Neighbor. Community Detection w Label Propagation</td>
<td>5</td>
</tr>
<tr>
<td><strong>Other</strong></td>
<td>Sampling, Partitioning</td>
<td>&lt;15</td>
</tr>
</tbody>
</table>

- + weighted graphs: Single-Source Shortest Paths (~35%)

---

Graphalytics = Distributed Graph Generation w DATAGEN

- Rich set of configurations
- More diverse degree distribution than Graph500
- Realistic properties, e.g., clustering coefficient and assortativity

Graphalytics

- Person Generation
- Edge Generation
- “Knows” graph serialization
- Activity Generation
- Activity serialization

Level of Detail
<table>
<thead>
<tr>
<th>Category</th>
<th>Experiment</th>
<th>Algo.</th>
<th>Data</th>
<th>Nodes/Threads</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline</strong></td>
<td>Dataset variety</td>
<td>BFS, PR</td>
<td>All</td>
<td>1</td>
<td>Run, norm.</td>
</tr>
<tr>
<td></td>
<td>Algorithm variety</td>
<td>All</td>
<td>R4(S), D300(L)</td>
<td>1</td>
<td>Runtime</td>
</tr>
<tr>
<td><strong>Scalability</strong></td>
<td>Vertical vs. horiz.</td>
<td>BFS, PR</td>
<td>D300(L), D1000(XL)</td>
<td>1—16/1—32</td>
<td>Runtime, S</td>
</tr>
<tr>
<td></td>
<td>Weak vs. strong</td>
<td>BFS, PR</td>
<td>G22(S)—G26(XL)</td>
<td>1—16/1—32</td>
<td>Runtime, S</td>
</tr>
<tr>
<td><strong>Robustness</strong></td>
<td>Stress test</td>
<td>BFS</td>
<td>All</td>
<td>1</td>
<td>SLA met</td>
</tr>
<tr>
<td></td>
<td>Variability</td>
<td>BFS</td>
<td>D300(L), D1000(L)</td>
<td>1/16</td>
<td>CV</td>
</tr>
<tr>
<td><strong>Self-Test</strong></td>
<td>Time to run/part</td>
<td>--</td>
<td>Datagen</td>
<td>1—16</td>
<td>Runtime</td>
</tr>
</tbody>
</table>
Graphalytics = Portable Performance Analysis w Granula

Minimal code invasion + automated data collection at runtime + portable archive (+ web UI) → portable bottleneck analysis
Graphalytics = Modern Software Engineering Process

Graphalytics code reviews
- Internal release to LDBC partners (first, Feb 2015; last, Feb 2016)
- Public release, announced first through LDBC (Apr 2015)
- First full benchmark specification, LDBC criteria (Q1 2016)

Jenkins continuous integration server
SonarQube software quality analyzer

https://github.com/tudelft-atlarge/graphalytics/
<table>
<thead>
<tr>
<th></th>
<th>MR2</th>
<th>Giraph</th>
<th>GraphX</th>
<th>PowerGraph</th>
<th>GraphLab</th>
<th>Neo4j</th>
<th>PGX.D</th>
<th>GraphMat</th>
<th>OpenG</th>
<th>TOTEM</th>
<th>MapGraph</th>
<th>Medusa</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCC</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>BFS</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>WCC</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>CDLP</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>P’Rank</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>SSSP</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

G=validated, on GitHub
V=validation stage

https://github.com/tudelft-atlarge/graphalytics/
# Implementation status

<table>
<thead>
<tr>
<th>MR 2</th>
<th>Giraph</th>
<th>GraphX</th>
<th>Power Graph</th>
<th>Graph Lab</th>
<th>Neo4j</th>
<th>PGX.D</th>
<th>Graph Mat</th>
<th>Open G</th>
<th>TOTEM</th>
<th>Map Graph</th>
<th>Medusa</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCC</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>BFS</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>WCC</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>CDLP</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>P’Rank</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>SSSP</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

G=validated, on GitHub
V=validation stage

Benchmarking and tuning performed by vendors
Graphalytics Capabilities: An Example

Graphalytics enables deep comparison of many systems at once, through diverse experiments and metrics.
Which system is the best? It depends… Algorithm + Dataset + Metric

OK, but … why is this system better for this workload for this metric?
Graph-processing is at the core of our society.
The data deluge vs. Analytics.
Graph processing @large.

Graphalytics:
Which system to select?
What to tune? What to re-design?

A performance comparison of graph-processing systems.
Take-home message.
Coarse-grained vs Fine-grained Evaluation (1)

**Coarse-grained Method**
- system viewed as a black-box
- Algorithms, Datasets, Resources
- Graph processing system
- Coarse-grained metrics
  - (Overall Execution Time)

**Fine-grained Method**
- system viewed as a white-box
- Algorithms, Datasets, Resources
- IO operations
- Processing
- Overheads
- Fine-grained metrics
  - (Stage 3 time, straggler tasks)

Fine-grained evaluation method is more comprehensive
Coarse-grained vs Fine-grained Evaluation (2)

**Abstract**

- **Coarse-grained Method**: knowledge at conceptual level
- **Graph Processing Systems**
- **Distributed Infrastructure**
- Few, coarse-grained results

**Granular**

- **Fine-grained Method**: knowledge at technical level
- **Fine-grained evaluation method is more comprehensive**
- **Many, fine-grained results**

… but more time-consuming, esp. to implement
Graphalytics: Granula Overview

Granular
Fine-grained Method

1. Modeling
   - Concepts
   - Feedback

2. Archiving
   - Information

3. Visualizing

https://github.com/tudelft-atlarge/granula/
Granula Modeller

Time-consuming, expert-only, Done only once per platform, but incrementally
Granula Modeller
Incremental Model of Graph-Processing in Giraph

**BSP Programming Model**
Granula Archiver

Performance Model

Graph Processing System

Logging Patch

Performance Analyzer

Granula Archiver

Job Performance Archive

Time-consuming, minimal code invasion, automated data collection at runtime, portable archive
Granula Visualizer
Portable choke-point analysis for everyone!

Computation imbalance!
Graph-processing is at the core of our society
The data deluge vs. Analytics
Graph processing @large
Graphalytics: Which system to select? What to tune? What to re-design?

A Performance Comparison of Graph-Processing Systems

Take-home message
No platform runs fastest for all graphs, but **Hadoop is the worst performer.** Not all platforms can process all graphs, but **Hadoop processes everything.**

*Always consider reliability!*
PageRank Algorithm: All GPU Platforms, Datasets

- Medusa, MapGraph fail on larger datasets, with MG failing earlier
- Medusa better for small datasets
- Totem the only system able to process all tried datasets

Better

Always consider reliability!

Runtime: The Platform Has Large Impact

2 orders of magnitude difference due to platform

M. Capota et al., Graphalytics: A Big Data Benchmark for Graph-Processing Platforms. SIGMOD GRADES 2015
Runtime: The Dataset Has Large Impact

Neo4j: MapReduce ~ 2:1

Neo4j: MapReduce ~ 1:2

Better

M. Capota et al., Graphalytics: A Big Data Benchmark for Graph-Processing Platforms. SIGMOD GRADES 2015
Throughput: The Dataset Has Large Impact

M. Capota et al., Graphalytics: A Big Data Benchmark for Graph-Processing Platforms. SIGMOD GRACES 2015
The Platform-Algorithm-Dataset (PAD) Triangle for Performance Engineering of Graph-Processing Systems

Algorithm

- In progress: Algorithms for different data types and graphs
- Overstudied: Performance is enabled
  Portability is disabled

Dataset

- Understudied: No systematic findings yet
  Must be correlated with the algorithm

Platform

Introduced by Ana Lucia Varbanescu.
Graph-processing is at the core of our society
The data deluge vs. Analytics
Graph processing @large
Graphalytics: Which system to select? What to tune? What to re-design?
A performance comparison of graph-processing systems

Take-Home Message
Take-Home Message

Graphalytics: unified view and benchmarking of tens of systems, with little effort

Granula: iterative, fine-grained, shareable performance evaluation to enable performance engineering

The P-A-D Triangle:
Performance = f (Algorithm, Dataset, Platform, …)

Towards addressing the graph data deluge with high performance, low development effort

See reading list on next slide.
Reading List


