GRAPHALYTICS

A Big Data Benchmark for Graph-Processing Platforms

https://github.com/tudelft-atlarge/graphalytics/

GRAPHALYTICS was made possible by a generous contribution from Oracle.

Alexandru Iosup,
Tim Hegeman,
Wing Lung Ngai,
Stijn Heldens,

Mihai Capotă, Arnau Prat, Peter Boncz, Hassan Chafi

Oracle

IBM

HUAWEI

Intel

UPC

CWI

Graphalytics: Benchmarking Graph-Processing Platforms
LDBC TUC Meeting
UPC Barcelona, March 2016
We now have 400 million LinkedIn members, more than half of whom live outside of the U.S. That’s enough to make LinkedIn the fourth largest country in the world. In celebration, we took a look back to see how much our membership has grown and diversified over the past five years. It’s a helpful reminder of not only where we’ve been, but also where we’re headed as we work to create economic opportunity for every professional in the world.

Sources: Vincenzo Cosenza, The State of LinkedIn, http://vincos.it/the-state-of-linkedin/
LinkedIn Is Not Unique: Data Deluge

LinkedIn

IBM

IBM 280k employee-users, 2.6M followers

company/day:
100+ posts, 1,000+ comments

LinkedIn

270M MAU
200+ avg followers
>54B edges

IBM

LinkedIn

Facebook

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

LinkedIn

LinkedIn

LinkedIn

LinkedIn

LinkedIn

LinkedIn

LinkedIn
Graph Processing @large

Interactive processing not considered in this presentation. Streaming not considered in this presentation.
A Graph Processing Platform

Ideally, N cores/disks → Nx faster

Distribution to processing platform

Ideally, N cores/disks → Nx faster

Interactive processing not considered in this presentation. Streaming not considered in this presentation.
Interactive processing not considered in this presentation.
Streaming not considered in this presentation.
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?
Graphalytics, in a nutshell

- An LDBC benchmark*
- Advanced benchmarking harness
- Diverse real and synthetic datasets
- Many classes of algorithms
- 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/
Benchmarking Harness

Iosup et al. LDBC Graphalytics: A Benchmark for Large Scale Graph Analysis on Parallel and Distributed Platform (submitted).
Graphalytics = Representative Classes of Algorithms and Datasets

- 2-stage selection process of algorithms datasets

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

Graphalytics

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

Level of Detail
Graphalytics = Portable Perf. Analysis w Granula

Modeling

Granula Performance Model

Monitoring

Performance Analyzer

Logging Patch

Graph Processing System

Archiving

Granula Archiver

Sharing

Granula Performance Archive

Minimal code invasion + automated data collection at runtime + portable archive (+ web UI) → portable bottleneck analysis
Graphalytics = Diverse Set of Automated Experiments

<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>Baseline</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>Scalability</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</td>
<td>Runtime, S</td>
</tr>
<tr>
<td>Robustness</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>Self-Test</td>
<td>Time to run/part</td>
<td>--</td>
<td>Datagen</td>
<td>1—16</td>
<td>Runtime</td>
</tr>
</tbody>
</table>
# Implementation status

<table>
<thead>
<tr>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>G</th>
<th>V</th>
<th>V</th>
<th>V</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>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
<td>V</td>
<td>V</td>
<td>V</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>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>WC C</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>G</td>
<td>G</td>
<td>G</td>
<td>V</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>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>PRank</td>
<td>--</td>
<td>G</td>
<td>G</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>G</td>
<td>V</td>
<td>V</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>SSSP</td>
<td>--</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>--</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>
</tbody>
</table>

G=validated, on GitHub
V=validation stage

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

<table>
<thead>
<tr>
<th></th>
<th>Map Reduce 2</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>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>
<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>WC C</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>--</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>--</td>
<td>G</td>
<td>G</td>
<td>--</td>
<td>--</td>
<td>G</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**

Delft University of Technology
Graphalytics Capabilities: An Example

Graphalytics enables deep comparison of many systems at once, through diverse experiments and metrics.

<table>
<thead>
<tr>
<th>Diverse datasets</th>
<th>Diverse algorithms</th>
<th>Diverse metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1(2XS)</td>
<td>PR</td>
<td>Edges per second</td>
</tr>
<tr>
<td>R2(XS)</td>
<td></td>
<td>(Vertices + Edges) per second</td>
</tr>
<tr>
<td>R3(XS)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R4(S)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G23(M)</td>
<td>BFS</td>
<td></td>
</tr>
<tr>
<td>D300(L)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Giraph
- GraphX
- P'Graph
- G'Mat
- OpenG
- PGX.D

Your system here!
Processing time (s) + Edges[+Vertices]/s

Which system is the best? It depends… Algorithm + Dataset + Metric

OK, but … why is this system better for this workload for this metric?
Granula Visualizer
Portable choke-point analysis for everyone!
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/
Graphalytics, in the future

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

+ more data generation
+ deeper performance metrics
+ choke-point analysis
PELGA – Performance Engineering for Large-scale Graph Analytics, workshop with EuroPar 2016