To Revisit Benchmarking Graph Analytics

A work collaborated by Shanghai Jiao Tong University and Alibaba Damo Academy

Presented by Longbin Lai, on behalf of Prof. Xuemin Lin
LDBC Graphalytics Benchmark

R1 Target platforms and systems

R2 Diverse, representative benchmark elements: Algorithms, Datasets, etc.

R3 Diverse, representative process: Performance, Scalability and Robustness

R4 Include a renewal process

R5 Modern software engineering

[1] Alexandru Losup, LDBC Graphalytics: A Benchmark for Large-Scale Graph Analysis on Parallel and Distributed Platforms, VLDB 2016
Why revisit the benchmark: Algorithm

- Selected algorithms are **representative** but not **diverse**

1. Algorithms: BFS, PR, WCC, CDLP, LCC, SSSP

2. Similar Computing Patterns: **ISVP (iterative, single-phased and value-propagation-based)**

3. The **appearance-dominated** selection procedure is biased

Our Proposal: Categorization

- **Centrality:** PageRank, Personalized PageRank, Degree Centrality, Betweenness Centrality, Closeness Centrality
- **Clustering/Community Detection:** Local Clustering Coefficient, Louvain, Label Propagation, Minimum Cut Algorithm
- **Similarity:** Cosine, Jaccard, SimRank
- **Community Search:** Core Decomposition, K-Truss, Clique, K-ECC, Biclique
- **Pattern Matching:** Triangle Counting, Subgraph Matching
- **Traversal/Path:** BFS, DFS, Single Source Shortest Path, Topological Sort, Minimum Spanning Tree, Max Flow, Cycle Detection
- **Other:** Strongly Connected Components, Weakly Connected Components, Maximum Independent Set, Color

Selection of LDBC
Our Proposal: Multi-dimensional

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Number of Papers</th>
<th>DBLP</th>
<th>Google Scholar</th>
<th>Web of Science</th>
<th>Time Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label Propagation</td>
<td>39</td>
<td>771</td>
<td>130000</td>
<td>699</td>
<td>$k \times m$</td>
</tr>
<tr>
<td>Single Source Shortest Path</td>
<td>33</td>
<td>584</td>
<td>17800</td>
<td>282</td>
<td>$m + n \times \log n$</td>
</tr>
<tr>
<td>K-Clique</td>
<td>31</td>
<td>352</td>
<td>39500</td>
<td>73</td>
<td>$k \times m \times a^{k-2}$</td>
</tr>
<tr>
<td>Core Decomposition</td>
<td>29</td>
<td>179</td>
<td>107000</td>
<td>454</td>
<td>$m + n$</td>
</tr>
<tr>
<td>PageRank</td>
<td>28</td>
<td>1012</td>
<td>21700</td>
<td>753</td>
<td>$k \times m$</td>
</tr>
<tr>
<td>Triangle Counting</td>
<td>27</td>
<td>252</td>
<td>21700</td>
<td>210</td>
<td>$m^{1.5}$</td>
</tr>
<tr>
<td>Betweenness Centrality</td>
<td>20</td>
<td>304</td>
<td>32100</td>
<td>283</td>
<td>$n^3$</td>
</tr>
<tr>
<td>Louvain</td>
<td>8</td>
<td>299</td>
<td>181000</td>
<td>127</td>
<td>$n \times \log n$</td>
</tr>
</tbody>
</table>

Categories: Appearsances: Academic Search Engines: Textbook Complexity:
Why revisit the benchmark: Datasets

- Selected datasets are narrow in

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Sizes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Real</strong></td>
<td><strong>Gen</strong></td>
</tr>
<tr>
<td>Social (Gaming)</td>
<td>SNB</td>
</tr>
<tr>
<td>Knowledge</td>
<td>Graph500</td>
</tr>
</tbody>
</table>

Graphs in real life are more diverse:
- Road/route networks are sparse
- Product-customer graphs are bi-partite
- etc.

<table>
<thead>
<tr>
<th>graph</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>datagen-9_3-zf</td>
<td>555M</td>
<td>1.3B</td>
</tr>
<tr>
<td>datagen-sf10k-fb</td>
<td>100M</td>
<td>18.8B</td>
</tr>
<tr>
<td>graph500-30</td>
<td>450M</td>
<td>34.0B</td>
</tr>
</tbody>
</table>

The latest graphalytics challenge includes much larger generated data.
Our Proposal: Gen with real-life characteristics

- GaoDe Road Network
- Taobao Product-Customer
- Ali Cloud Network Traffic

Graph characteristics Profiling

Massive data generator
Why revisit the benchmark: Process

• Platform-oriented Process
  • Performance: Makespan, Processing time
  • Scalability: Speedup
  • Robustness: Stress-test, Performance variability

• Our proposal
  • Platform-oriented + User-oriented
  • User-oriented
    • Expressiveness: can user implement certain algorithm
    • Productivity: how (easy) can user implement certain algorithm
Why revisit the benchmark: Software

• Modern but not **golden**
  • Software dependency issues
  • Repeated generation of some data
  • Hard to deploy in a cluster for large-scale

• Our Proposal:
  • Go cloud-native
    • Docker image: resolve software dependency issues
    • Cloud storage: for archiving the data (without repeatedly generating)
    • K8s for easy deployment in a cluster
    • etc
Wait, will this complicate the benchmark?

• More algorithms
• More/Larger datasets
• More metrics to evaluate
Our Proposal: Benchmark Hierarchies

- 8 core algorithms
- Others remain as LDBC Graphalytics
  - Suitable for competition/challenge

Core

- 8 core algorithms
- Massive datasets + Docker/Cloud Services, Perf metrics remain as LDBC Graphalytics
  - Suitable for PoC & HPC

Massive

- 20+ algorithms
- Massive datasets + Docker/Cloud Services + User-oriented perf metrics
  - Suitable for new-product promotion

Production
THANKS