Large-scale graph processing with emerging storage devices

Huibing Dong
GRAPHS ARE EVERYWHERE

Facebook
Amazon
Google
Instagram
Alibaba

...
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageRank</td>
<td>Ranking nodes based on their incoming/outgoing edges</td>
</tr>
<tr>
<td>Breath-First Search</td>
<td>Graph traversal based on breath</td>
</tr>
<tr>
<td>Shortest Path</td>
<td>Finding the path between 2 nodes where the sum of weights is minimal</td>
</tr>
<tr>
<td>Pattern Matching</td>
<td>Finding certain structures (e.g. path, star)</td>
</tr>
<tr>
<td>Triangle Count</td>
<td>Counting the number of triangles</td>
</tr>
<tr>
<td>Connected Component</td>
<td>Finding the subgraphs in which any two vertices are connected</td>
</tr>
</tbody>
</table>
## ALGORITHMS

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PageRank</td>
<td>Ranking nodes based on their incoming/outgoing edges</td>
</tr>
<tr>
<td>Breath-First Search</td>
<td>Graph traversal based on breath</td>
</tr>
<tr>
<td>Shortest Path</td>
<td>Finding the path between 2 nodes where the sum of weights is minimal</td>
</tr>
<tr>
<td>Pattern Matching</td>
<td>Finding certain structures (e.g. path, star)</td>
</tr>
<tr>
<td>Triangle Count</td>
<td>Counting the number of triangles</td>
</tr>
<tr>
<td>Connected Component</td>
<td>Finding the subgraphs in which any two vertices are connected</td>
</tr>
</tbody>
</table>

\[
PR(v^t) = 1 - d + d \times \sum_{\text{inedges}(v)} \frac{PR(u^{t-1})}{|\text{outedges}(u)|}
\]

*Iterative*  
*Neighbors matter*
CHALLENGES

- Expensive DRAM
- Graph layout
- Physical layout
- Irregular access

Expensive DRAM

Irregular access
<table>
<thead>
<tr>
<th>Architecture</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Shared-memory</strong></td>
<td>Memory, Single machine, Limited graph size</td>
<td>2013-Ligra, 2014-GraphX, 2015-Chaos</td>
</tr>
<tr>
<td><strong>Distributed</strong></td>
<td>Memory, Clusters, Costly</td>
<td>2010-Pregel, 2010-Graphlab, 2012-PowerGraph, 2014-GraphX, 2014-Blogel, 2015-Chaos, 2016-Gemini</td>
</tr>
</tbody>
</table>

Others architecture
- Graph-optimized database: Neo4j

*Blue: Semi-external systems*
Large scale graph processing systems: survey and an experimental evaluation, Omar Batarfi et. al, Cluster Computing'15
GRAPH REPRESENTATION

Graph

Adjacency list

- Sparse matrix
- Much less storage space needed

Adjacency matrix

Storage format: Compressed Sparse Column (CSC) & Compressed Sparse Row (CSR) files
EMERGING STORAGE DEVICES

Semi-external systems
- Vertex data
  - In the main memory
  - Fine-grained accesses, byte-addressable
- Edge data
  - On the secondary storage
  - Coarser accesses

External systems
- Even the vertex data itself is too large
- Both Vertex & Edge data on the secondary storage

HDD
- Slow
- Random access is a disaster

SSD
- Continue to scale
- Large capacity with lower latency
### Programming Model
- Vertex-centric
- Edge-centric
- *IO-centric

### Execution Model
- Bulk synchronous
- Asynchronous
## External systems

<table>
<thead>
<tr>
<th>Conference</th>
<th>System</th>
<th>Framework</th>
<th>Memory and Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSDI’12</td>
<td>GraphChi</td>
<td>Parallel Sliding Window</td>
<td>8GB DRAM + 256GB SSD + 750GB HD</td>
</tr>
<tr>
<td>KDD’13</td>
<td>Turbograph</td>
<td></td>
<td>12GB DRAM + 2x 512GB SSD</td>
</tr>
<tr>
<td>SOAP’13</td>
<td>X-Stream</td>
<td>Edge-centric</td>
<td>64GB DRAM + 2x 200GB</td>
</tr>
<tr>
<td>ICDE’15</td>
<td>Venus</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATC’15</td>
<td>GridGraph</td>
<td>2D partition</td>
<td>30.5GB DRAM + 2x 2TB HDD/1800 GB SSD</td>
</tr>
<tr>
<td>FAST’15</td>
<td>FlashGraph</td>
<td>semi-external; merges I/O</td>
<td>512GB DRAM + 15x OCZ Vertex 4 SSD</td>
</tr>
<tr>
<td>SC’16</td>
<td>G-Store</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAST’17</td>
<td>Graphene</td>
<td>Fine-grained IO</td>
<td>128GB DRAM + 16x 500GB SSD</td>
</tr>
<tr>
<td>EuroSys’17</td>
<td>Mosaic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISCA’18</td>
<td>GraFBoost</td>
<td></td>
<td>48GB DRAM + Xilinx VC707 FPGA + 2x 512GB SSD</td>
</tr>
<tr>
<td>FAST’19</td>
<td>GraphOne</td>
<td></td>
<td>512GB DRAM + 512GB SSD</td>
</tr>
<tr>
<td>ATC’19</td>
<td>Lumos</td>
<td>Dependency-Driven</td>
<td>64GB DRAM + 4x 1.9TB SSD</td>
</tr>
</tbody>
</table>
GraphChi
- Parallel Sliding Window
  - one sub-graph at a time
  - 3 phases
  - 1. Load
  - 2. Compute
  - 3. Write

X-stream
- Edge-centric
- Edge Centric Scatter
  - Edges (sequential read)
- Vertices (random read/write)
- Updates (sequential write)

GridGraph
- 2D partitioning
- Source Chunk 2
- Destination Chunk 1:
  - (1, 2)
  - (2, 1)
  - (1, 3)
  - (2, 4)
- Edge Block (2, 1)

FlashGraph
- Semi-external
- Merge edges I/O requests

Graphene
- Active vertices
- V3, V6, V1, V2, V3
- Bitmap
- Block id
- Neighbor list file
- Block
POTENTIAL RESEARCH PROBLEMS

- Programming model optimization
- Execution model support
- Partitioning
- Serializing
- Emerging Storage devices selection:
  - Zone-named space SSD
    - ZAC/ZBC
  - Open-channel SSD
  - Hybrid devices
  - Storage tiers