Computing Graphs at Scale

Felix Cuadrado

Centre for Intelligent Sensing
Queen Mary University of London
Outline

- Introduction to graphs
- Distributed graph processing
- Managing large dynamic graphs
Graph Definition

• A graph $G = (V,E)$, where
  - $V$ represents the set of vertices (nodes)
  - $E$ represents the set of edges (links)
  - Both vertices and edges may contain additional information

• Different types of graphs:
  - Directed vs. undirected edges
  - Presence or absence of cycles
Graphs model real world sources
Real-World Graph Applications

- Finding shortest paths
  - Routing Internet traffic and UPS trucks
- Finding Max Flow
  - Airline scheduling
- Bipartite matching
  - Monster.com, Match.com
- Rank search results
  - PageRank
- Identify leaders in a community
  - Measure influence
- Identify “special” nodes and communities
  - Breaking up terrorist cells
  - Track spread of ebola propagation
Graph-Processing Algorithms

• Literature survey of 10 top research conferences in databases and systems: SIGMOD, VLDB, WWW … [1]

<table>
<thead>
<tr>
<th>Class</th>
<th>Examples</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Statistics</td>
<td>Diameter, PageRank</td>
<td>16.1</td>
</tr>
<tr>
<td>Graph Traversal</td>
<td>BFS, SSSP, DFS</td>
<td>46.3</td>
</tr>
<tr>
<td>Connected Component</td>
<td>Reachability, BiCC</td>
<td>13.4</td>
</tr>
<tr>
<td>Community Detection</td>
<td>Clustering, Nearest Neighbor</td>
<td>5.4</td>
</tr>
<tr>
<td>Graph Evolution</td>
<td>Forest Fire Model, PAM</td>
<td>4.0</td>
</tr>
<tr>
<td>Other</td>
<td>Sampling, Partitioning</td>
<td>14.8</td>
</tr>
</tbody>
</table>
Outline

- Introduction to graphs
- Distributed graph processing
- Managing large dynamic graphs
Size of real world graphs

- World Wide Web:
  - Several billions of pages and links
- Twitter’s social graph has billions of follower relationships
- Facebook has one billion users

- ... in memory footprint: TBs to PBs
Distributed graph processing
Pregel: BSP Vertex-Centric Computation

- “Think like a vertex” [2]
- Inspired by Valiant’s Bulk Synchronous Parallel model [3]
Example: SSSP – Parallel BFS in Pregel

- Superstep 1
Example: SSSP – Parallel BFS in Pregel

- Superstep 1 - **Messaging**
Example: SSSP – Parallel BFS in Pregel

- Superstep 2
Example: SSSP – Parallel BFS in Pregel

- Superstep 2 - **Messaging**

![Graph with node labels and edge weights showing the SSSP and Parallel BFS process in Pregel.](image-url)
Example: SSSP – Parallel BFS in Pregel

- Superstep 3
Example: SSSP – Parallel BFS in Pregel

- Superstep 3 - **Messaging**
Example: SSSP – Parallel BFS in Pregel

• Superstep 4
Example: SSSP – Parallel BFS in Pregel

• Superstep 4 - **Messaging**
Example: SSSP – Parallel BFS in Pregel

• Superstep 5
How Pregel parallelises graph computations

- Goal: compute the graph algorithm when the graph is large (scale), in a reasonable time (performance)

- Approach:
  - **Partition** the graph among the computation nodes
  - Assign each worker to compute the function on its nodes

- Performance pitfalls
  - Uneven **load balancing**: some nodes wait for another
  - **Overheads** from the parallel execution: time wasted not doing the computation work
Balanced K-way graph Partitioning

• Given a graph \( G = (N, E) \)
  – \( N \) = nodes (or vertices),
  – \( E \) = edges

• Choose a partitioning \( N = N_1 \cup N_2 \cup \ldots \cup N_k \) such that
  – The cardinality of each \( N_i \) is “about the same”
  – The sum of all cut edges (edges connecting nodes from different partitions \( N_i \) and \( N_j \)) is minimized

• Problem is NP-Complete
Graph partitioning effect on performance

• In vertex-centric distributed computations partition is key
  – Balanced partitioning -> Load balancing
  – Not all edges are equal (some are cut, and take orders of magnitude more time to send messages through)

• Communication is the bottleneck
  – time required to send messages across cut edges is by far the dominant factor in computation time (>80%)

• …but scalability reasons make the default solution is random partitioning (uniform hashing).
System Architecture
Outline

• Introduction to graphs
• Distributed graph processing
• Managing large dynamic graphs
Real graphs are dynamic

Graph changes with time
Dynamic Graph Computation Cycle

W 1   W 2   W 3
Compute, Messaging
Update Graph
BSP SYNC BARRIER
Compute, Messaging
Update Graph
BSP SYNC BARRIER
Compute, Messaging
Partitioning dynamic graphs

Initial partitioning is not suitable for dynamic graphs

Repartitioning carries a high cost

Approach: iterative adaptive vertex migrations
Algorithm Key principles [5]

**Migrate** where your neighbors are

**Iterative** algorithm

**Migration Quotas**
Adaptive Migration Integration

BSP SYNC BARRIER

Migrate
Compute, Messaging
Update Graph
Decide Migrations

BSP SYNC BARRIER
CDR Analysis

One month of Call Detail Records
21 million nodes from phone calls
Nodes and edges expire with a sliding window

Quality of Partitioning  Maximum clique performance
Adapting Real Time Social Graphs

TunkRank (User influence metric) computation
Dataset: one week of tweets published from London
Biomedical Simulation

Heartbeat Stem Cells Simulation
110M Nodes
64 nodes cluster
3TB memory footprint
Summary

• Large graphs require parallel computation
  – Distributed systems research challenges include graph partitioning, load balancing, minimising communications overhead

• Partitioning is a key aspect of the performance of large-scale graph processing
  – Ratio of cuts $\leftrightarrow$ Performance
  – Scalability
  – Performance overhead

• Dynamic graphs make the problem more challenging
  – Adaptive migration algorithm for dynamic graph partitioning
References


THANKS!!

Félix Cuadrado

http://eecs.qmul.ac.uk/~fcuadrado
felix.cuadrado@qmul.ac.uk