Recent Advances in Multi-GPU Graph Processing

G. Carbone\textsuperscript{1}, M. Bisson\textsuperscript{2}, M. Bernaschi\textsuperscript{3}, E. Mastrostefano\textsuperscript{1}, F. Vella\textsuperscript{1}

\textsuperscript{1}Sapienza University Rome - Italy
\textsuperscript{2}NVIDIA U.S.
\textsuperscript{3}National Research Council – Italy

March 2015
Why Graph Algorithms

• Analyze large networks
  – Evaluate structural properties of networks using common graph algorithms (BFS, BC, ST-CON, ...)
  – Large graphs require parallel computing architectures

• High performance graph algorithm:
  – Most of graph algorithms have low arithmetic intensity and irregular memory access patterns
  – How do GPU perform running such algorithms?
  – GPU main memory is currently limited to 12GB
  – For large datasets, cluster of GPUs are required
Large Graphs

- Large scale networks include hundred million of nodes
- Real-world large scale networks feature a power law degree distribution and/or small diameter

<table>
<thead>
<tr>
<th></th>
<th># Vertices</th>
<th># Edges</th>
<th>Diameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>wiki-Talk</td>
<td>2.39E+06</td>
<td>5.02E+06</td>
<td>9</td>
</tr>
<tr>
<td>com-Orkut</td>
<td>3.07E+06</td>
<td>1.17E+08</td>
<td>9</td>
</tr>
<tr>
<td>com-LiveJournal</td>
<td>4.00E+06</td>
<td>3.47E+07</td>
<td>17</td>
</tr>
<tr>
<td>soc-LiveJournal1</td>
<td>4.85E+06</td>
<td>6.90E+07</td>
<td>16</td>
</tr>
<tr>
<td>com-Friendster</td>
<td>6.56E+07</td>
<td>1.81E+09</td>
<td>32</td>
</tr>
</tbody>
</table>

Source: Stanford Large Network Dataset Collection
Distributed Breadth First Search

• Developed according to the Graph 500 specifications
  – Generate edge list using RMAT generator
  – Support up to SCALE 40 and Edge Factor 16 (where $|V| = 2^{\text{SCALE}}$ and $|M| = 16 \times 2^{\text{SCALE}}$)
  – Use 64 bits for vertex representation
• Performance metric: Traversed Edges Per Second (TEPS)
• Implementation for GPU clusters
• Hybrid Programming paradigm: CUDA + Message Passing (MPI and APEnet)
• Level Synchronous Parallel BFS
• Data structure divided in subsets and distributed over computational nodes
1-D BFS

- 1-D Graph Partitioning
- Balanced thread workload
  - Map threads to data by using scan and search operations
- Enqueue vertices only once (avoiding duplicates)
  - Local mask array to mark both local and connected vertices
- Reduce message size
  - Communication pattern to exchange predecessor vertices only when BFS is completed avoiding sending them at each BFS level
  - Use 32 bits representation to exchange vertices instead of 64 bits
1-D Results

Weak Scaling Plot (RMAT Graph SCALE 21 – 31)
2-D BFS

• 2-D Graph partitioning
  – Improved scalability avoiding all-to-all communications

• Atomic Operations
  – Local computation leverages efficient atomic operations on Kepler
  – 2.3x improvement from S2050 (Fermi) to K20X (Kepler) on single GPU

• Further reduction of message size
  – Use a bitmap to exchange vertices among nodes
2-D Results

Weak Scaling Plot (RMAT Graph SCALE 21 – 33)
2-D Results

Weak Scaling Plot (RMAT Graph SCALE 21 – 33)

Up to \(\approx\)30 millions CUDA Threads
2-D BFS Bitmap based transfer

Use bitmap to exchange vertices information

**Without bitmap**

**With bitmap**

![Graph showing computation vs communication](image)

![Graph showing computation vs communication](image)
# 2D BFS Results on Real Graph*

<table>
<thead>
<tr>
<th>Data Set Name</th>
<th>Vertices</th>
<th>Edges</th>
<th>Scale</th>
<th>EF</th>
<th># GPUs</th>
<th>GTEPS</th>
<th>BFS Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>com-LiveJournal</td>
<td>4.00E+06</td>
<td>3.47E+07</td>
<td>22</td>
<td>9</td>
<td>2</td>
<td>0.77</td>
<td>14</td>
</tr>
<tr>
<td>soc-LiveJournal1</td>
<td>4.85E+06</td>
<td>6.90E+07</td>
<td>22</td>
<td>14</td>
<td>2</td>
<td>1.25</td>
<td>13</td>
</tr>
<tr>
<td>com-Orkut</td>
<td>3.07E+06</td>
<td>1.17E+08</td>
<td>22</td>
<td>38</td>
<td>4</td>
<td>2.67</td>
<td>8</td>
</tr>
<tr>
<td>com-Friendster</td>
<td>6.56E+07</td>
<td>1.81E+09</td>
<td>25</td>
<td>27</td>
<td>64</td>
<td>15.68</td>
<td>24</td>
</tr>
</tbody>
</table>

*Source: Stanford Large Network Dataset Collection*
ST-CON

• Decision problem
  – Given source vertex $s$ and destination vertex $t$ determine if they are connected
  – Output the shortest path if one exists

• Straightforward solution by using BFS
  – Start a BFS from $s$ and terminate if $t$ is reached

• Parallel ST-CON
  – Start two BFS in parallel from $s$ and $t$
  – Terminate if the two paths meet
Distributed ST-CON

- Atomic-operations based solution
  - Use atomic operations to update visited vertices
  - Finds only one s-t path

- Data structure duplication solution
  - Use distinct data structures to track s and t paths
  - At each BFS level check if there are vertices visited by both
  - Finds all s-t paths

- Performance metric
  - Number of s-t Pairs Per Second (NSTPS)
  - Execute ST-CON algorithm over a set of s-t pairs randomly selected
ST-CON Results

Weak Scaling Plot (RMAT Graph SCALE 21 – 27)
ST-CON Results

Weak Scaling Plot (RMAT Graph SCALE 19 – 26)
Only Parallel Atomic with different Edge Factor
Bernaschi, M., Carbone, G., Mastrostefano, E., & Vella, F.
Solutions to the st-connectivity problem using a GPU-based distributed BFS.
*Journal of Parallel and Distributed Computing, Volume 76, Pages 145-153 February 2015*
Betweenness Centrality

Misure of the influence of a node in a given network used in network analysis, transportation networks, clustering, etc.

$$BC(v) = \sum_{s \neq t \neq v} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

- $\sigma_{st}$ is the number of shortest paths from s to t
- $\sigma_{st}(v)$ is the number of shortest paths from s to t passing through v

- Best known sequential algorithm requires $O(mn)$ time-complexity and $O(n+m)$ space-complexity (Brandes2001)
- No satisfactory performance for large-scale graphs (biology systems and social networks)
Distributed BC

- Parallel distributed based on Brandes algorithm

Dependency is:
\[ \delta_s(v) = \sum_{w \in \text{Succ}(v)} \frac{\sigma_{sv}}{\sigma_{sw}} (1 + \delta_s(w)) \]

BC scores become:
\[ BC(v) = \sum_{s \neq v} \delta_s(v) \]

- 2D BFS as building block
- Distributed dependency accumulation

- Preliminary results - R-MAT graph Scale 21 with 2M nodes and \( \approx 32M \) Edges requires about 20 hours on 4 K40 GPUs !!
Conclusions

• Best algorithm has still $O(mn)$ complexity

• Reduce $n$
  – 1-degree reduction ($\approx 15\%$ on R-MAT) Sarýyüce2013, Baglioni2012
  – 2-degree reduction ($\approx 8\%$ on R-MAT)
  – Further heuristics to reduce the size of the graph to be analyzed

• Improve parallelism
  – Multi-source BFS
Thank You!

giancarlo.carbone@uniroma1.it

Please complete the Presenter Evaluation sent to you by email or through the GTC Mobile App. Your feedback is important!