Fast Iterative Graph Computation with Block Updates

Wenlei Xie*, Guozhang Wang+, David Bindel*, Alan Demers* and Johannes Gehrke*
*Cornell University, +LinkedIn

Multicore Speedup for Graph Applications

![Graph Applications Speedup](image)

Datasets and Applications

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Vertices $\times 10^8$</th>
<th>Edges $\times 10^8$</th>
<th>Partition Time (s)</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBLP</td>
<td>968</td>
<td>7,050</td>
<td>38</td>
<td>PPR</td>
</tr>
<tr>
<td>Web-Google</td>
<td>876</td>
<td>5,105</td>
<td>34</td>
<td>PPR</td>
</tr>
<tr>
<td>LiveJournal</td>
<td>4,848</td>
<td>68,994</td>
<td>659</td>
<td>SSSP</td>
</tr>
<tr>
<td>3D Grid</td>
<td>1,728</td>
<td>9,858</td>
<td>N/A</td>
<td>Eich Sm</td>
</tr>
<tr>
<td>UK02</td>
<td>18,520</td>
<td>298,114</td>
<td>103-4</td>
<td>PPR</td>
</tr>
</tbody>
</table>

Computation Load: Heavy vs. Light

![Computation Load](image)

Two-Level Scheduling

- Define block update as iteratively applying vertex update
  - BlockUpdate = VertexUpdate $\times$ InnerScheduler
- Block-Level Scheduler
- Benefits
  - Better Cache Utilization
  - Reduced Scheduling Overhead

Block-Oriented Computation

- Block Formulation
  - Block: Closely connected subgraph
  - Graph is pre-partitioned into disjoint blocks
  - Efficient software: (e.g. METIS)
- Block Update Function
  $S^{'new}_{B} = \text{BlockUpdate}(S^{'old}_{B}, S_{\text{V}(B)}, S_{\text{NE}(B)})$.
  - Naturally extends the vertex update function

Experimental Results

- Effect of block size (Time vs. Block size)
- Effect of block strategy (Time vs. scheduling policy)
- Effect of load balance (Time vs. inner iterations)

Example: Shortest Path

![Shortest Path Example](image)

Vertex vs. Block Update

(a) Vertex-Oriented Computation  (b) Block-Oriented Computation

![Vertex vs. Block Update](image)