Fast Iterative Graph Computation with Block Updates

Xie, et al.
(Proceedings of the Very Large Database Endowment, 2013)

Review by Matthew Huxtable
R212: 13th November 2014
The problem
The problem

Memory access bandwidth doesn't scale with processor performance

Image reproduced from STREAM project website
http://www.cs.virginia.edu/stream/ref.html
The problem

Vertex-centric computation performance poor

Computationally light algorithms suffer the most
(i.e. the common ones: PageRank, shortest paths, SCC, etc.)
The problem (as seen in practice)
The novel computation model
The novel computation model

(using cache blocking)
The novel computation model

- Process updates in cache line granularity
The novel computation model

- Process updates in cache line granularity

Keep the vertex-centric programming abstraction. (How?)
In practice
In practice

- Dual layer scheduler (Eager, Prior)
In practice

- Dual layer scheduler (Eager, Prior)
- Multiversion concurrency control

CPU 1  CPU 2  CPU 3  ...

Block Boundary
The evaluation
Scheduling policies

Effect of block size (Time vs block size)

<table>
<thead>
<tr>
<th>PPR(Google) (Default BS=100)</th>
<th>Etch Sim (Default BS=125)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run time (s)</td>
<td>Run time (s)</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>0  500  1,000  1,500</td>
<td>0  500  1,000  1,500</td>
</tr>
</tbody>
</table>

SSSP (Default BS=100)

<table>
<thead>
<tr>
<th>Run time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0  500  1,000  1,500</td>
</tr>
</tbody>
</table>

PPR(UK) (Default BS=400)

<table>
<thead>
<tr>
<th>Run time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0  500  1,000  1,500</td>
</tr>
</tbody>
</table>

Static  Eager  Prior
The problem
What problem?
How do we pick vertices to form blocks?
Conclusions

- Block-parallel graph-centric framework
- Dynamically scheduled blocks containing >1 vertex per block
- Better cache interaction over alternatives → faster programs (dubious?)
- Useful in common cases: Dijkstra, SCC, PageRank
Conclusions

- Block-parallel graph-centric framework
- Dynamically scheduled blocks containing >1 vertex per block
- Better cache interaction over alternatives → faster programs (dubious?)
- Useful in common cases: Dijkstra, SCC, PageRank
- Verdict on paper...