A Case for Work-stealing on FPGAs with OpenCL Atomics

Nadesh Ramanathan, John Wickerson, Felix Winterstein, George A. Constantinides

22nd February 2016
Motivation

- HLS tools do well on “regular” structures with static analysis
- But what if your computation is “irregular”?
  - data-dependent execution time
  - dynamic task creation
  - e.g. tree traversal
- Workload imbalance may result in idle processing units
- Processing units need to communicate with each other to balance workload
Atoms

- Inter-thread communication: Atomic operations (atomics) are fastest in multiprocessor world!
- Atomics appear to occur *instantaneously* (read-modify-write)
- For FPGAs, atomics are either not supported or “expensive”

**Attention:** The use of atomic functions might *lower* the *performance* of your design.


- Atomics allow for parallel algorithms without critical sections
Work-stealing with atomics

- Task-based dynamic load balancing for irregular workloads
- Cederman-Tsigas explored work-stealing on GPUs in 2008
- Primitive: Double-ended queues with atomics
Case Study: \(K\)-means Clustering

- \(kd\)-tree traversal to find closest center for every data point
- Task of traversing a node is irregular because
  - Pruning of potential center candidates (Elimination process)
  - Dynamic traversal decisions
- Previous FPGA implementations rely on static partitioning
- With work-stealing on OpenCL, \(1.5 \times \) faster!
### Results

<table>
<thead>
<tr>
<th>Work-items</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speedup</td>
<td>0.8×</td>
<td>1.0×</td>
<td>1.2×</td>
<td>1.5×</td>
<td>1.7×</td>
<td>1.9×</td>
</tr>
<tr>
<td>RAM overhead</td>
<td>57%</td>
<td>45%</td>
<td>68%</td>
<td>58%</td>
<td>58%</td>
<td>39%</td>
</tr>
</tbody>
</table>
Conclusion

- We show **1.5× speedup** for K-means clustering **despite the use of atomics**
- The use of atomics in the right context can lead to improved hardware performance
- Our paper describes more details and OpenCL code is available at 
  https://github.com/nadeshr/kmeans-stealing.git
- See you at our poster!

Email: n.ramanathan14@imperial.ac.uk