A GPU Implementation of Tiled Belief Propagation on Markov Random Fields

Hassan Eslami
Theodoros Kasampalis
Maria Kotsifakou
BP-M AND TILED-BP
Tiled BP
Tiled BP

- Reading boundary messages from memory
- Local computation on local data
- Writing the resulting boundary messages to memory
Tiled BP
BACKGROUND ON GPU
GPU Programming Model

- Number of threads and thread blocks is specified at kernel launch
- All threads execute the same kernel function
GPU Memory Model

Global Memory: accessible by all threads

Shared Memory: scratchpad memory, shared by threads within a thread block

Other components of the memory hierarchy are not shown (registers, constant memory, caches)

Threads within a thread block are cooperative and can synchronize. 
```cpp
__syncthreads()
```
Kernel Execution

Thread blocks are assigned to SMs. SMs contain simple processors with deep pipelines (throughput-oriented architecture).

An SM can accommodate multiple thread blocks simultaneously. The exact number depends on hardware restrictions.

A thread block resides in an SM until its execution is completed.
OUR METHOD AND EVALUATION
Tiled BP on GPU

A thread block for each tile
Tiled BP on GPU—A Big Picture
Tiled BP on GPU—Finer Granularity
Looking into a Tile

One thread per message vector element

Different Groups of threads in a thread block
Tiled BP on GPU—Finer Granularity
Looking into a Tile

The same for Up and Down
Optimization 1—Shared Memory

- Load data to shared memory at the start of local BP-M for each tile
  - Boundary messages
  - Data vectors of pixels
- Set reserved space in shared memory for other data in computation
  - Internal messages vectors
  - Outgoing boundary messages
Optimization 1—Shared Memory

- All data loading is coalesced
  - Row-wise storage of vertical and horizontal boundary messages for memory coalescing
- Tile at most 13 by 13 to accommodate all data in (48 KB) shared memory
  - At most 13x16=208 threads in a thread block
Optimization 1—Shared Memory

- With maximum tile size all shared memory storage is used for one thread block
- Given that, each SM can accommodate just one thread block
- Underutilizing SMs, but suitable for inter-block barrier synchronization
Optimization 2—Fast Global Barrier

• State-of-the-art GPU global barrier\(^1\)

• Requirements
  – One thread block per SM
  – Number of thread blocks at kernel launch equal to number of SMs

No need to launch multiple kernels, significantly reducing overheads

Manual scheduling of thread blocks on tiles

---

\(^1\) S. Xiao and W.-c. Feng, “Inter-block gpu communication via fast barrier synchronization,” in IEEE International Symposium on Parallel & Distributed Processing (IPDPS), 2010
Other Optimizations

- Fast and parallelized message calculation\(^1\)

- Manual analysis and tuning of the code
  - Removing some of "\_\_syncthreads" instructions

## Evaluation

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Hardware</th>
<th>Price (USD)</th>
<th>Tsukuba</th>
<th>Judging Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Exec. Time (ms)</td>
<td>Accuracy</td>
</tr>
<tr>
<td>BP-M ¹</td>
<td>CPU</td>
<td>$300</td>
<td>39,802</td>
<td>79.8</td>
</tr>
<tr>
<td>TiledBP ²</td>
<td>CPU</td>
<td>$300</td>
<td>1,585.85</td>
<td>82.1</td>
</tr>
<tr>
<td>TiledBP</td>
<td>GPU NVIDIA GTX 680 ³</td>
<td>$500</td>
<td>9.29</td>
<td>82.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TiledBP</td>
<td>GPU NVIDIA Tesla C2050 ³</td>
<td>$1350</td>
<td>7.96</td>
<td>82.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

¹ Given reference code on Intel® Xeon® E5-1620 @ 3.60GHz
² TiledBP CPU implementation on Intel® Xeon® E5-1620 @ 3.60GHz
³ GTX 680 with 8 SMs and Tesla C2050 with 14 SMs
⁴ Percentage of accurate depth labels compared to ground truth
Conclusion

• New GPU implementation of tiledBP for stereo matching
  – Wavefront computation
  – Inter-block GPU barrier synchronization

• Evaluation
  – Comparable accuracy
  – Comparable price
  – 200X speed up compared to CPU tiledBP
Thank You
Optimization 2—Fast Global Barrier

- One thread block per SM

- Number of thread blocks at kernel launch equal to number of SMs
  - Manual scheduling of thread blocks on tiles

Code snippet from:
S. Xiao and W.-c. Feng, “Inter-block gpu communication via fast barrier synchronization,” in IEEE International Symposium on Parallel & Distributed Processing (IPDPS), 2010