Accelerating ‘fields’ by revamping the Cholesky decomposition

Vinay Ramakrishnaiah
Raghu Raj P Kumar
John Paige
Dr. Dorit Hammerling

07/29/2015
Introduction

What is ‘fields’?

- Widely used R package for spatial statistics.
  - Literally thousands of users.
  - Developed by Dr. Douglas Nychka, director of The Institute for Mathematics Applied to Geosciences (IMAGe).
  - Major methods – thin plate splines, Kriging and Compact covariances for large data sets.
- ‘fields’ - single thread version on CPU.

http://burandtfieldmethods.blogspot.com/
Introduction

- Accelerated version of ‘fields’: fieldsMAGMA [Paige et al.]
  - SIParCS, 2014 work – accelerate on a Laptop.
  - Make it user friendly.

- MAGMA: Matrix algebra on GPU and Multicore Architectures
  - Aims to develop a dense linear algebra library similar to LAPACK but for heterogeneous/hybrid architectures. [http://icl.cs.utk.edu/magma/]

- Biggest computational problem in Kriging
  - Evaluation of data likelihood function given the covariance.
    - Requires Cholesky Decomposition
Motivation

- Unconventional behavior


http://www.aarp.org/money/investing/info-2014/5-bad-money-moves.html#slide1


http://www.kaizen-news.com/focus-performance/
Motivation

- Opposite impacts

http://www.bogotobogo.com/cplusplus/constructor.php
Motivation

- There is always more room!

http://funny-pictures.funmunch.com/funny-picture-545.html
Approach

1. Identify performance bottlenecks/cause for unconventional behavior
2. Analyze the existing code
3. See how to further accelerate the code
Performance analysis - Profiling

- NVIDIA visual profiler

- Why profile GPU?
- Why NVVP?
Performance analysis – Profiling dpotrf_m

<table>
<thead>
<tr>
<th>Time</th>
<th>Process *magmaOutput.exe...</th>
<th>[0] Tesla K20Xm</th>
<th>Context 1 (CUDA)</th>
<th>MemCpy (HtoD)</th>
<th>MemCpy (DtoH)</th>
<th>Compute</th>
<th>Streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.125 s</td>
<td>[1] Tesla K20Xm</td>
<td>Context 2 (CUDA)</td>
<td>MemCpy (HtoD)</td>
<td>MemCpy (DtoH)</td>
<td>Compute</td>
<td>Streams</td>
<td></td>
</tr>
<tr>
<td>11.15 s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.175 s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.2 s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.225 s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Non concurrent memory transfers

Computations

CPU involved in device to device data exchange
Performance analysis – Profiling dpotrf_mgpu

Asynchronous simultaneous data transfer

<table>
<thead>
<tr>
<th>Time</th>
<th>2.9 s</th>
<th>3.2 s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process <em>testing_dpotrf_mgpu</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Profiling Overhead</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[0] Tesla K20Xm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Context 1 (CUDA)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MemCpy (HtoD)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MemCpy (DtoH)</td>
<td></td>
</tr>
<tr>
<td>Compute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Streams</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[1] Tesla K20Xm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Context 2 (CUDA)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MemCpy (HtoD)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MemCpy (DtoH)</td>
<td></td>
</tr>
<tr>
<td>Compute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Streams</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Computations
Performance analysis - Timing

Comparison of dpotrf_m and dpotrf_mgpu called from the R wrapper

Time in seconds

Square matrix dimension N

Comparison of dpotrf_m and dpotrf_mgpu called from the R wrapper
Program Flow

1. R Program
   magmaChol(…)

2. fieldsMagma.r

3. Rwrapper.r
   dyn.load(…)
   .Call(…)

4. C function

5. MAGMA wrapper
   CUDA calls

6. Compute
Overheads in R

Sample square matrix size N=32768
OPTIMIZATION APPROACHES

- Reduce function call overheads in R.
- Optimize R environment for ‘fieldsMAGMA’.
- Accelerate the underlying C function.
Reducing R overheads

- Intel LD_PRELOAD
- Single dynamic load
- Move the dynamic loads to the highest level of R calls

R Program

magmaChol(…)

fieldsMagma.r

Rwrapper.r

dyn.load(…)

.Call(…)

C function

MAGMA wrapper

CUDA calls

Compute
Reducing R overheads

Reduced overheads

- Before
- After

Matrix dimension N vs. Time (s)
Performance of double precision Cholesky Decomposition on Intel Compilers

Comparison of accelerated code (double precision) - Intel 2012 vs Intel 2015 Compiler

- Intel 12.1.5
- Intel 15.0.3
For single precision calculations we obtained an improvement up to 30% for large sized matrices.
## Mysteries of Deep Copy

- Deep copy performed in R acts as a overhead.
- Used ‘perf’ to unravel the mystery!

### Abbreviations:
- SP – Single Precision
- DP – Double Precision
- DPCPY – Deep copy
- IP – In Place

<table>
<thead>
<tr>
<th></th>
<th>Deep Copy</th>
<th>In-place</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page Faults</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>LLC Misses</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>dTLB Misses</td>
<td>Constant</td>
<td>Constant</td>
</tr>
</tbody>
</table>

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DPCPY, 1GPU</td>
<td>DPCPY, 2GPU</td>
<td>IP, 1GPU</td>
<td>IP, 2GPU</td>
<td></td>
</tr>
<tr>
<td>Page Faults</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8192</td>
<td>65.177</td>
<td>82.212</td>
<td>64.091</td>
<td>81.593</td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td>67.151</td>
<td>84.652</td>
<td>65.686</td>
<td>83.194</td>
<td></td>
</tr>
<tr>
<td>24576</td>
<td>73.528</td>
<td>88.921</td>
<td>69.383</td>
<td>87.455</td>
<td></td>
</tr>
<tr>
<td>32768</td>
<td>80.518</td>
<td>97.482</td>
<td>75.359</td>
<td>91.349</td>
<td></td>
</tr>
<tr>
<td>LLC Misses</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8192</td>
<td>157.6</td>
<td>133.4</td>
<td>143.8</td>
<td>119.7</td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td>505.1</td>
<td>402</td>
<td>451.8</td>
<td>394</td>
<td></td>
</tr>
<tr>
<td>24576</td>
<td>1059.7</td>
<td>823</td>
<td>955.3</td>
<td>715</td>
<td></td>
</tr>
<tr>
<td>32768</td>
<td>1821</td>
<td>1414</td>
<td>1650.7</td>
<td>1222</td>
<td></td>
</tr>
<tr>
<td>dTLB Misses</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8192</td>
<td>7.8</td>
<td>6.6</td>
<td>8</td>
<td>6.5</td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td>11.8</td>
<td>8.4</td>
<td>11.6</td>
<td>7.6</td>
<td></td>
</tr>
<tr>
<td>24576</td>
<td>17.9</td>
<td>9.8</td>
<td>18</td>
<td>9.9</td>
<td></td>
</tr>
<tr>
<td>32768</td>
<td>22.6</td>
<td>12.7</td>
<td>23.4</td>
<td>12.7</td>
<td></td>
</tr>
<tr>
<td>HtoD time (msecs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8192</td>
<td>65.1</td>
<td>307</td>
<td>65.1</td>
<td>274.53</td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td>224.38</td>
<td>945</td>
<td>224.5</td>
<td>945.54</td>
<td></td>
</tr>
<tr>
<td>24576</td>
<td>508.3</td>
<td>2166.55</td>
<td>509.11</td>
<td>2162</td>
<td></td>
</tr>
<tr>
<td>32768</td>
<td>1159.2</td>
<td>3806</td>
<td>1159.55</td>
<td>3802</td>
<td></td>
</tr>
<tr>
<td>DtoH time (msecs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8192</td>
<td>45.1</td>
<td>85</td>
<td>45.1</td>
<td>84.9</td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td>169.23</td>
<td>366</td>
<td>169.3</td>
<td>365</td>
<td></td>
</tr>
<tr>
<td>24576</td>
<td>384.5</td>
<td>796</td>
<td>379.23</td>
<td>803</td>
<td></td>
</tr>
<tr>
<td>32768</td>
<td>2991.53</td>
<td>1422</td>
<td>2842.55</td>
<td>1419</td>
<td></td>
</tr>
<tr>
<td>Elapsed Execution time-5 trials average (secs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8192</td>
<td>0.88</td>
<td>0.5</td>
<td>0.89</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td>4.03</td>
<td>2.12</td>
<td>4.08</td>
<td>2.13</td>
<td></td>
</tr>
<tr>
<td>24576</td>
<td>10.95</td>
<td>5.37</td>
<td>10.74</td>
<td>5.38</td>
<td></td>
</tr>
<tr>
<td>32768</td>
<td>22.29</td>
<td>11.83</td>
<td>22.3</td>
<td>11.75</td>
<td></td>
</tr>
</tbody>
</table>
Results

Speed up of accelerated single precision versions with respect to CPU implementation

- SP_DCPY_nGPU1
- SP_DCPY_nGPU2
- SP_IP_nGPU1
- SP_IP_nGPU2

Speed up vs Square matrix dimension N
Results

Speed up of accelerated double precision versions with respect to CPU implementation

Square matrix dimension N

Speed up

DP_DPCPY_nGPU1

DP_DPCPY_nGPU2

DP_IP_nGPU1

DP_IP_nGPU2
Conclusion

- Achieved an improvement in performance of about 40% - 60% over the previous version of fieldsMAGMA for large matrices.
  - Ability to allocate pinned memory makes significant difference.
  - Reduce R overheads.
  - Fine-tuned the R environment for fieldsMAGMA.
- Detailed technical report.
Future work
References

- Tomov, S., Dongarra, J., Volkov, V., & Demmel, J. *MAGMA library*. University of Tennessee and University of California, Knoxville, TN, and Berkeley, CA.
Thank you!

- Questions?
Introduction

Cholesky Decomposition Times

- Time (Minutes)
- Number of Observations

- Default
- 1 GPU, with copy
- 1 GPU, in-place
- 2 GPUs, with copy
- 2 GPUs, in-place
Initial timing results

Comparison of testing_dpotrf & testing_dpotrf_mgpu

Square matrix dimension N vs. GFlops and time (s)

- testing_dpotrf_mgpu - Gflops
- testing_dpotrf - Gflop/s
- testing_dpotrf_mgpu - time (s)
- testing_dpotrf - time (s)
Single precision calculations

Using pinned memory for single precision

- Non-pinned
- Pinned

Square matrix dimension N vs. Time (s)
Comparison of execution times of different implementations

- DP_DPCPY_nGPU1
- DP_DPCPY_nGPU2
- DP_IP_nGPU1
- DP_IP_nGPU2
- SP_DCPY_nGPU1
- SP_DCPY_nGPU2
- SP_IP_nGPU1
- SP_IP_nGPU2

Square matrix dimension N vs Time (s)
Results

Performance improvement of the accelerated Kriging over CPU version

Speed up

Square matrix dimension N

Performance improvement of the accelerated Kriging over CPU version

- mKrig_1_DP
- mKrig_1_SP
- mKrig_2_DP
- mKrig_2_SP
Results

Performance improvement of the accelerated Kriging workflow over the CPU version

- mKrigWF_1_DP
- mKrigWF_2_SP
- mKrigWF_2_DP
- mKrigWF_1_SP

Square matrix dimension N

Speed up