CUDA-Accelerated Satellite Communication Demodulation

Renliang Zhao, Ying Liu, Liheng Jian, Zhongya Wang

School of Computer and Control
University of Chinese Academy of Sciences
Outline

- Motivation
- Related Work
- Multiple Symbol Detection (MSD) Algorithm in PCM/FM Demodulation
- Parallelization of MSD on CUDA
- MSD-based Demodulation on CUDA
- Performance Evaluation
- Conclusion
Motivation

A typical flow of digital communication in telemetry system of space exploration

- Demodulation is a critical step
Motivation

- Multiple Symbol Detection (MSD) is an important technique in digital communication
  - Estimate the sequence of the received signal by maximum-likelihood estimation
  - Correlate the subsequent 3 to 5 symbols of the received signal with all the possible transmitted waveforms
  - Determine the sign of the symbol by the correlation peak
Motivation

- MSD Algorithm in PCM/FM Demodulation

- Used in space TT&C (Telemetry, Tracking and Control), digital communication, etc.
Motivation

- MSD-based demodulation is a critical technique in space telemetry systems
  - Outperform other methods in Lower BER (Bit-Error-Rate) at high SNR (Signal-to-Noise Ratio)
  - Computational complexity is high, $O(Nl^2s/c)$
    - e.g. $N = 112$ million floats per second
    - a continuous data stream, possibly infinite
  - Serial MSD on a single PC or workstation takes 232 seconds (e.g. 10 microseconds bit stream with 56M sampling rate, 1M floats, 2M code rate, code length 28)
  - Real-time demodulation is in demand
- Parallel MSD is a promising solution
Motivation

- Explore the performance of MSD in telemetry on CUDA-enabled GPU
  - Identify the computation kernel, *sliding correlation*
  - Propose a *CUDA-based sliding correlation scheme*
  - Implement *sliding correlation* kernel
  - Apply various optimization techniques
  - Implement a CUDA-enabled MSD-based demodulation algorithm

- Experimental results present up to **52.8 x** speedup comparing with the execution time of the serial MSD on CPU

- Good scalability is observed
Related Work

- FPGAs are used to implement MSD-based demodulation
  - Expensive, specialized device, difficult to program
- M. Geoghgen et al. used Viterbi method to reduce the computation of MSD
  - Dynamic programming method
  - Complexity of the worst case, $O(Nl^2s/c)$
- Noncoherent MSD, Multiple Symbol Differential Detection, etc. are proposed
  - To solve many problems in digital communication
  - Complexity is not reduced
MSD Algorithm in PCM/FM Demodulation

- Estimate the symbol (1 or 0) of a number in the received signal by maximum likelihood principle
  - Enumerate all the reference sequences
  - Calculate the correlation between a segment of the received signal and each reference sequence
  - Determine a symbol by the symbols in the reference sequence which makes the largest correlation
MSD Algorithm in PCM/FM Demodulation

- **Problem Statement —— Sliding Correlation**
  - Correlation between a long sequence and a set of short sequences
  - Parallelism is high
    - Massive number of independent dot products

- **Dot product in CUDA SDK is not applicable**
  - Only support two vectors
  - Long vectors, bringing massive CUDA threads
Parallelization of MSD on CUDA

- **Two solutions**
  - **Scheme 1**
    - Each thread calculates a dot product for two vectors
    - Different threads in a block take different segments of the long vector
    - All the threads in a block share a common short vector
    - Repeat above steps for each reference sequence
Parallelization of MSD on CUDA

- Two solutions
  - Scheme 2
    - Each thread calculates a dot product for two vectors
    - Different threads in a thread block take a different short vector
    - All the threads in a block share a common segment of the long vector
Parallelization of MSD on CUDA

- Two solutions
  - Scheme 1
    - Massive threads and blocks, high thread-level parallelism
  - Scheme 2:
    - # threads in a block is limited by the number of short vectors, may hinder the thread-level parallelism
    - A total of $N-Is/c+1$ thread blocks may exceed the capability of CUDA
  - Scheme 1 won
MSD-based Demodulation

- CUDA-enabled MSD-based demodulation
  - For each segment of the received signal
    - Call *sliding correlation* kernel for in $l$ branch
    - Call *sliding correlation* kernel for in $Q$ branch
    - Calculate amplitudes
    - Select the largest amplitude and the index of the corresponding reference sequence
  - Overflow the shared memory
In order to reduce the usage of shared memory by each thread block

- Adopt *sliding correlation* kernel in an interleaved manner —— calculate the correlation between the given segment with its *I* reference sequence and *Q* reference sequence in turn
Performance Evaluation

- **Hardware platform**
  - A SuperMicro server, two 6-core 2.4 GHz Intel Xeon CPUs, 24G Memory
  - NVIDIA’s Tesla C2070 card with 448 1.15 GHz SPs and 6GB global/device memory

- **Software**
  - Windows Server 2008 R2 Enterprise
  - CUDA 4.0

- **Datasets**
  - 4 sets of simulated data generated by an space PCM/FM integrated baseband system
  - 10 microseconds, each set containing 1,075,200 floats
## Performance Evaluation

### Speedups

<table>
<thead>
<tr>
<th>Observation Length</th>
<th>Reference Sequence Length</th>
<th>Serial (ms)</th>
<th>CU-MSD (ms)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>28*3</td>
<td>1497</td>
<td>67</td>
<td><strong>22.3</strong></td>
</tr>
<tr>
<td>5</td>
<td>28*5</td>
<td>8407</td>
<td>195</td>
<td><strong>43.11</strong></td>
</tr>
<tr>
<td>7</td>
<td>28*7</td>
<td>46230</td>
<td>944</td>
<td><strong>48.97</strong></td>
</tr>
<tr>
<td>9</td>
<td>28*9</td>
<td>232927</td>
<td>4722</td>
<td><strong>49.3</strong></td>
</tr>
</tbody>
</table>

2M code rate
code length 28

<table>
<thead>
<tr>
<th>Observation Length</th>
<th>Reference Sequence Length</th>
<th>Serial (ms)</th>
<th>CU-MSD (ms)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>14*3</td>
<td>904</td>
<td>46</td>
<td><strong>19.65</strong></td>
</tr>
<tr>
<td>5</td>
<td>14*5</td>
<td>4,719</td>
<td>93</td>
<td><strong>50.74</strong></td>
</tr>
<tr>
<td>7</td>
<td>14*7</td>
<td>24,053</td>
<td>499</td>
<td><strong>48.2</strong></td>
</tr>
<tr>
<td>9</td>
<td>14*9</td>
<td>123,159</td>
<td>2,388</td>
<td><strong>51.57</strong></td>
</tr>
</tbody>
</table>

4M code rate
code length 14
## Performance Evaluation

### Speedups

<table>
<thead>
<tr>
<th>Observation Length</th>
<th>Reference Sequence Length</th>
<th>Serial (ms)</th>
<th>CU-MSD (ms)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>7*3</td>
<td>597</td>
<td>37</td>
<td>17.4</td>
</tr>
<tr>
<td>5</td>
<td>7*5</td>
<td>3,184</td>
<td>61</td>
<td>50.7</td>
</tr>
<tr>
<td>7</td>
<td>7*7</td>
<td>14,212</td>
<td>307</td>
<td>51.39</td>
</tr>
<tr>
<td>9</td>
<td>7*9</td>
<td>70,892</td>
<td>1,426</td>
<td>52.8</td>
</tr>
</tbody>
</table>

8M code rate  
code length 7

<table>
<thead>
<tr>
<th>Observation Length</th>
<th>Reference Sequence Length</th>
<th>Serial (ms)</th>
<th>CU-MSD (ms)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>6*3</td>
<td>529</td>
<td>31</td>
<td>17.06</td>
</tr>
<tr>
<td>5</td>
<td>6*5</td>
<td>2,431</td>
<td>50</td>
<td>48.6</td>
</tr>
<tr>
<td>7</td>
<td>6*7</td>
<td>10,752</td>
<td>224</td>
<td>48</td>
</tr>
<tr>
<td>9</td>
<td>6*9</td>
<td>49,079</td>
<td>1,053</td>
<td>47.2</td>
</tr>
</tbody>
</table>

10M code rate  
code length 6
Performance Evaluation

- Overall performance on GPU scales up on all data sets when varying observation length
- The best case, 52.8x speedup, data set 3 with 8M code rate, observation length 9
- Speedup goes up as observation length increases
  - More computation as the number of correlations and the length of the vectors increase
  - Better utilize stream processors (SPs) of GPU
Conclusion

- Identified the computational kernel of MSD algorithm, *sliding correlation*
- Proposed a CUDA-enabled scheme to accelerate *sliding correlation* kernel
- Implemented a MSD-based demodulation algorithm by adopting *sliding correlation* kernel
- Performance are evaluated on 4 sets of data generated by an aerospace PCM/FM integrated baseband system
- Good speedups are observed for all data sets
- Good scalability is observed when varying the computation

➢ *Funded by China Space*