APPLIED PARALLEL PROGRAMMING ON GPU

Assignment 2

Deadline: 28 November 2010 – 22:00

1. Software Requirement:

1. Microsoft Visual Studio 2005 or 2008,
2. CUDA SDK 3.x and Toolkit 3.x.

2. Prerequisites:

1. You may refer to your lecture notes and the textbook for the GPU programming concepts.
2. You may refer to the following Wikipedia webpage for a general overview of sparse matrices,:
   http://en.wikipedia.org/wiki/Sparse_matrix
3. You may refer to the following NVIDIA webpage for sparse matrix generation and multiplication applications on CUDA, with the technical report and the source codes (Recommended);
   http://www.nvidia.com/object/nvidia_research_pub_001.html
4. You may refer to the recent publication following the technical report in the following webpage;
   http://www.nvidia.com/object/nvidia_research_pub_013.html
5. You may refer to the following links for CUSP package;
   http://code.google.com/p/cusp-library/
   http://code.google.com/p/cusp-library/downloads/list

3. Assignment Requirements:

In the assignment, you will implement sparse matrix multiplication algorithms with CUDA.

You will test your algorithms on ***DIA***, ***CSR***, ***CSC*** and ***COO*** type sparse matrix kernels. For more detailed information on the matrix types, you may refer to the links that are provided in the previous section.

In your assignment, you are required to complete the following requirements:

1. In your assignment, you may use the matrix generation codes that are provided in the NVIDIA webpage or elsewhere. However, you ***can not use*** the matrix multiplication codes that are provided in these packages and papers.

2. In the first part of your assignment, you will introduce and discuss on 4 different matrix types for the sparse matrix representation in CUDA, by considering their advantages and disadvantage. Then, you will introduce your matrix multiplication solutions that will attack to their problems. You do not need to introduce a complete pseudo-code of your algorithm.

However;

2.a The files that include the implementation codes, must be explained briefly. For example, briefly explain the functionalities of the files that will run on GPU and/or CPU.

2.b Explain your implementation methodologies briefly, such as the management of the memory operations and the thread operations, the determination of the block numbers (if you use the block-matrix operations), etc.

3. In the implementation part, you should discuss the implementation results for

- different number of matrix elements, such that, the number of elements in the matrix will change from 500x500 to 5000x5000 in the increasing order of 500x500, by specifying the sparseness of the matrices using

  Sparseness = (Number of Non-zero Elements) / (Total Number of Elements)

- each sparse matrix type (on ***DIA***, ***CSR***, ***CSC*** and ***COO)***,
- different implementations with cache and without cache, using shared memory. (Bonus: If you use textures in addition to shared memory, you will get additional points.)

by considering memory and time constraints, and the cons and pros of your solutions.

Please introduce your solutions either with tables or figures that are the visualizations of the tables. Moreover, you are required to explain each of the table and the figure in your document.

For example, you may compare DIA type matrix multiplications for varying number of matrix elements using shared memory options. However, there is no restriction on the structures of tables or figures that you will use in your analysis. You should use your research skills in order to answer the following questions:

3.a How does the computational complexity change with varying number of elements and the sparseness for each matrix type?

3.b Which memory management technique is suitable in order to solve this computational complexity problem in order to decrease the running time?

4. If your implementation requires specific configuration settings (such as the configuration of path settings or 3rd party libraries), please indicate them in your document.

4. Submission and Grading Policy

Assignments will be submitted via METU-Online. Create a rar or zip archive containing plain source codes (*.cu and c++ files) and the document explaining your design. ***You are required to introduce your assignment with an assignment document. Therefore, please take about your documentation.*** As a reference, you may refer to the paper titled, "***High Performance Computing Via a GPU***", which is uploaded to *Metu Online / Lecture Notes / Reference_papers / CUDA_HPC.pdf*.

Please note that late submissions may not be accepted!

You are expected to work individually NOT in groups. You will also be expected to follow the academic integrity rules.

Policy for Copying: ***Passing the work of others (either from another student or a code on internet etc.) off as your own is a breach of academic ethics and also of the University's disciplinary rules. When you submit a work it automatically implies that you claim the ownership of the work.***

Note that METU is subscribed to some tools which allow cross checking of submitted works as well as checking with any work on internet or any university subscribed to the system. No exceptions will be allowed and any work found to be copied will result in failing the course.

Please send your questions about the assignment to meteozay@gmail.com or you may post a message to the forum on METU-Online.