Spatial Distance Histogram Processing of Large Scale Data

University of South Florida
CIS 4930 - Summer 2016
Programming on Massively Parallel Systems

Kevin Wagner (wagnerk1@mail.usf.edu)
Elijah Malaby (emalaby@mail.usf.edu)
John Casey (jcasey2@mail.usf.edu)

July 1, 2016

Contents

1 Abstract

Graphic Processing Units (GPUs) today have a great deal of global memory a programmer can use for whatever reason deemed necessary. But what if the project and data at hand is too large even for the 2GB of the GTX 960, the 6GB of the GTX 980 TI, or even 12GB of the TITAN X card? Here the goal is to design an implementation of memory management systems or mechanisms to artificially limit the GPUs global memory in an attempt to evaluate the effect of large data sets on the running time of a piece of kernel code.

Dividing and loading the data into global memory as blocks is unequivocally the easiest approach but, how the data is loading is going to be the challenging part of this experiment. Since data transfer from a host to a device's global memory is extremely costly, the efficiency of the data transfer is paramount.

To evaluate the kernel, the CUDA program will be run and the running time of the kernel will be plotted against data of different sizes.

2 Introduction

The objective of this project is to simulate the scenario where the data at hand is too large to all fit into global memory at once. This is a very large data set since some GPUs today come with up to 12GB of global memory on board. Loading the data is the trivial part of this project. The efficient loading of data is what this project is about.

The expectation of the project is for the running times to be slower but, how much is the mystery. Loading to global memory is, as stated before, and extremely time comsuming process. An efficient algorithm to load data into the artificailly small global memory is again paramount and the results will solely depend on how this algorithm is implemented.

3 Strategies and Algorithms

Starting with the code that was written for Project 2. Even with the best attempts in place the code was still 10x slower than what the Teaching Assistant(TA) had produced. Taking that into consideration, the project started with the TA's code in order to improve on that already optimized code. This code of course did not take into consideration if the data set was too large to fit into global memory of the GPU itself.

Firstly, the naïve solution was used. The data was loaded in global memory as blocks one after the other.

4 Code Examples

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

```
//Test code for listing package
#include <stdio.h>

int main(){
   unsigned int counter = 1;

   while(counter <= 10{
      printf("%u ", counter);
      ++counter;
```

```
    }

    puts("");
}
```

5 Results

This section highlights the results obtained from our experiments and the control data we used to compare our data to.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

5.1 Control

For a baseline to compare our experiments to we are using the data posted on CANVAS from the TA. Number of Atoms versus the Running time in seconds. See Table 1 for the times and Fig. 1 for a graphical representation of the data.

5.2 Experimental

This section has the results we obtained in the project when we assume that the global memory is not big enough to hold all of the data that we have to to offer. We have to feed in the data a little at a time (say 200MB).

5.3 Test Cases

Thes

6 Conclusion

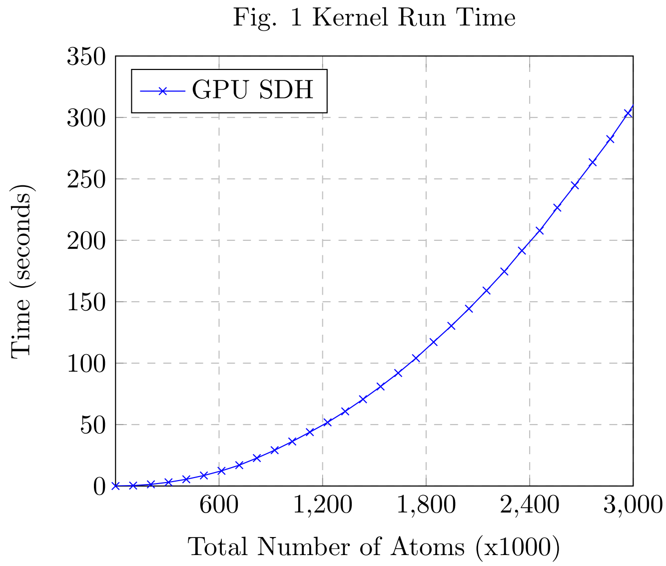
Go on about what we expected versus what we saw in testing. Does the loading of data the way we implemented it a success or failure? Go into some great detail why or why not.

7 References

1. NVIDA Corporation, CUDA Runtime API, 2015, http://docs.nvidia.com/cuda/cuda-runtime-api/index.html, [Online: accessed Summer 2016]

2. NVIDIA Corportion, CUDA Programming Guide, 2015, http://docs.nvidia.com/cuda/cuda-c-programming-guide/, [Online: accessed Summer 2016]

8 Figures



9 Tables

Table 1: Run times for contol experiment

Atoms	Time(s)	Atoms	Time(s)
512	0.00101494	1638912	91.99478912
102912	0.34378347	1741312	104.07542420
205312	1.36084175	1843712	117.17488860
307712	3.05233979	1946112	130.36553960
410112	5.48577261	2048512	144.33065800
512512	8.57051182	2150912	159.09375000
614912	12.34655285	2253312	174.61938480
717312	16.97405243	2355712	191.60043330
819712	22.75370407	2458112	207.86152650
922112	29.15684891	2560512	226.56158450
1024512	36.23521423	2662912	244.75129700
1126912	43.85413361	2765312	263.50692750
1229312	51.80871201	2867712	282.38305660
1331712	60.74640274	2970112	303.45535280
1434112	70.61425018	3072512	325.80584720
1536512	80.98942566		