Parallelization of Shortest Path Finder on GPU: Floyd-Warshall

Dhananjay Kulkarni
Department of Computer Engineering
GES RHS COE MSR
Nasik

Neha Sharma
Department of Computer Engineering
GES RHS COE MSR
Nasik

Prithviraj Shinde
Department of Computer Engineering
GES RHS COE MSR
Nasik

Vaishali Varma
Department of Computer Engineering
GES RHS COE MSR
Nasik

ABSTRACT

Till now serial computing has been used for performing all the tasks.But as the data for processing increases the load on the single node system also increase. This paper elaborates the use of parallel computing instead of the serial computing at times when the system has to process a large amount of data. The project deals with implementation of Floyd Warshall Algorithm i.e All Pair Shortest Path. This algorithm is implemented using parallel programming concept for faster solution. This is a research based project in which the serial and parallel computations are compared.Floyd Warshall algorithm has overcome the drawbacks of Dijkstra's and Bellman Ford Algorithm. For parallel programming, the project is implemented using NVIDIA GPU(NVIDIA GeForce 820M , 410M) for which CUDA(CUDA Toolkit 6.0) is used.[13] The purpose of developing this project is to find the shortest path between all the present nodes in a graph.This system is designed to work on a large dataset(set of 4 to 1024 nodes).This project can be implemented for Airline Systems,Transportation services, Courier Services, Networking.

General Terms:

Reduces the time of computation ,Parallel computation of the algorithm ,Guarantee of optimal solution.

Keywords:

Floyd Warshall Algorithm,Parallel Programming,CUDA,NVIDIA GPU.

1. INTRODUCTION

1.1 Parallel Computing

Traditionally Serial computing was widely used. In serial computing the execution was carried out serially i.e after completion of first instruction the next instruction was executed, this was time consuming process. To overcome this problem Parallel computing was introduced.Parallel computing is a form of computation in which many computations are carried out simultaneously. In this a large problem is divided into multiple sub problems and then executed simultaneously on the processor. This reduces the time complexity as compared to serial computation. In parallel computing more than one ALU is used hence the instructions are executed simultaneously. Parallel Computing mainly uses Flyn's Taxonomy from which it uses SIMD(Single Instruction Multiple Data).In this the instruction remains constant,but the data changes continuously. This increases the data handling capacity of the processor.

1.2 Floyd Warshall

Floyd Warshall algorithm is All Pair Shortest Path finder.It is mainly used to overcome the drawbacks of Dijkstra's and Bellman Ford Algorithm. It considers negative weight present in the graph. In Floyd Warshall algorithm every node of the graph is visited and the shortest path is computed.

2. PROPOSED SYSTEM

2.1 Graphical Processing Unit.

Graphics Processing units have evolved into a very attractive hardware platform for general purpose computations due to their extremely high floating point processing performance[3]. GPU-accelerated computing is the use of a graphics processing unit (GPU) together with a CPU to accelerate scientific, analytics, engineering, consumer, and enterprise applications[4].GPUs are used in computers, mobiles, game consoles. Modern GPUs are very efficient for high definition computer graphics.The highly parallel structure of GPU makes it far more efficient than general purpose CPU for processing of large data.The rapid evolution of GPUs in performance, architecture and programmability can provide application potential beyond their primary purpose of graphics processing[2]. The building blocks of GPU are Grids.Grids are further divided into blocks and the blocks are divided into threads. There are various parallel computing platforms like MPI, OpenCl, CUDA, this project is implemented using CUDA.

2.2 CUDA(Compute Unified Device Architecture)

CUDA was first introduced by NVIDIA in 2007. It was developed for both Windows and Linux platform later version 2.0 was compatible with Mac OS. NVIDIA has developed CUDA specially for NVIDIA Graphic Cards. It is an unique programming language for the NVIDIA Graphic cards as it uses the all the cores of the graphic card efficiently.

2.2.1 Programming Model. CUDA uses an extended C language that allows the user to program using the CUDA architecture.A user defined C function that is executed in the GPU is called a kernel. The number of times the kernel to be executed is decide by the programmer by providing the number of threads. So, if the user specifies the number of threads as N, the kernel will be executed N times by N different threads. The Kepler architecture also supports concurrent global kernel execution by allowing up to 16 kernels to execute simultaneously.

2.3 CUDA Memory

A large amount of time of computation is spent on reading the data. As the GPU has multiple threads it has to select the most appropriate memory for computation. The four types of memory are.

— Global Memory :

It is the largest memory present on the device. It is read and write type of memory. It the slowest memory.

— Shared Memory :

The Shared memory is read-and-write memory physically resides on the GPU. It is faster than the global memory. The threads present in the block are only allowed to access the memory.Thread in one block can access that block shared memory but threads in other block don't have access to shared memory in different blocks. This results in high speed as the thread access the shared memory simultaneously.The threads in the block communicate with each other using the shared memory. Shared memory is limited to 16 kilobytes per block.

— Texture Memory :

Texture memory is accessed using read only cache and it is used for performing floating point operations.It is very costly operation.

— Constant Memory :

Constant memory is read only memory and it does not change during the kernel execution.

3. SYSTEM FEATURES

The features of our system are as follows:

— Reduces the time of computation.

— Parallel computation of the algorithm.

— Guarantee of optimal solution.

3.1 Operating Environment

— NVIDIA GEForce GPU.

— Ubuntu 12.04

— CUDA Toolkit 6.0

— NSIGHT Eclipse.

3.2 Hardware Interfaces

— NVIDIA GEForce Graphics Card.

— Intel Core I3 4th Generation.

— 4GB RAM.

3.3 Software Interfaces

— Languages Used : CUDA,C.

— CUDA ToolKit 6.0.

— NSIGHT Eclipse Edition.

— UBUNTU 12.04.

— Documentation Tools :Dia,Tex Maker.

4. IMPORTANT MODULE AND ALGORITHM

The System is divided into three major modules :

—The algorithm implemented on CPU.
—The algorithm implemented on GPU.
—The graphical user interface(GUI).

4.1 Implementation on CPU

The algorithm is also implemented on CPU for computation of sequential time required for the algorithm.The algorithm is implemented using single thread.
The pseudo code is as follows:

```
for i = 1 to N
  for j = 1 to N
     if there is an edge from i to j

        dist[0][i][j] = the length of the edge from i to j
```

```
      else

         dist[0][i][j] = INFINITY
 for k = 1 to N

   for i = 1 to N

      for j = 1 to N

         dist[k][i][j] = min(dist[k-1][i][j], dist[k-1][i][k]
          + dist[k-1][k][j])
```

4.2 Implementation on GPU

The same algorithm is implemented on GPU for comparing its computation time with CPU and for calculating the speed-up and efficiency. The large matrix is subdivided into smaller matrix and threads are assigned to the matrix.Each thread executes sequential algorithm on each sub matrix simultaneously.

The code for allocation of threads and blocks areas follows:

```
CUDA_APSP(int *d_mat,int k,int N)
{
    int i = blockIdx.x * blockDim.x + threadIdx.x;
    int j = blockIdx.y * blockDim.y + threadIdx.y;
 if(i<N&&j<N){
int i0 = i*N + j;
int i1 = i*N + k;
int i2 = k*N + j;
if(d_mat[i1] != -1 && d_mat[i2] != -1)
d_mat[i0] =
(d_mat[i0] != -1 && d_mat[i0] < d_mat[i1] + d_mat[i2])
? d_mat[i0] : (d_mat[i1] + d_mat[i2]);
//__syncthreads();
}
}
```

4.3 Graphical user interface

Graphical user interface is designed using JAVA. GUI displays the general CPU and GPU configuration of the system.The block diagram of the system is also displayed.The graph that shows the comparison between the CPU and GPU computation is also displayed in the form of graph and also the table for comparison between computation value is also displayed.

—GPU has ability to process the information quicker.

—Save time and/or money: In theory, throwing more resources at a task will shorten its time to completion, with potential cost savings. Parallel clusters can be built from cheap, commodity components.

—Provide concurrency: A single compute resource can only do one thing at a time. Multiple computing resources can be doing many things simultaneously.

—Economic limitations - it is increasingly expensive to make a single processor faster. Using a larger number of moderately fast commodity processors to achieve the same (or better) performance is less expensive.

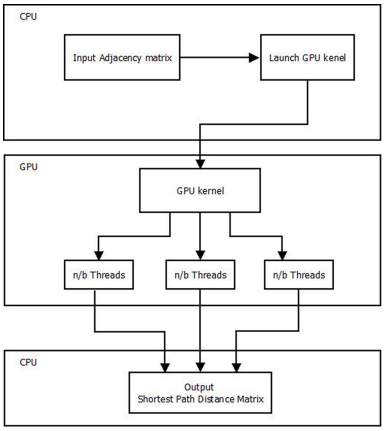
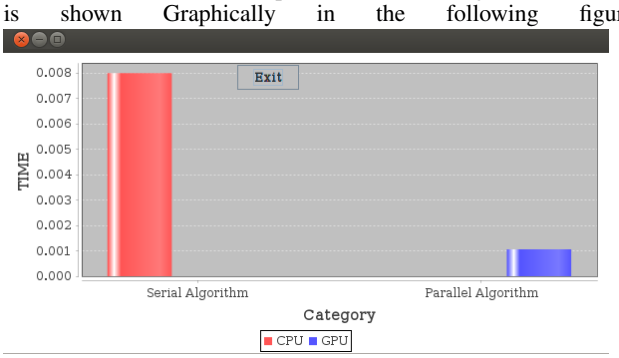


Fig. 1. Blockdiagram.

5. RESULTS

The computation time of Central Processing Unit(CPU) and Graphical Processing Unit(GPU) is shown Graphically in the following figure.



Graphical Representation.

The computation value of Central Processing Unit(CPU) and Graphical Processing Unit(GPU) is shown in the following table.

Matrix Size	CPU Time	GPU Time	Speed up
4	0.008	0.001062	0.067
16	0.078	0.001163999	0.188
32	0.543	0.002068	0.677
64	4.034	0.0011743	2.274
128	31.066	0.0048330000	7.082

6. CONCLUSION

Parallel computing is a emerging technology with a wide range of applications. We have applied parallel computing concept with the help of NVIDIA GPU. Also, We have compared different data sets for Floyd-Warshall Algorithm. In order to test our implementation we generated random matrix with increasing number of vertices, ranging from 4 to 1024. The GPU and CPU time for various data sets is calculated. The speed up gained by the GPU over CPU is also calculated and is displayed in tabular format and displayed graphically. Thus, this shows the efficiency of GPU over CPU on different data sets.

7. REFERENCES

[1] Jian Ma ; Sch. of Transp. Eng., Tongji Univ., Shanghai, China ; Ke-Ping Li ; Li-yan Zhang, A Parallel Floyd-Warshall algorithm based on TBB, IEEE.

[2] http://ati.amd.com/products/streamprocessor/index.html

[3] John D.owens, David Luebke, Naga Govindaraju, Mark Harris, Jens Kruger, Aaron E.Lefohn, Timothy. Purcell,A survey of general-purpose computation on graphics hardware, Computer Graphics Forum 34(March) (2007)80-113.

[4] Kairanbay Magzhan, Hajar Mat Jani"A review and evaluation of shortest path algorithms".

[5] Olaf Schenk,Matthias Christen,Helmar Burkhart"J. Parallel Distrib. Comput".

[6] http://www.nvidia.com/object/what-is-gpu-computing.html

[7] Efficient multi GPU algorithm for All pair shortest path.

[8] G. Venkataraman, S. Sahni, and S. Mukhopadhyaya, A blocked all-pairs shortest-paths algorithm, J. Exp. Algorithmics.

[9]P. Harish and P. Narayanan, Accelerating large graph algorithms on the GPU using CUDA, Lecture Notes in Computer Science

[10]A. Buluc, J. R. Gilbert, and C. Budak, Solving path problems on the GPU, Parallel Computing, vol. In Press, Corrected Proof, 2009.

[11]R. Seidel. On the all-pairs-shortest-path problem in unweighted undirected graphs. Journal of Computer and System Sciences

[12]U. Bondhugula, A. Devulapalli, J. Fernando, P. Wyckoff, and P. Sadayappan, Parallel fpga-based all-pairs shortest-paths in a directed graph, Parallel and Distributed Processing Symposium, International,

[13]International Journal of Computer Applications (0975 8887) Volume 110 - No. 16, January 2015.