# 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

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 . 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 48 or 72 or 100 cities).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. LITERATURE SURVEY

As mentioned earlier, a graph can be used to represent a map where the cities are represented by vertices and the routes or roads are represented by edges within the graph. In this section, a graph representation of a map is explained further, and brief descriptions and implementations of the three shortest path algorithms being studied are presented. The following algorithms are discussed below:

— Dijkstra's Algorithm.

Dijkstra's Algorithm is used to compute the single source shortest path.In this algorithm the minimum distance is computed from a single source to single destination.Dijkstra's algorithm does not support the use of negative weights in the graph,hence all the weights in the graph should have positive weight.Although Dijkstra's algorithm does not solve the all pair shortest path problem,it is possible to get the shortest distances for all pairs of vertices by running Dijkstra's algorithm for every vertex of the graph[2].

```
INPUT: A graph G(V,E) where V is a set of vertices
and E is set of weighted edges between these
```

```
vertices.
A source vertex from V.

OUTPUT: The distance of shortest paths between
the source vertex and every vertex in V.

for each vertex v in V
dist[v]=infinity
previous[v]=undefined
end for
dist[source]=0
Q=V
while Q is not Empty
u= vertex in Q with smallest distance in dist[]
Q=Q/{u}
if dist[u]=infinity
break

for each neighbour v of u in Q
alt = dist[u] + dist_between(u,v)
if alt < dist[v]
dist[v] = alt
previous[v] = u
decrease-key v in Q
end if
end for
end while
return dist
```

— Bellman-Ford Algorithm.

In comparison to Dijkstra's Algorithm, the Bellman-Ford Algorithm admits or acknowledges the edges with negative weights. That is why, a graph can contain cycles of negative weights, which will generate numerous number of paths from the starting point to the final destination, where each cycle will minimize the length of the shortest path[4].

```
INPUT: A graph G(V,E) where V is A set of vertices
and E is set of weighted edges between these
vertices.

A source vertex from V.

d[0] = 0;
for(int i=0;i<n-1;i++)
{
for(int j=0;j<n;j++)
{
if (d[a[[j][0]-1]< 30000)
{
d[d[j][1] - 1] =
math.min (d[a[j][1]-1],d[a[[j][0]-1]  = a[j][2]);
}
}
}
```

Table 1. Comparison of algorithms

| Algorithm | Description | Negative Values | Time Complexity |
|---|---|---|---|
| Dijkstra's Algo. | SSSP | NO | O($n^2 + m$) |
| Bellman Ford Algo. | SSSP | NO | O($n^2$) |
| Floyd Warshall Algo. | APSP | YES | O($n^3$) |

— Floyd-Warshall Algorithm.

Floyd-Warshall Algorithm is an all pair shortest path algorithm. As the negative values are not considered in Dijkstra's Algorithm it considers the negative weights in the graph. In this algorithm every node present in the graph is visited in order to compute the shortest path.

The serial computation of these algorithms is time consuming so to increase the efficiency of the result This algorithm is implemented using GPU. Due to this the computation time is reduced drastically. Floyd Warshall overcomes the disadvantages of Dijkstra's and Bellman Ford Algorithm so Floyd Warshall Algorithm is implemented parallely using CUDA on GPU.

Here 'n' is Number of Vertices and 'm' is Number of edges.
SSSP- Single source shortest path.
APSP- All pair shortest path.

## 3. PROPOSED SYSTEM

### 3.1 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.

### 3.2 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.

*3.2.1 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.

### 3.3 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.

## 4. SYSTEM FEATURES

The features of our system are as follows:

— Reduces the time of computation.

— Parallel computation of the algorithm.

— Guarantee of optimal solution.

### 4.1 Operating Environment

— NVIDIA GEForce GPU.

— Ubuntu 12.04

— CUDA Toolkit 6.0

— NSIGHT Eclipse.

### 4.2 Hardware Interfaces

— NVIDIA GEForce Graphics Card.

— Intel Core I3 4th Generation.

— 4GB RAM.

### 4.3 Software Interfaces

— Languages Used : CUDA,C.

— CUDA ToolKit 6.0.

— NSIGHT Eclipse Edition.

— UBUNTU 12.04.

— Documentation Tools :Dia,Tex Maker.

## 5. MODULES

The Project is divided into six modules.The modules are as follows:

— Read Data from file.

— Convert raw file data into matrix form.

— Launch GPU Kernel.

— Compute shortest Path.

— Return Output from GPU to CPU.

— Display output on CPU.

### 5.1 Read data from file

The input to the algorithm will be a large data set of 45 to 70 cities.The data from the file will be fetched and processed in the further modules.

### 5.2 Convert raw file data into matrix form

The input will be in the form of matrix hence, the raw data from the file will be converted in to adjacency matrix form using File handling.

### 5.3 Launch GPU Kernel

The algorithm will be computed parallely on GPU hence the data processed in the above modules will be given to GPU using the kernel function.In the Kernel function the Floyd-Warshall

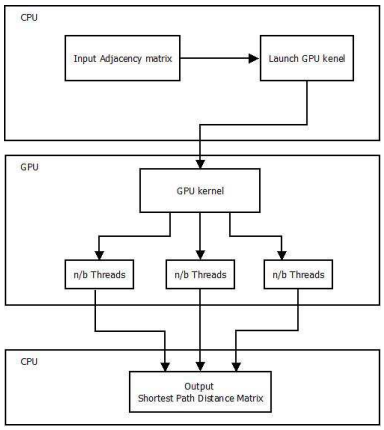


Fig. 1. blockdiagram

algorithm will be executed .

### 5.4 Compute shortest path

The shortest path will be computed with the help of the adjacency matrix passed from the CPU to GPU.The operation on the data will be performed parallely with the help of threads and blocks.

### 5.5 Return Output from GPU to CPU

After computing the shortest path using the Floyd-Warshall algorithm the result matrix will be again send from the GPU to CPU.The Computations are made on GPU and the final result are send to the CPU.

### 5.6 Display output on CPU

The output matrix which is send by the GPU to CPU is displayed on the CPU.

## 6. PSEUDO CODE

The Floyd-Warshall algorithm compares all possible paths through the graph between each pair of vertices. Consider a graph G(V,E) where V is no. of vertices and E is no. of edges. For computing minimum path between each pair of node Wk is computed where k ranges from 1 to V. For computing shortest from i to j, W[i, j] = W[i, k] +W[k, j]

INPUT: A graph G(V,E) where V is as et of vertices and E is set of weighted edges between these vertices. A source vertex form V.

OUTPUT: The distance of shortest paths between the source vertex and every vertex in G.

### 6.1 Code

Read the input from the provided file.
Convert the present data of file into Matrix form.
Launch kernel of GPU which will execute the Floyd Warshall Algorithm parallely.
Compute the shortest path
for all nodes
for all nodes
if there is an edge from one node to other node
dist[0][node1][node2] = the length of the edge from one node to other node
else
dist[0][node][node2] = INFINITY
for k = 1 to all nodes
for i = 1 to all nodes
for j = 1 to all nodes
dist[k][i][j] = min(dist[k-1][i][j], dist[k-1][i][k] + dist[k-1][k][j])
After all iterations the distance dist(N,i,j) represents the shortest distance between i and j.
Send final shortest path from GPU to CPU.
Display the shortest path on the CPU.

## 7. CONCLUSION

Parallel computing is an emerging technology with a wide range of applications.The concept of parallel computing is applied using NVIDIA GPU and applying Floyd-Warshall algorithm using NVIDIA's parallel computing platform CUDA(CUDA ToolKit 6.0). This is a research based project which compares the time complexity of serial as well as parallel algorithm. CUDA is efficient in using maximum cores of GPU, hence this is considered as the best parallel computing platform. Thus, the system implements Parallelization of shortest path finder algorithm: Floyd-Warshall on GPU. This system can be implemented in the areas where the operation is to be performed on large dataset and the computation time should be less hence to reduce the time of computation parallelization can be implemented using GPU.

## 8. 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,