Comparative Analysis of Algorithms to Discover Shortest Path in Social Networks

Rida Fatima
Department of Computer Science
National Textile University, Faisalabad, Pakistan

Muhammad Asif
Department of Computer Science
National Textile University, Faisalabad, Pakistan

Muhammad Kashif Hanif
Department of Computer Science
Government College University, Faisalabad, Pakistan

ABSTRACT
The huge growth and popularity of social media networks have created unprecedented research opportunities. Finding the affiliation networks and shared interest of user groups within the social network are important and well-studied problems. Graph algorithms provide a measure to characterize the social network structure. Bipartite graphs can be used as a representative model of these problems. The solution depends on efficient discovery of geodesic distance between any two random nodes. To this end, two algorithms are studied and parallelized for comparative performance analysis. In this paper we present the formulation of both algorithms on Graphic Processing Unit platform. The performance is compared on random-generated social network data-sets.

General Terms
Graphs, Algorithms, Floyd Warshall Algorithm, Tropical algebra, Matrix Multiplication.

Keywords
Social Network, Online Social network, Shortest Path, Graphs, Bipartite Graphs, APSP.

1. INTRODUCTION
Online social networks have emerged as a platform to share information in the last decade. Millions of users use social network sites for different purposes e.g. Facebook, Twitter etc. Social network has become popular for interaction, communication and collaboration among people. A social network site is web-based service that allows to create a profile within a restricted system, share with a list of further users, view and traverse their list of links and those set by others within the system [1]. There exist hundreds of social network sites which attract people based on their interest. These sites provide different technological features like chatting, mobile connectivity, blogging, and photo/ video-sharing [1, 2]. The tremendous growth of social data poses challenges to provide insights in social network behavior. In addition a rapid, proliferation of technologies such as mobile phones and medical sensors have amassed a wide variety of social data, much of it in real time. It can leverages mass outreach based on readily available information about preferences, Semantics and opinion about virtually any product, topic or trend. In addition to the shared contents, the structural properties of social network offers great source for designing advertisement campaigns promotions. A network structure can be represented as social graph in order to apply graph computational models.

One of the key aspects of network measurement is to discover goodies distance between two nodes in the immense graph. It is often termed as social distance or shortest path in different context. This enables exploration of community structure and influence of individual nodes among a community to discover an affiliation network in a large social graph. The idea to generate the affiliation network based on the social network is to find link among people from common group affiliation. Ultimately, such computing structural properties greatly benefit all kind of networks such as Sensor networks [3] physical networks[4] transportation and internet of things. However given the size of prevalent social graph such as Facebook etc. Calculating all pair shortest path is computationally expensive and often limiting its application.

It is well known that an affiliation network can be modeled as bipartite graph [5, 6]. In affiliation networks, nodes are partitioned in two disjoint subsets such that nodes in one subset represents individuals and nodes in other subset represents affiliation. Nodes in these subsets are connected to each other by edges.

The size of these graphs makes their analysis enormously challenging. There exist different methods to calculate the shortest path from the huge dynamic social network by different researchers. They vary in term of performance and scalability. However, the well-known efficient algorithms can even become time-consuming on series computing platform. Computing shortest paths is important to find the user interactions[3,4].

The emergence of parallel and distributed architecture has provided new approaches for large scale data processing and analysis. These trends dramatically broadened the scope of computational social system research. Modern Graphics Processing Units (GPUs) are becoming increasingly popular as low cost parallel processer to platform computation intensive tasks. GPUs offer additional computational power instead of CPUs and are employed for over-all computations to attain high speed-ups in numerous areas such as computational biology, simulations, and scientific computing. The programming model for GPUs called Compute Unified Device Architecture (CUDA) permits programmers to write C-like programs with some extensions. CUDA offers a three features first one is hierarchy of thread groups, second one is shared memories, and the last one is barrier synchronization to accelerate the applications [9].

Many algorithms to compute shortest path in a graph exist. They vary in terms of performance and scalability. The objective of this work is to investigate the all-pairs shortest path problem in social network using Floyd-Warshall and Torgasin and Zimmermann algorithms [10]. These Algorithms will be implemented using GPUs to attain high speed-up.

1.1 Social Networks

Online social media have increased amazing worldwide development and fame. Online social network enable the people to make the profile to communicate, to interact and share data among people. Social networks are infinitely dynamic objects; when adding the new nodes, edges and implying the presence of new connections can produce and quickly modify the social network. A social network is a set of individuals interrelated by social links, e.g. contacts between friends or family members. In the social network set of nodes indicates individuals are connected by edges where edges represent the relation between individuals.

The term social network was introduced by Mattia Lambertini. He describes this term as: "A Social Network is a graph (V, E) where V is a set of individual and E ⊆ V × V is a set of social connections" [14].There are many social networks like Facebook, twitter, Myspace etc. When the profile or connection is established the new user can search any other network of his/her connection and make many other connections according to their desire and need. Although the fact that with time all eras have come to hold the changes of the social network. According to the researchers study online social sites have great impact of the lives of youth. A social networks are also called social websites [15].

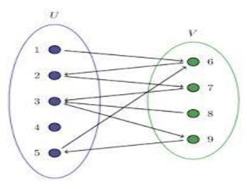
Many problems are occurring due to constantly increasing size of graphs, such as betweenness centrality computation, node separation, and shortest path, which scales desperately with graph or network size. The size of these gigantic graphs makes their analysis enormously challenging, as even efficient algorithms can become time-consuming. Current serial algorithms can only just be used due to space and time constraints. These confines motivated us to use the parallel algorithms for Social network graphs [16].

2. BACKGROUND

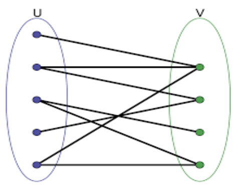
2.1 Bipartite Graph

A graph G is a data structure containing of vertex set V and edge set E. Bipartite is a graph whose vertices are two disjoint sets of U and V and the vertices of set U are connected to the vertices of set V by using the edges and vice versa.

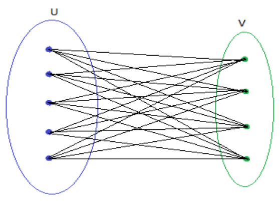
Graphs have been widely used to solve different real world problems. Adjacency matrix or adjacency list are used to store the graph based on graph density



a) Directed bipartite graph



b) Undirected bipartite graph



c) Complete undirected graph

Figure 1: Bipartite Graphs.

A weighted graph is a graph which has weights associated with edges. Weighted graph can be represented as a weight matrix W= (W_{ij}), this weight entry W_{ij} is

$$W_{ij} = \begin{cases} \infty & \mathit{if\ there\ is\ no\ edge\ between\ Vi\ and\ Vj\ and\ i \neq j} \\ 0 & \mathit{if\ i = j,} \\ \mathit{weight,} & \mathit{otherwise} \end{cases}$$

A graph whose vertex set can be divided into two non-empty disjoints subsets is called bipartite graph. Each single edge in one subset attaches to a vertex in the other. A bipartite graph can be directed or undirected. A complete bipartite graph is a graph such that every pairs of vertices in two sets are adjacent [15].

The matrix for weighted bipartite graph G = (V, E) with the vertex set V = V_1 U V_2 is

$$W = \begin{pmatrix} U1 & W1 \\ W2 & U2 \end{pmatrix}$$

Where $n_1 = |V_1|$ and $n_2 = |V_2|$, W_1 and W_2 are $n_1 \times n_2$ and $n_2 \times n_1$ matrices, respectively. U_1 and U_2 are tropical identity matrices of size $n_1 \times n_1$ and $n_2 \times n_2$.

$$U_1 = \begin{pmatrix} 0 & \infty & \dots & \infty \\ \infty & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \infty \\ \infty & \cdots & \infty & 0 \end{pmatrix}$$

2.2 Shortest Path Algorithms

Finding shortest path in graph is a well-studied problem and has countless applications. Researchers have proposed variety of algorithm to find shortest path [20, 8, 7 ,21]. They vary on computational complexity and structure of graph. The existing algorithms that are used for shortest path are Breath First Search Algorithm, Dijkstra and Floyd Warshall Algorithm.

Breath First Search (BFS) has the O (|V|+|E|) and O ($|V|^2$) time and memory complexity wherever |V| is the amount of vertices and |E| is the amount of edges in the graph. Breadth First Search take minutes to compute so it is not efficient. After that Dijkstra's algorithm is used. It works on a weighted graph G= (V, E) in which all edge weights are non-negative. Then the concept of Floyd–Warshall algorithm is introduced.

2.2.1 Floyd-Warshall Algorithm

Floyd-Warshall algorithm was familiarized by Floyd [17]. And then in 1962 to resolve the problem of all-pairs shortest path for weighted graph this algorithm was extended by Warshall [18].Additional important feature of algorithm is to find transitive closure for a graph. Weighted graphs are defined by the weights matrix M= (m_{pq}), which is similar to adjacency matrix. It holds the weights m_{pq} for the parallel edges:

$$D_{pq}^{(k)} \begin{cases} W_{pq} & \mathit{if\ k = 0,} \\ \mathit{Min}\ (d_{pq}^{(k-1)}, d_{pk}^{(k-1)} + d_{kq}^{(k-1)}) & \mathit{if\ k \geq 1.} \end{cases}$$

Above algorithm follow the pattern of dynamic programming (DP) and contains of more than a few steps. Each step updates the entire dynamic programming matrix. The Dynamic Programming matrix D is modified by the weight matrix of the graph.at all steps of k, D_{pq} keeps the lowest weight of the path from the source to destination vertices. It uses the intermediate vertices from source vertices to destination vertices. After performing all steps, the matrix holds the weight of the shortest paths among all pairs of the vertices from source to destination vertices.

Algorithm 1 Floyd-Warshall Algorithm

Given: Graph G= (V, E) with the weight matrix M

```
1.  D=M
2.   for k = 1 to |V | do
3.      for p = 1 to |V | do
4.           for q = 1 to |V | do
5.                $D_{pq} = \min (D_{pq}; D_{pk} + D_{kq})$
6.           end for
7.      end for
8.  end for
9.  return D
```

An algorithm for discovering shortest paths in a weighted graph with positive or negative edge weights only detecting the Presence of a Negative Cycles but it is also not efficient. Which can produce shortest paths for single source shortest paths in $O(n \log n + m)$ time, and shortest-paths for all pairs of nodes in $\Theta(n^3)$. For small graphs it is acceptable but, the computation required for a single node distance, But for a large million-node graph can take up to a minute on modern computers. By giving the prohibitively high costs of storing pre-computed distances, researchers have little choice but to sample portions of the graph or seek approximate results. The computational complexity of above mentioned algorithm are high.

3. RELATED WORK

In 2002, Vana, Dimitrios, and D. Zeinalipour-Yazti devote their time for searching and recovering the correct facts in peer to peer network. In their work they used the Breadth first search algorithm. BFS mechanism that allows to search with key words and also decrease the amount of messages to explore the network. Here they used the entirely distributed procedure for addressing the recovery problem. The upgraded BFS mechanism, is an extension lead of Gnuttela protocol that permits searching by using the key words and reduces the amount of messages that are required for explore the network. BFS device is locally run on every peer and robotically choose a casually subgroup of peer neighbors to forward every single query message. The Intelligent Search mechanism was also used to increase the competence of the search.

In paper " (On the bias of BFS (Breadth First Search)" [22]. According to Maciej Kurant, Athina and Thiran in the large networks where thousands of nodes and edges are included like peer to peer network (P2P), Online social network (OSN) and internet, as compare to further graph traversal methods such as Depth-First Search (DFS), Snowball Sampling (SBS), Breath first search (BFS) and Forest Fire (FF), BFS is very biased to find out the high degree nodes. BFS is famous and extensively used algorithm for large sampling network because it is very relaxed to understand and one other significant purpose is that half-finished BFS gathers the whole view of all nodes and edges of the specific selected zone of the graph. In their work authors also defined that in some conditions BFS fails. They also explain fact that in the measurement of Facebook [23], where crawlers found average node grade is k approximately is 324, while its actual value is k = 94 that is about 3.5 times lesser. This make it worst case and make the BFS far from the popularity. But in this research the goal is not about the popularity here the discussion about the bias of BFS. BFS have a tendency to learn high degree nodes first when crawling large networks and undirected networks.

According to Salvatore A. Catanese, Pasquale De Meo, Ferrara, Fiumara and Provetti in 2011, they work on online social network like one of network is Facebook to discover its friendship graph. In their work they considered Facebook, a massive online social network where they collect more than 500 million nodes and edges. Nodes represents the users and for showing its relationship uses the edges. Data related to users and edges that show relationships between users are not openly available, so to extract a valuable sample from web data they use some techniques. To find out the friendship graph among users two techniques are used i-e., Breath First search and uniform. Since BFS algorithm is well known to provide the results even if graph is not visited completely and if incomplete dataset is available [24].

According to Lijuan Luo, Wong and Wen-mei Hwu, in their paper "An Effective GPU Implementation of Breadth-First Search" they use the Breadth First search (BFS) algorithm with the graphic processing unit (GPU) to achieve the better speedup in many applications of electronic design automation (EDA) and many other fields. In their paper they explained that previously they used the BFS with the CPU implementation. But the parallel execution of the GPU can easily achieve the ten to hundred time's better speedup as compared to CPU. In their methodology they used the classified queue management method and a three level kernel arrangement approach. Hierarchical kernel arrangement is used to decrease the synchronization overhead. The BFS process broadcasts from all the boundary vertices parallel at the current level. They perform their experiments on a host device that equipped with dual socket, dual core 2.4 GHz Opteron processor and 8 GB of memory. In their work they used the single NVIDIA GeForce GTX280 GPU to run CUDA requests. In the following tables show the results after the implementation of BFS with the GPU

According to the author, Dijkstra algorithm is also castoff to estimate the shortest path tree inside each area of the network in OSPF- Open Shortest Path First. In Internet direction-finding Open Shortest path first is used. In this work author implement the Dijkstra algorithm and for implementation required a sample. So firstly Author consumes a link-state in the discrete areas that planned the hierarchy. After making the hierarchy they implement the Dijkstra algorithm and obtained the satisfactory results [27].

Himanshu Garg and Paramjeet Rawat in 2012, [28] proposed that Floyd-Warshall algorithm is easy to understandable and widely used algorithm to compute the quick results between all pairs of nodes that contains weighted and directed graph and also perceive the negative cycles if it is exist in the graph.

Many researchers have given many techniques for finding the shortest path for each pair of nodes to all other nodes of the graph but they used the complex data structure to reduce the complexity. In their paper Authors use the Floyd-Warshall algorithm for shortest path problem with reduced time complexity. In Floyd-Warshall they do not used the complex data structure. According to authors their proposed methodology provide the optimal results within small period of time and its time complexity is O (n^3).

In 2012, according to Asghar Aini and Amir Salehipour Floyd-Warshall algorithm is castoff to discover out the shortest path that contains least cost to discover the shortest path between every pairs of nodes in the graph. The network that contains at least one positive or negative cycle Floyd-Warshall algorithm compute its best result. In this work authors provide another algorithm that provides much better result instead of Floyd-Warshall algorithm. Rectangular algorithm, improvement of Floyd-Warshall algorithm. Here the performance of both algorithms are same but due to the small calculation of the Rectangular algorithm it computes the result faster than the Floyd-Warshall algorithm [29].

In the work of P.Swarnambigai1 and Dr A.Murugan2 they practice the Floyd-Warshall algorithm to check the all pair shortest path in DNA sequence. Researcher have gave many techniques that reduce the complexity to find the shortest path and then compare it with Floyd-Warshall algorithm. In their work they run algorithm on i3 machine that equipped with 2 GB Ram and it is executed in Java 2.0 and it provides best outcomes [30].

4. METHODOLOGY

GPU is a vector processor. Simulate the social network to characterize the performance of the algorithms on variety of graphs with respect to various structural properties. For this, social network is modeled and full bipartite unweighted graphs are generated by using the R studio and implement the shortest path algorithms using matrix product on GPU.

The small computational complexity of BIPARTITE APSP has motivated as to estimate and analyze the performance in context of affiliation networks. The shortest path from node I to j is given by $d_{ij}^{(n)}$. The complexity of APSP problem was improved by many researchers using different techniques and hardware architectures[46,49,50-56]. Romani[40] formulated the matrix product base version of Floyd-Warshall algorithm to solve the APSP problem using tropical algebra. The tropical matrix product of weight matrix with itself n-1 times provides the key of APSP problem. Moreover, frequently squaring techniques decreases the time complexity to O ($n^3 \log_2(n)$).

$$D = \underbrace{M \Theta M \Theta \ldots\ldots M}_{n\text{ -1 times}} = M^{\Theta\, n-1}.$$

Torgasin and Zimmermann[19] proposed on APSP algorithm for the bipartite graph using tropical matrix multiplication. The algorithm BIPARTITE APSP taking two blocks of the weight matrix as input. The dynamics programming matrix D is prepared with $W_1 \Theta\, W_2\, \Theta\, U_1$. Four blocks of weight matrix are calculated by multiplying matrix D with itself n1- 1 times. The computational complexity of this algorithm is O ((n^3) $\log_2(n)$).This algorithm is at least eight times quicker and needs four times less space in worst case when compared with Floyd-Warshall algorithm[20]. Later on, this algorithm was implemented on GPU[21].

Algorithm 2 Bipartite APSP

Require: two blocks of the weight matrix: M1; M2

Ensure: four blocks of $M^{\Theta 2|V1|}$: $L_1^{(2|V1|)}$, $L_2^{(2|V1|)}$, $M_1^{(2|V1|)}$, $M_2^{(2|V1|)}$

```
1. D = D1 = (M1 _M2 _ U)          // DP matrix initialization
2. for k = 2 to |V1|-1 do
3.     D = D Θ D1
4. end for
5. $L_1^{(2|V1|)}$ = D ΘD1
6. $L_2^{(2|V1|)}$ = M2 Θ D Θ M1 ⊕ U
7. $M_1^{(2|V1|)}$ = D Θ M1
8. $M_2^{(2|V1|)}$ = M2 Θ $L^{(2|V1|)}$
9. return $L_1^{(2|V1|)}$ , $L_2^{(2|V1|)}$ , $M_1^{(2|V1|)}$ , $M_2^{(2|V1|)}$
```

5. EXPERIMENT

Firstly Social network is generated by using the R studio and its version is 0.98.1102. For achieving the bipartite graphs as social network, two packages are loaded igraph package and MASS package. The package igraph is used for network analysis and visualization and Mass package is used for support functions and Datasets for Venables and Ripley's MASS. For the bipartite graph here used the make_full_bipartite_graph function that helps to create the full bipartite graph. For the adjacency matrix get. Adjacency function is used in it. A full bipartite 32 unweighted graph is shown given below.

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Figure 2: Full Bipartite 32 Unweighted Graph.

After creating the bipartite graph implementing the Algorithms on it that are given below and exposed the results. The most widely used algorithm to discover the shortest path among all pair of vertices is Floyd-Warshall algorithm with time complexity O (V^3)

The Floyd Warshall algorithm and APSP Bipartite have been considered.

- Floyd: serial implementation of Floyd Warshall on Intel CPU.
- Bipartite: serial implementation of Bipartite graph in Intel CPU

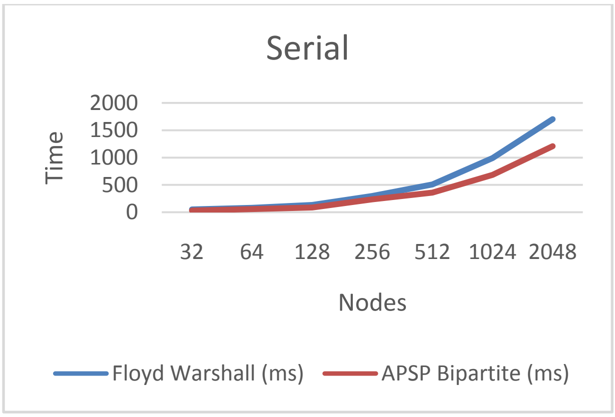


Figure 3: Running time of Floyd Warshall algorithm and APSP Bipartite on Intel CPU.

Using the speed-up and running time to find out the performance of the algorithms. Randomly generated the directed unweighted graph. Firstly, implement the Floyd Warshall algorithm and all pair shortest path Bipartite on Intel CPU for serial implementation. From the figure results shows that when nodes are minimum both Floyd Warshall and Bipartite works better. With the passage of time when nodes are increases Floyd Warshall algorithm takes more time as compare to Bipartite. So from the results it verifies that Bipartite is better than Floyd Warshall.

Here considered the Floyd Warshall algorithm and Bipartite APSP algorithm for implementation.

- Floyd: Parallel execution of Floyd Warshall algorithm on GPU.
- Bipartite: Parallel execution of Bipartite APSP on GPU.

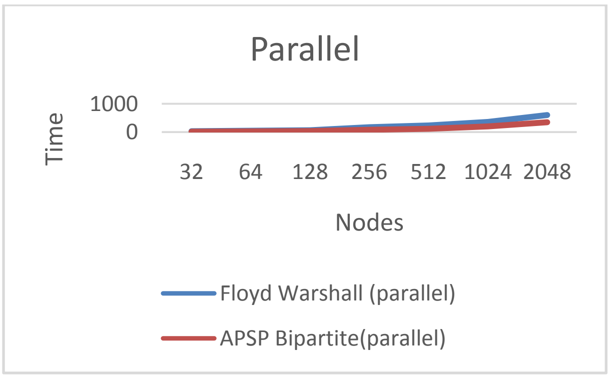


Figure 4: Comparison of Floyd Warshall and BipartiteAPSP algorithms for bipartite graphs on NVIDIA.

For better performance runtime and speed-up is considered. Randomly graphs are created. The runtime is be around over ten executions. In experiment we have considered worst case that is $n_1 = n_2$. Here the runtime of the Bipartite all pair shortest path algorithm and Floyd Warshall algorithm have been calculated. From the figure concluded that Bipartite APSP algorithm is work well than Floyd Warshall algorithm. According to the theory, Bipartite APSP algorithm is eight times better in a worst case.

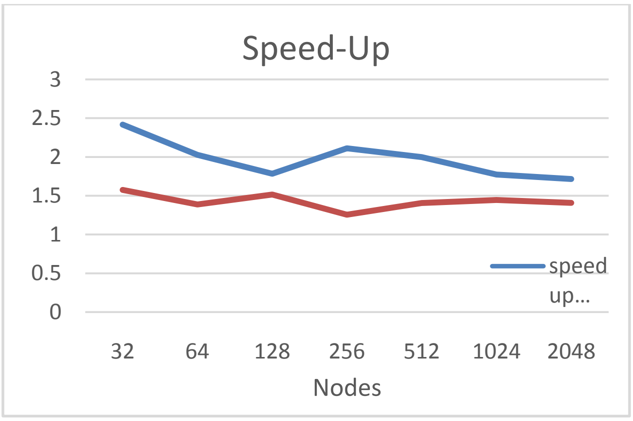


Figure 5: Speed-up of BipartiteAPSP over Floyd Warshall algorithm for bipartite graphs

6. CONCLUSION

The speed-up is achieved with the GPU based all pair shortest path executions for bipartite graphs when it is compared with the serial bases Floyd Warshall have been considered. From the speed-up it is proved that Parallel bipartite performs better than the Serial Floyd Warshall algorithm. According to the theory when the speed-up is greater than 1 its show that it has improved performance, it provides shorter execution time and higher throughput. Matrix- matrix product based APSP time complexity is O ((n^3) $\log_2$ (n)). Its time complexity indicates that Floyd Warshall is in worst case eight times slower than the bipartite APSP algorithm.

7. FUTURE WORK

In this work publicized the performance of two dynamic programming algorithms based on a single GPU. The performance improvement in as many GPU environment should be studied. The parallel formulation of biological networks. The long term determination of biological has two next aspects. In a practical life it helps to boost the competence of the computations of the biological network; this ultimately saves the time, finances and laboratory resources. In the scientific context one is gaining knowledge of the mechanisms of the life.

8. REFERENCES

[1] D. M. Boyd and N. B. Ellison, “Social Network Sites: Definition, History, and Scholarship,” *J. Comput. Commun.*, vol. 13, pp. 210–230, 2007.

[2] V. Batagelj and A. Mrvar, “Pajek – program for large network analysis,” *Connections*, pp. 47–57, 1998.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks : a survey,” vol. 38, pp. 393–422, 2002.

[4] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities Biophysics : Hopfield I T ., V .,” vol. 79, no. April, pp. 2554–2558, 1982.

[5] M. E. J. Newman, D. J. Watts, and S. H. Strogatz, “Random graph models of social networks,” vol. 99, 2002.

[6] M. Tavakolifard, *Mozhgan Tavakolifard On Some Challenges for Online Trust and Reputation Systems*, no. August. 2012.

[7] M. Potamias, F. Bonchi, C. Castillo, and A. Gionis, “Fast

shortest path distance estimation in large networks," *Proc. 18th ACM Conf. Inf. Knowl. Manag.*, p. 867, 2009.

[8] X. Zhao, A. Sala, H. Zheng, and B. Y. Zhao, "Efficient shortest paths on massive social graphs," *7th Int. Conf. Collab. Comput. Networking, Appl. Work.*, pp. 77–86, 2011.

[9] J. Sanders, "CUDA by example: an introduction to general-purpose GPU programming," p. 290, 2011.

[10] S. Torgasin, "Graph-based methods for the design of DNA computations," 2012.

[11] B. F. S. Revisited, "CS2 Algorithms and Data Structures Note 11 Breadth-First Search and Shortest Paths," no. February, pp. 1–10, 2005.

[12] R. La, G. F. Italiano, D. Informatica, S. Produzione, and R. Tor, "Speeding Up Dijkstra ' s Algorithm for All Pairs Shortest Paths ∗ Dipartimento di Informatica e Sistemistica An all pairs variant of Dijkstra ' s algorithm," pp. 1–7.

[13] A. B. Sadavare and R. Kulkarni, "A Review of Application of Graph Theory for Network," *Ijcsit.Com*, vol. 3, no. 6, pp. 5296–5300, 2012.

[14] and C. T. Mohar, Bojan, "No Title," *Graphs on surfaces.*, vol. JHU PRESS , 2001.

[15] L. W. Beineke, F. Harary, and J. W. Moon, "Cambridge Philosophical Society," vol. 60, pp. 1–5, 1964.

[16] B. J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU Computing," 2008.

[17] R.W. Floyd, "No Title," *Algorithm 97 Shortest path*, vol. 5, no. 6, p. 345, 1962.

[18] S. Warshall, "No Title," *A theorem boolean matrices*, vol. 1, no. ACM, 9(1):11, p. 12, 1962.

[19] D. Der Naturwissenschaften, "Graph-based Methods for the Design of DNA Computations Svetlana Torgasin aus," 2012.

[20] S. Torgasin and K. Zimmermann, "An all-pairs shortest path algorithm for bipartite graphs," vol. 3, no. 4, pp. 149–157, 2013.

[21] M. K. Hanif, "Mapping Dynamic Programming Algorithms on Graphics Processing Units Muhammad Kashif Hanif," 2014.

[22] M. Kurant and P. Thiran, "On the bias of BFS (Breadth First Search)."

[23] M. Gjoka, U. C. Irvine, C. T. Butts, U. C. Irvine, and U. C. Irvine, "Walking in Facebook : A Case Study of Unbiased Sampling of OSNs."

[24] S. A. Catanese, P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti, "Crawling Facebook for Social Network Analysis Purposes," pp. 0–7, 2011.

[25] L. Luo and W. Hwu, "An Effective GPU Implementation of Breadth-First Search ∗," pp. 52–55, 2010.

[26] J. Lee and J. Yang, "A Fast and Scalable Re-routing Algorithm based on Shortest Path and Genetic Algorithms 1 Introduction Related Work : Genetic Algorithm for Shortest Path Problem," vol. 7, no. 3, pp. 482–493, 2012.

[27] A. Roozbeh, "Resource monitoring in a Network Embedded Cloud Resource monitoring in a Network Embedded Cloud."

[28] H. Garg, "An Improved Algorithm for Finding All Pair Shortest Path," vol. 47, no. 25, pp. 35–37, 2012.

[29] A. Aini and A. Salehipour, "Speeding up the Floyd – Warshall algorithm for the cycled shortest path problem," *Appl. Math. Lett.*, vol. 25, no. 1, pp. 1–5, 2012.

[30] P. Swarnambigai, A. Govt, and A. College, "An Efficient algorithm to compute all pairs shortest path using DNA sequence," pp. 1–4.

[31] B. Ca, "On the All-Pairs-Shortest-Path."

[32] F. F. Dragan, "Estimating all pairs shortest paths in restricted graph families : a unified approach ☆," vol. 2204, no. June 2001, pp. 1–21, 2005.

[33] S. Hougardy, "The Floyd-Warshall Algorithm on Graphs with Negative Cycles," vol. 110, pp. 279–281, 2010.

[34] L. Roditty, "All-Pairs Shortest Paths with a Sublinear Additive Error."

[35] P. Harish and P. J. Narayanan, "Accelerating Large Graph Algorithms on the GPU Using CUDA," pp. 197–208, 2007.

[36] G. J. Katz, "All-Pairs Shortest-Paths for Large Graphs on the GPU All-Pairs Shortest-Paths for Large Graphs on the GPU," pp. 47–55, 2008.

[37] T. Okuyama, F. Ino, and K. Hagihara, "for Finding All-Pairs," 2004.

[38] T. M. Chan, "Downloaded 12 / 27 / 12 to 129 . 173 . 72 . 87 . Redistribution subject to SIAM license or copyright ; see http://www.siam.org/journals/ojsa.php Copyright © by SIAM . Unauthorized reproduction of this article is prohibited . Copyright © by SIAM . Unauthorized reproduction of this article is prohibited .," vol. 39, no. 5, pp. 2075–2089, 2010.

[39] T. Takaoka, "A simplified algorithm for the all pairs shortest path problem with O (n 2 log n) expected time," pp. 326–337, 2013.

[40] B. Lund, R. Hall, J. W. Smith, and R. Hall, "A Multi-Stage CUDA Kernel for Floyd-Warshall."