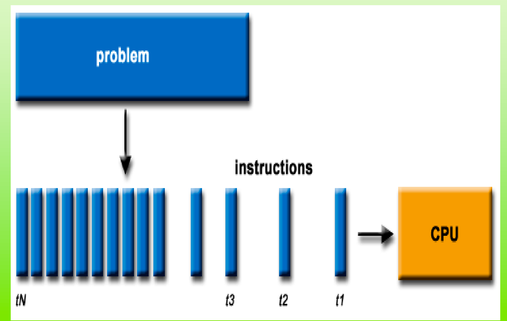
Parallelization of Graph Algorithms on GPU using CUDA

[1] Chetan D. Pise, [2] Shailendra W. Shende, [3] Rakhi D. Wajgi, [4] Chetan A. Umredkar, [5] Chittaranjan D. Pise
[1] chetspise@gmail.com, [2] shailendra.shende@gmail.com, [3] wajgi.rakhi@gmail.com, [4] chetan01umredkar@ymail.com, [5] chittupise@gmail.com
[1, 2, 3] Yeshwantrao Chavan College of Engineering Nagpur, [4] RTM Nagpur University, [5] SGGS Institute of Engineering & Technology Nanded, INDIA.

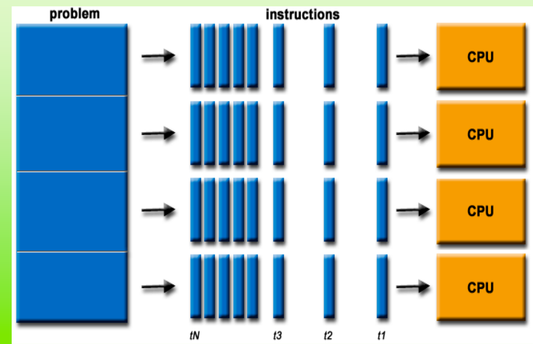
Abstract: Graphs plays very important role in the field of Science and Technology for finding the shortest distance. Here well known Graph Traversal algorithm i.e. BFS is implemented on CUDA (Compute Unified Device Architecture) framework. Graphs are used in multivariate domain including scientific and engineering applications consisting operation on million of vertices and edges. For faster execution parallel computation is very essential to reduce computation time. CUDA software interface by NVIDIA is becoming a new programming approach applicable for general purpose computing on graphics processing units (GPGPU). Massively Multithreaded architecture of CUDA device makes various threads to run in parallel and hence making optimum use of available computation power of GPU. Here we have demonstrated the comparison between serial and parallel implementation of BFS and DIJKSTRA algorithms. Generally GPU is used as an inexpensive co-processor. Breadth-first search and DIJKSTRA are very popular algorithms for graph traversal and graph analysis. Comparison between serial and parallel execution is mentioned in terms of speed up factor.

Introduction: Parallel programming is a generic concept describing a range of technologies and approaches. General Purpose computation on Graphics Processing Units (GPGPU) is a new and active field. The goal of GPGPU is to execute parallel algorithms capable of processing concurrently huge amounts of data over a number of Graphic Processing Units (GPU). Algorithms are impractical for very large graphs on normal System configurations. Parallel algorithms can achieve practical times on basic graph operations but at a high hardware cost. Pioneered in 2007 by NVIDIA Corporation, GPUs now power energy-efficient datacenters in government labs, enterprises, universities and businesses around the world.

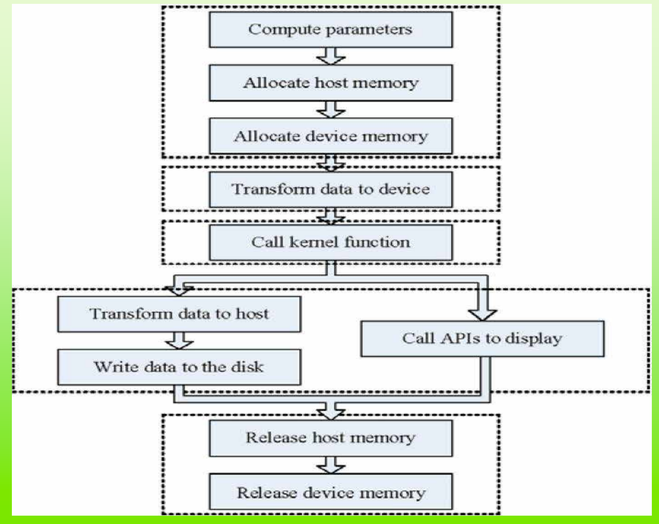
Representation of a Sequential Algorithm



Representation of a Parallel Algorithm

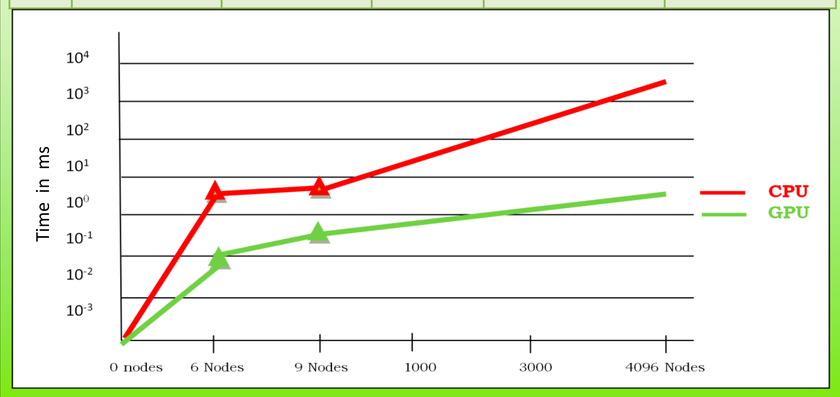


The program execution flow on CUDA



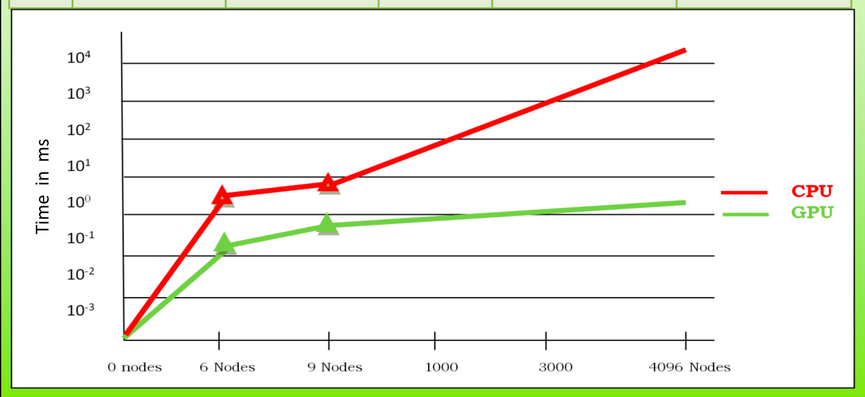
Experimental Result for BFS Algorithm

Sr. No.	Graph Data Set (No. of NODES)	Time to Execute in GPU	No. of Times Kernel Execute	Time to Execute in CPU	Speed-Up Factor (CPU/GPU)
1.	6 Nodes	0.212512 ms	3	3 ms	14.11x
2.	9 Nodes	0.291680 ms	6	3.1 ms	10.285x
3.	4096 Nodes	1.992544 ms	8	3.12 ms	1500x



Experimental Result for Dijkstra Algorithm

Sr. No.	Graph Data Set (No. of NODES)	Time to Execute in GPU	No. of Times Kernel Execute	Time to Execute in CPU	Speed-Up Factor (CPU/GPU)
1.	6 Nodes	0.229664 ms	4	3 ms	13.06x
2.	9 Nodes	0.353624 ms	6	4 ms	11.31x
3.	4096 Nodes	1.602752 ms	15	21.657 ms	13500x



Conclusion: In this project for parallelization we used NVIDIA Graphics card with 2 Multi-processors. As per this project various serial and parallel algorithms can be develop to compare the results between then and analyze which algorithms are best one for upcoming challenges. In this I used NVIDIA GTX630M for my implementation of BFS and Dijkstra algorithms with version CUDA 5.5. I used various similar data sets and find that how many times kernel is going to execute and also find the execution time of such a large data set.

Future scope:

- Create graph library of different algorithms like DFS, Prims, Kruskals.
- Enhance area of parallel computing in Bio-Informatics using CUDA.
- To bring double precision support to the GPU hardware in the future.

References:

[1] Pawan Harish and P. J. Narayanan, "Accelerating large graph algorithms on the GPU using CUDA", Center for Visual Information Technology, IIIT Hyderabad 2008.
[2] J. M. Kemp, I. A. Stewart, "Parallel Graph Searching Algorithms using CUDA", Board of Examiners in the School of Engineering and Computing Sciences Durham University 2011.
[3] Book: Jason Sanders, Edward Kandrot, "Programming based on CUDA".
[4] Book: CUDA C Programming Guide.