A GPU-based parallel computing framework for accelerating graph theoretical analyses

Tsang-Chu Yu[1], Yi-Ping Chao[1], Li-Wei Kuo[2], Chung-Chih Lin[1], Shih-Yen Lin[2,3], Hengtai Jan[2], Claudia Metzler-Baddeley[4], and Derek Jones[4]

[1]Department of Computer Science and Information Engineering, Chang Gung University, Taoyuan, Taiwan, [2]Institute of Biomedical Engineering and Nanomedicine, National Health Research Institutes, Miaoli, Taiwan, [3]Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, [4]School of Psychology, Cardiff University, Cardiff, United Kingdom

Introduction

Recent studies have suggested that a combination of multi-modal brain magnetic resonance imaging (MRI) techniques (e.g., structural MRI, functional MRI and diffusion MRI) together with graph theory approaches can help us to noninvasively map structural and functional connectivity patterns of the human brain [1]. Combination with structural MRI, functional MRI and graph theoretical analyses (GTA), these studies provide the insights into the architecture and organization of large-scale brain networks (known as the "human connectome") [2]. Notably, the graph is comprised of "node"(general cortical/subcortical regions) and "edge"(functional or structural connection between them) to provide a number of metrics to characterize the local and global efficiency of the network [3]. The aim of this study is to implement an acceleration platform for brain network analysis based on GPUs and CUDA. In comparison with multi-cores CPU or clusters, GPUs with CUDA is provided with the powerful of parallel computing and the benefit of low cost. By implementing the fundamental graph algorithm-all pairs shortest path (APSP), our GPU-based platform of human connectome will make the construction and analysis of large-scale brain network (number of node > 8k) more efficiently and quickly.

Materials and Methods

All pair shortest path (APSP): The equation of APSP is based on Rubinov's study[3]. The d_{ij} means shortest path length (distance), between nodes i and j, where $g_{i\leftrightarrow j}$ is the shortest path between i and j. Note that $d_{ij} = \infty$ for all disconnected pairs i, j. In this study, we only focus on binary brain networks, which can be built by applying a threshold on the weight of edges (such as fractional anisotropy or number of fiber).

Simulated network matrix:

In order to validate the correctness and evaluate the performance of our algorithm using GPUs with CUDA, we use a random matrix generator based on Erdos-Renyi model to estimate three matrices of 2048, 4096 and 8192 nodes with the matrix density of 25%, 50%, 75% and 95% respectively. The calculations of APSP were then executed by Brain Connectivity Toolbox (BCT, http://www.brain-connectivity-toolbox.net/), Gretna (http://www.nitrc.org/projects/gretna) and our GPU/CUDA version. Finally, the results and time costs would be compared between BCT, Gretna and our algorithm.

Real brain network matrix derived from diffusion tractography:

Data were acquired using the CUBRIC 3T GE HDx MRI system. Cardiac-gated HARDI diffusion MRI employed an optimized 60 directions gradient vector scheme and b-value 1200s/mm2, 60 slices (2.4mm), FoV 24 cm, matrix 96x96, TE 87ms. Images were corrected for distortions and motion, with re-orientation of gradient directions and restored in ExploreDTI_4.8.3[4]. The CSD reconstruction is employed to estimate the fiber orientations. The parameters for fiber tracking are listed here: step size of 1mm, angle threshold of 30^o, fiber length range of 50- 500(mm). Two kinds of brain parcellation including 116 and 180 subdivisions were employed for the construction of brain network here.

The implementation of our algorithm using GPUs/CUDA:

The BCT takes a lot of time in matrix multiplication. Therefore, we try to reduce the time of matrix multiplication in large-scale network through block matrix. There are three advantages of dividing into block matrix. The first is the data size of each block matrix is small so we can put it into shared memory. The second is the number of data accessing will reduce because the matrix can load one block size per time. The third is each block matrix is independent so that we can use the character of GPU to assign each thread a block matrix. Then, the computation can be reduced.

Results

BCT and Grenta ran at 2 Intel xeon processors E5-2670 CPU running at 2.6GHz, 128GB DDR3 memory. Our algorithm ran at Intel i7-3770K CPU running at 3.5GHz, 16 GB DDR3 memory and the graphic card we use is tesla K20C at 706MHZ and 5GB GDDR5 memory. From the results, our implementation could reduce half of time with BCT and 638x speedup with Gretna in simulation random network with larger number of nodes (>8k). Moreover, our algorithm also shows better performance in human brain data with 1.37x and 21x speedup in comparison with BCT and Gretna respectively.

Table 1. Simulated network matrix results (unit: seconds)

Matrix size	Matrix density	Gretna	BCT	GPU/CUDA
2048	25%	144.98	0.57	0.28
	50%	163.59	0.56	0.28
	75%	153.18	0.61	0.28
	95%	152.2	0.61	0.28
4096	25%	1178.9	4.36	1.89
	50%	1169.3	4.17	1.89
	75%	1166.8	4.17	1.89
	95%	1168.9	4.08	1.89
8192	25%	9060	28.72	14.19
	50%	No result	19.75	14.19
	75%	No result	19.34	14.15
	95%	No result	18.48	14.17

Table2 Real brain network matrix results (unit:seconds)

	Matrix size	Matrix density	Gretna	BCT	GPU/CUDA
Subject 1	116	37.12%	0.077	0.0042	0.00258
	180	28.7%	0.16	0.0115	0.0048
Subject 2	116	42.67%	0.032	0.0023	0.00258
	180	31.53%	0.093	0.0061	0.0048
Subject 3	116	40.66%	0.031	0.0024	0.0026
	180	31.4%	0.094	0.0056	0.0048

References

[1] Bullmore E., et al, Nature. 10:186-98, 2009. [2] Di Wu et al, ICPADS, 593-600, 2010. [3] Rubinov, M., et al, NeuroImage. 52:1059-69, 2010. [4] Leemas A. et al., ISMRM, No. 3537, 2009.

Discussions & Conclusion

The main contribution of the study is to exploit the power of the GPU, to significantly accelerate processing time and facilitate large-scale network analyses. The results showed that the acceleration of our algorithm applied in human brain network is less than in simulated random network. The reason is that the computing using GPU needs to move the data from CPU to GPU memory. So, if the matrix size is smaller, the advantages of GPU computing might be not obvious. On the contrary, if the size of network matrix is larger, all computations could be divided into several tasks for parallelism and then the processing time could be decreased. Therefore, our development provides a potential for speedy and comprehensive analysis of brain network with detailed parcellations in the cortex or large amount of dataset in the studies of brain research.