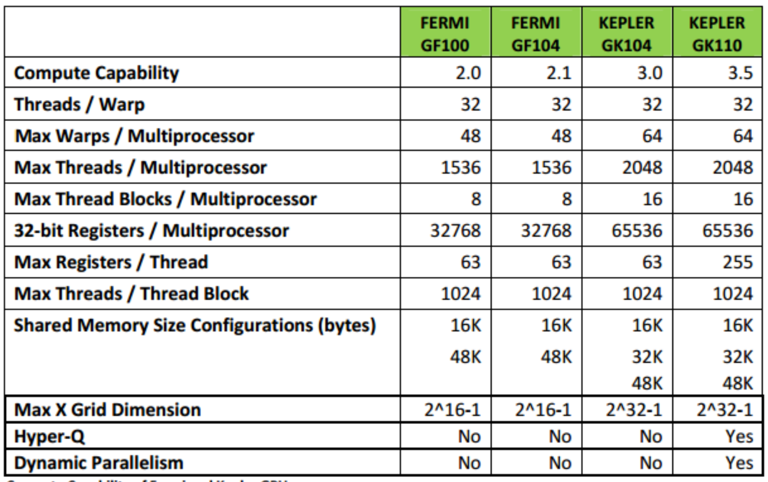


	FERMI GF100	FERMI GF104	KEPLER GK104	KEPLER GK110
Compute Capability	2.0	2.1	3.0	3.5
Threads / Warp	32	32	32	32
Max Warps / Multiprocessor	48	48	64	64
Max Threads / Multiprocessor	1536	1536	2048	2048
Max Thread Blocks / Multiprocessor	8	8	16	16
32-bit Registers / Multiprocessor	32768	32768	65536	65536
Max Registers / Thread	63	63	63	255
Max Threads / Thread Block	1024	1024	1024	1024
Shared Memory Size Configurations (bytes)	16K 48K	16K 48K	16K 32K 48K	16K 32K 48K
Max X Grid Dimension	2^16-1	2^16-1	2^32-1	2^32-1
Hyper-Q	No	No	No	Yes
Dynamic Parallelism	No	No	No	Yes

Compute Capability of Fermi and Kepler GPUs

http://www.nvidia.com/content/PDF/kepler/NVIDIA-kepler-GK110-Architecture-Whitepaper.pdf

Maximum x-dimension of a grid of thread blocks, 65535, 2^{31}-1

So the max number of threads that can contain in 1D-grid = 65535*1024=67,107,804.

We now use a thread to compute an element in vector. Since the core operation is matrix-vector multiplication, we have tested several methods.

Environment: Windows 7 GPU: Nvidia Geforce GT525M

Matrix: 8192*8192 Vector: 8192

Type: Float

Naïve multiplication	289.2ms
Coalesce memory	187.3ms
Shared memory	200ms
CuBLAS cublasSgemv	To be tested

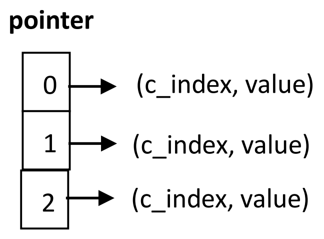
It is confused that the optimization using shared memory performs worse than the second one. Perhaps it is due to the improper size of shared memory, which leads to low usage of processor. Another reason may be the cost of synchronization. We will test it in the future work.

According to Pegasus, PageRank can be computed as iterated matrix-vector multiplication. Therefore in a single machine, there are steps

1. Generate adjacent matrix and initial vector from file.
2. Iteratively run matrix-vector multiplication in GPU.
3. If the value of vector is converged, or it reaches the maximum iteration times, stop computing and load vector data to memory.

When running on local machine, there are several problems in each step.

Due to the limit, it is impossible to store the whole matrix in memory when there are 10,000 nodes. A direct solution is to load data segment separately. However, this method require data transferring from disk to memory, then memory to graph memory, both of which are expensive costs. Another solution is to store coordination pair of element with non-zero value, instead of the whole matrix.



Since most of adjacent matrix of a large graph in real life is sparse. The space complexity reduces from S(n^2) to S(n). Therefore, if the matrix is 1M*1M*4B, the compressed one is only 2M.
The time complexity is also decreased when using sparse matrix method. According to a paper, the complexity reduces from O(n^3) to O(n). So we choose to use this method.