Distributed Programming

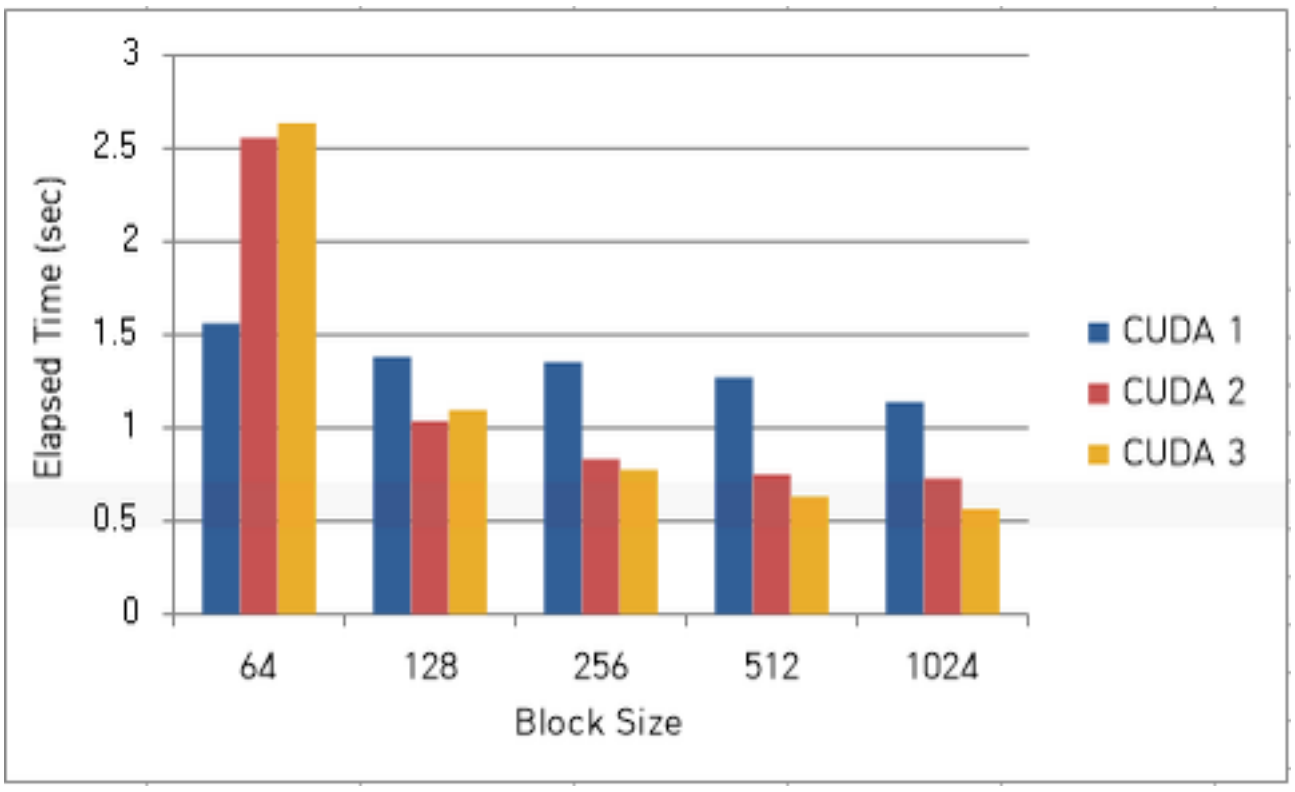
Assignment #4

by 20141500 권태국

1. CUDA Programming 1 – Matrix Multiplication

1.1 (a) Compare CUDA implementations in terms of different block and thread sizes.

MATRIX SIZE = 4096, BLOCK_WIDTH=32, GPU IO시간제외					
Block Size	64	128	256	512	1024
CUDA 1	1.564	1.385	1.356	1.272	1.141
CUDA 2	2.56	1.039	0.835	0.75	0.731
CUDA 3	2.639	1.099	0.776	0.634	0.566



위는 Matrix Size = 4096, Block Width = 32 로 고정했을 때, block size를 변화해가면서 수행시간을 측정한 결과이다.

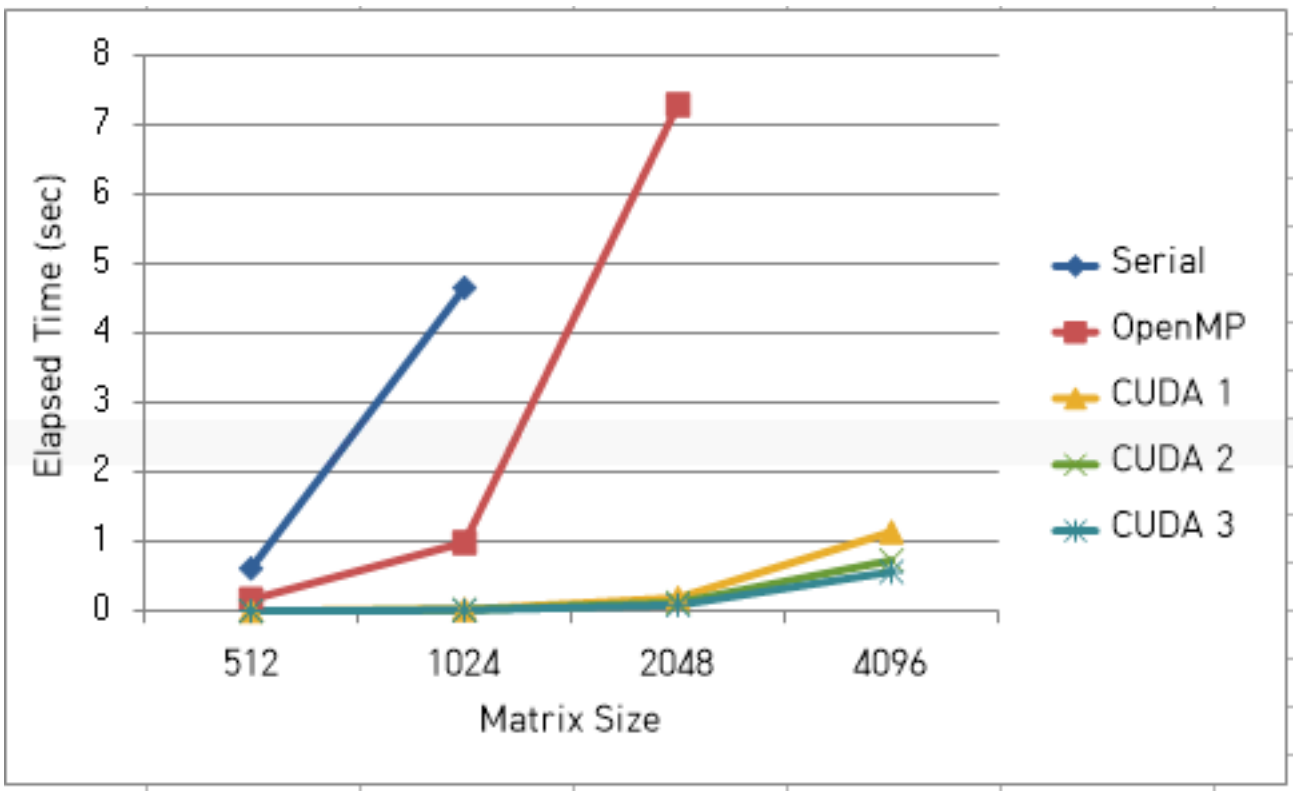
결과분석을 하기전에 CUDA 1, 2, 3에 대해 설명하겠다. CUDA 1은 단순하게 구현한 버전이다. CUDA 2는 C = A x B에서 B에 대해 shared memory를 사용한 버전이다. CUDA 3는 A와 B모두에 대해 shared memory를 사용한 버전이다.

결과분석을 해보겠다. Block Size가 커질수록 수행 속도가 빨라짐을 알 수 있다. 특히 Block Size가 64로 내려갔을 때, CUDA 2와 3의 수행 시간이 급격하게 느려지는 것을 볼 수 있다.

1.2 (b) Repeat comparison by varying the matrix size. (c) Compare with serial and OpenMP versions.

BLOCK_WIDTH=32, BLOCK_HEIGHT=32, GPU IO시간제외

MatrixSize	512	1024	2048	4096
Serial	0.615	4.658		
OpenMP	0.167	0.988	7.296	
CUDA 1	0.003	0.024	0.191	1.141
CUDA 2	0.002	0.017	0.122	0.731
CUDA 3	0.002	0.013	0.088	0.566



위는 Block Width와 Height를 32로 고정하고, GPU IO시간을 수행시간에서 제외했을 때 Matrix Size에 따른 수행 시간 결과를 측정한 결과이다.

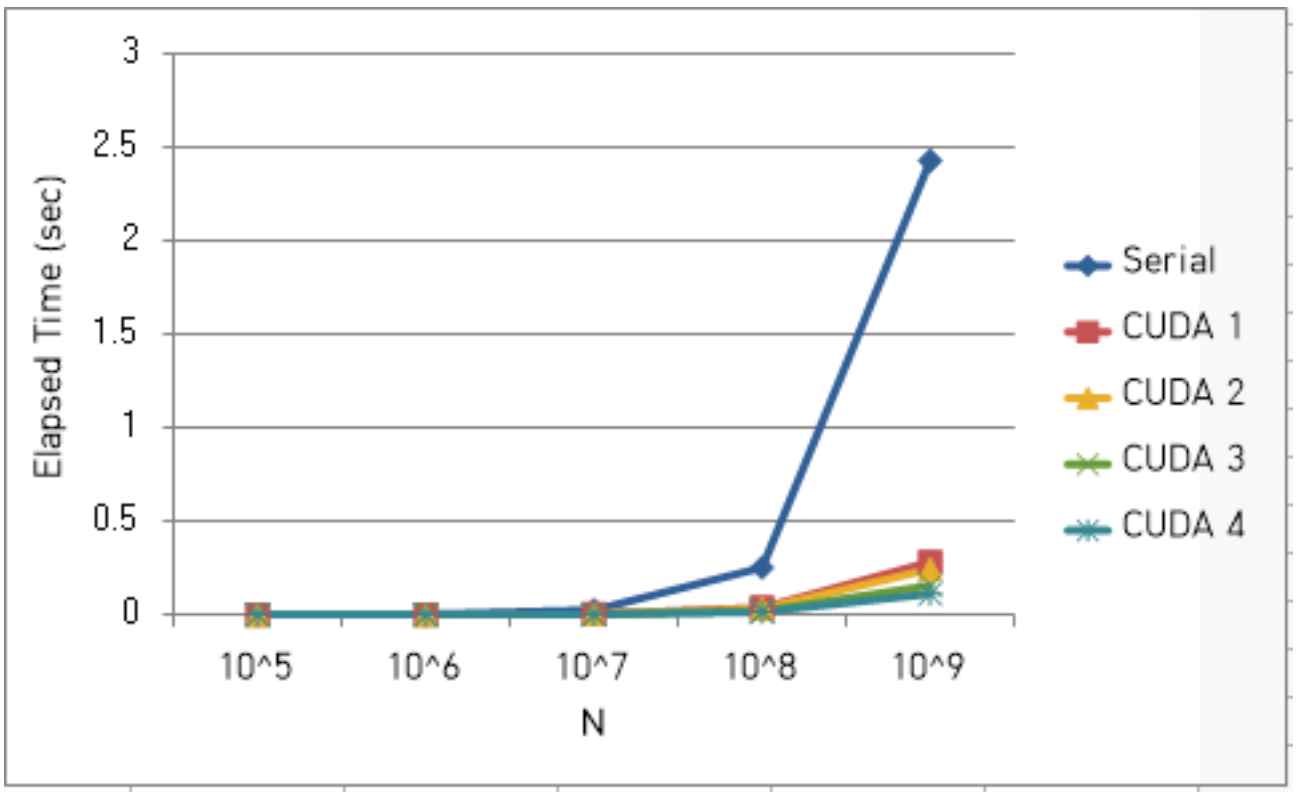
일단 Serial과 OpenMP version에 비해 CUDA를 사용한 version들이 월등히 빠름을 알 수 있다. 그리고 OpenMP가 Serial Version에 비해 4~5배정도 빠른 것을 볼 수 있는데, 실험 수행 환경이 Physical Core 3개 (hyperthreading이 적용되어 Logical Core 6개) 인 환경이므로 올바른 결과라고 할 수 있다. 그리고, Matrix Size가 2048이 넘어가면, Serial의 수행시간이 너무 길어서 측정에서 제외하였고, 마찬가지로 Matrix Size가 4096일 때, Serial과 OpenMP version을 측정에서 제외하였다. 그리고 CUDA 버전끼리를 비교해보면, CUDA 1보다 2가, 2보다 3가 더 빠름을 볼 수 있다. 당연하게도 2와 3는 shared memory를 사용하고 bank conflict가 없음으로 1보다 당연히 빠를 것이고, 3는 2에 비해서 행렬 B에 대해서도 shared memory를 사용함으로, 이런 결과가 나온 것은 이론에 부합하다고 할 수 있다.

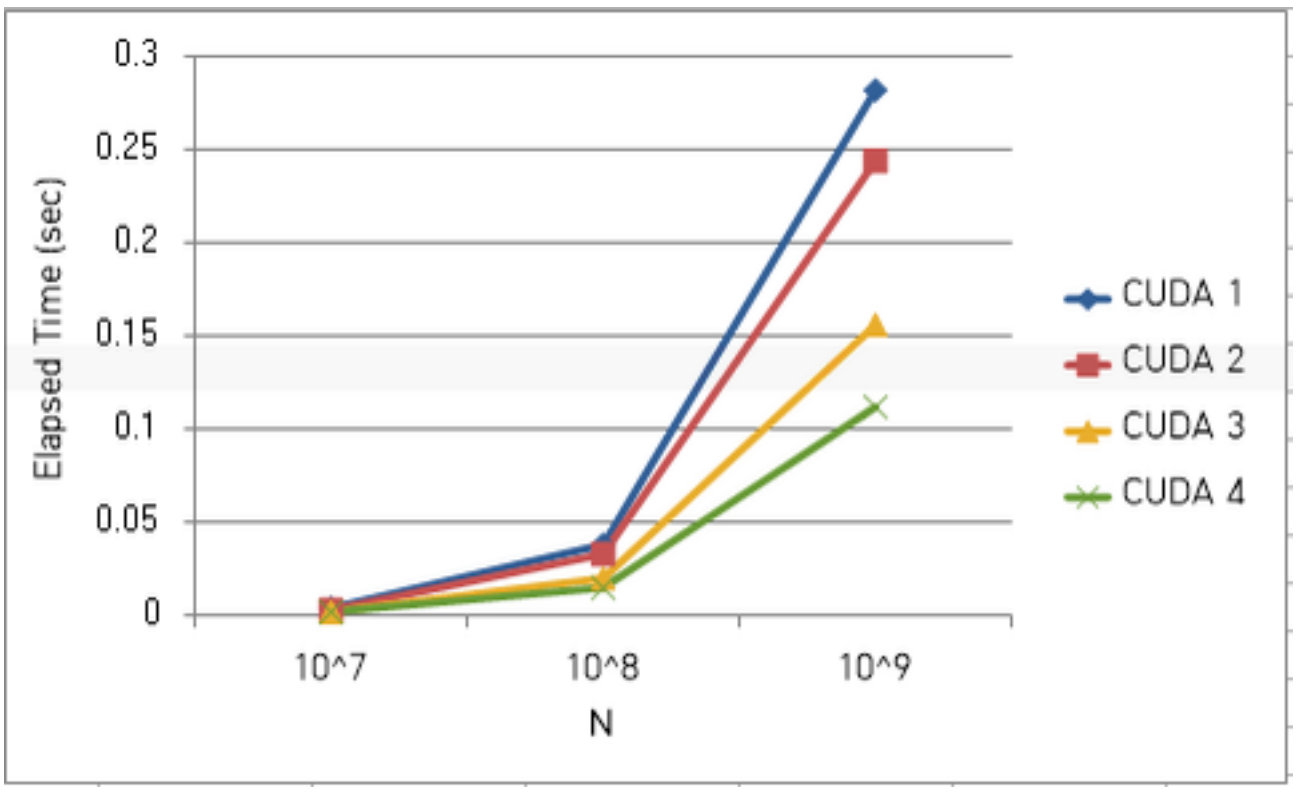
2. CUDA Programming 2 – Reduction

(1) Compare the performance of each version and discuss your results and findings.

(2) Repeat the same comparison with different array sizes.

BLOCK_SIZE=1024, GPU IO시간제외					
N	10^5	10^6	10^7	10^8	10^9
Serial	0	0.002	0.025	0.252	2.43
CUDA 1	0	0	0.004	0.038	0.282
CUDA 2	0	0	0.003	0.033	0.244
CUDA 3	0	0	0.002	0.02	0.156
CUDA 4	0	0	0.002	0.015	0.112





위는 Block Size를 1024로 고정하고 GPU IO시간을 제외했을 때, Serial과 각종 CUDA version들의 Max Reduction 수행 시간을 측정한 결과이다.

CUDA 1은 path divergence를 고려하지 않았을 때, cuda implementation이다. CUDA 2는 1에서 path divergence를 고려하여 최적화한 버전이다. CUDA 3는 2에서 shared memory를 사용하여 최적화한 버전이다. CUDA 4는 CUDA 3에 존재하는 bank conflict를 제거하여 최적화한 최종 버전이다.

역시나 Serial에 비해서 GPU들은 매우 빠른 속도를 보여주고 있다. Serial의 경우 특징은

N이 10배 증가하면 수행 시간도 10배 증가한다는 것이다. 그러나 CUDA version의 경우 N이 10배 증가할 때, 수행 시간이 7~8배정도 증가한다.

CUDA 1~4를 비교해보면, CUDA 1 > 2 > 3 > 4 순서로 수행시간이 긴 것을 알 수 있다. (4가 가장 성능이 좋음.) 1의 경우 단순히 tree 구조의 reduction을 구현한 것인데, global memory를 사용하고 path divergence가 존재해 비효율적이다. 그래서 2는 path divergence를 없앴고, 3에서는 shared memory를 사용했다. 그리고 4에서는 3에 존재하는 bank conflict도 없앴다. 그러나 성능이 점점 더 빨라지는 것이 당연하다.