Matrix Multiplication using GPU

Naveen Himthani (120010001)

Algorithm 1

- Uses multiple blocks in only one direction x
- Works for any matrix dimensions without conditions on divisibility

Algorithm 2

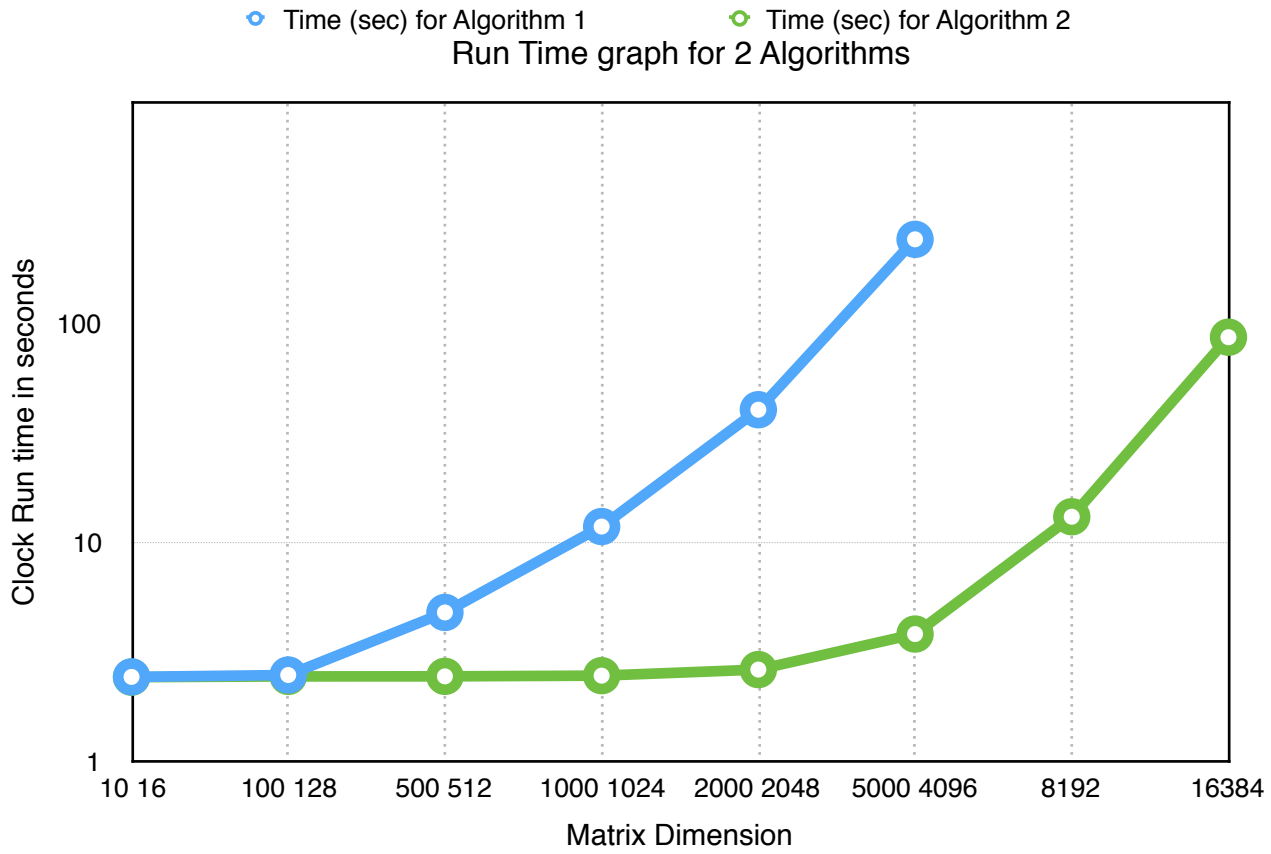
- Uses multiple blocks in both dimensions x & y
- For matrix dimensions greater than 512, works for dimensions which are powers of 2
- Will have to figure out, how to make it work for arbitrary dimensions

Observations:

- Algorithm 1 is slower than 2 major because of the use of blocks in both dimensions
- In Algorithm 1, the sum of the product of individual elements is done by a single thread in each block, hence the slower speed
- In algorithm 2, multiple blocks write and add to the resulting C matrix at the same time
- However no race conditions are checked in Algorithm 2

Timing Observations

N (matrix size) for Algorithm 1	N (matrix size) for Algorithm 2	Time (sec) for Algorithm 1	Time (sec) for Algorithm 2
10	**16**	2.416	2.409
100	**128**	2.464	2.428
500	**512**	4.754	2.430
1000	**1024**	11.721	2.448
2000	**2048**	39.990	2.609
5000	**4096**	239.481	3.789
	8192		12.994
	16384		85.646



Machine Specifications:

VGA compatible controller: NVIDIA Corporation GK104 [GeForce GTX 680] (rev a1) (prog-if 00 [VGA controller])

- Physical Slot: 3
- Flags: bus master, fast devsel, latency 0, IRQ 146
- Memory at dc000000 (32-bit, non-prefetchable) [size=16M]
- Memory at d0000000 (64-bit, prefetchable) [size=128M]
- Memory at d8000000 (64-bit, prefetchable) [size=32M]
- I/O ports at 3000 [size=128]
- [virtual] Expansion ROM at dd000000 [disabled] [size=512K]
- Capabilities: <access denied>
- Kernel driver in use: nvidia