MA 513 Report I
Optimization of Matrix Multiplication

Deepak Kumar Gouda
160123054

Theoretical estimate of time required for matrix multiplication :

The CPU speed is as follows

```
     CPU MHz : 2100.212
     CPU max MHz : 3800.0000
     CPU min MHz : 800.0000
```

And the cache sizes are as follows :

```
      L1d cache:           32K
      L1i cache:           32K
      L2 cache:            256K
      L3 cache:            6144K
```

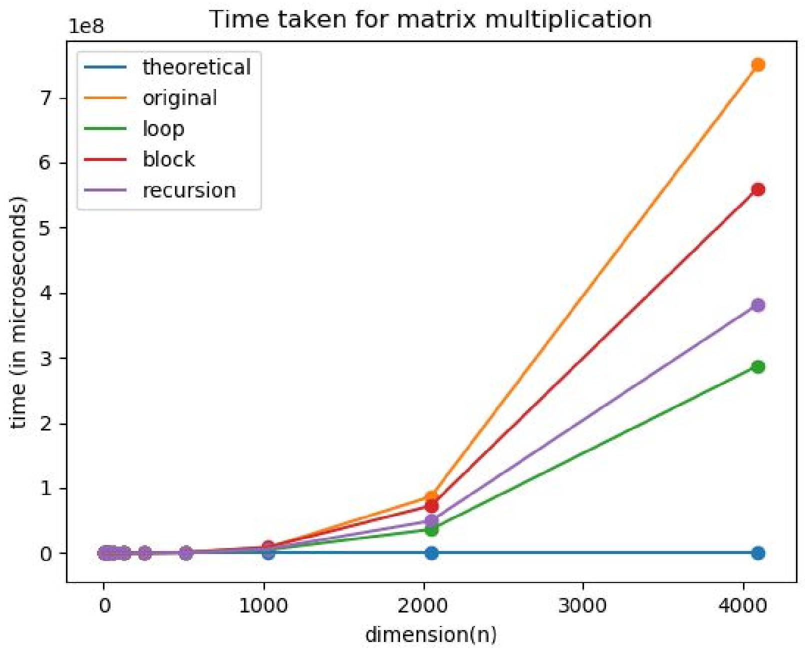
Matrix multiplication takes 2*n^3 operations. As 2.1*10^9 operations can be performed in 1 second, the time taken for matrix multiplication is

Total operations / CPU Speed = 2*n^3/speed

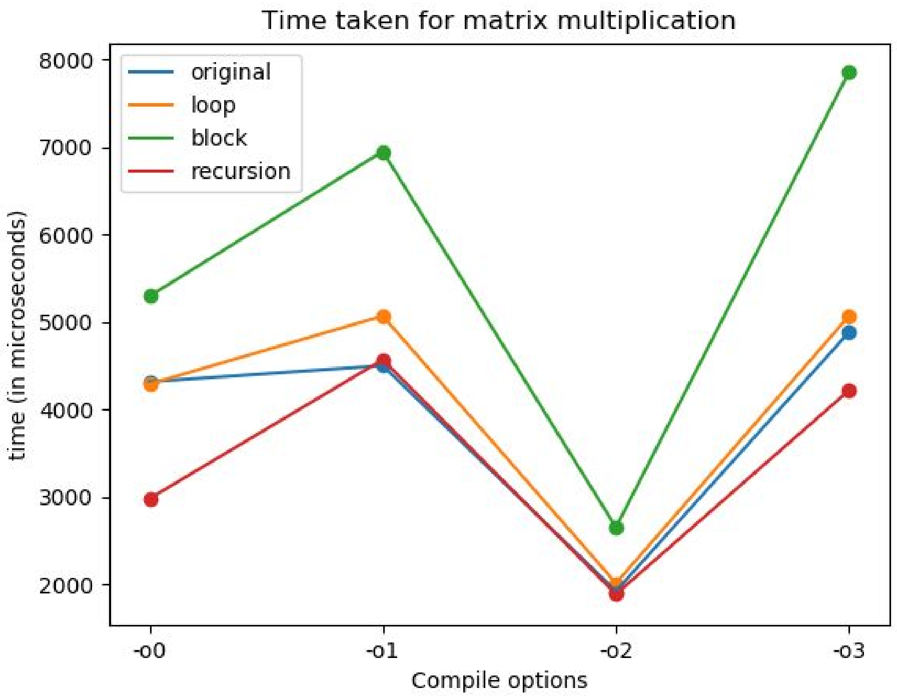
The size of L1 cache is 32 Kilobytes. The value of ‘s’ should be such that a matrix must fully reside in the cache. Hence, the size of the matrix must be less than 32 Kilobytes. An integer takes 4 bytes. So, we can keep 8000 integers in the cache. A matrix stores n^2 integers and hence, we can keep an 89X89 matrix (sqrt(8000) ~ 89). We keep the dimensions as powers of 2, hence we take the value of ‘s’ as 64.

Plots :

Dimension vs Time



Compiler options vs Time



It can be observed that the option ‘-o2’ provides the best performance.

Analysis :

For smaller dimensions, block and recursion methods perform worse than the naive method. As we increase the dimension, block and recursion perform better than naive method and the loop interchange method performs the best overall.

The theoretical time estimated is substantially lesser than the actual values. This can be explained by the fact that accessing values from cache and main memory requires time. In recursion we also need to store the recursion stack and make a function call. All of these factors are not considered in the theoretical estimate where we assume that the numbers can be accessed with no latency and there are no other overheads.