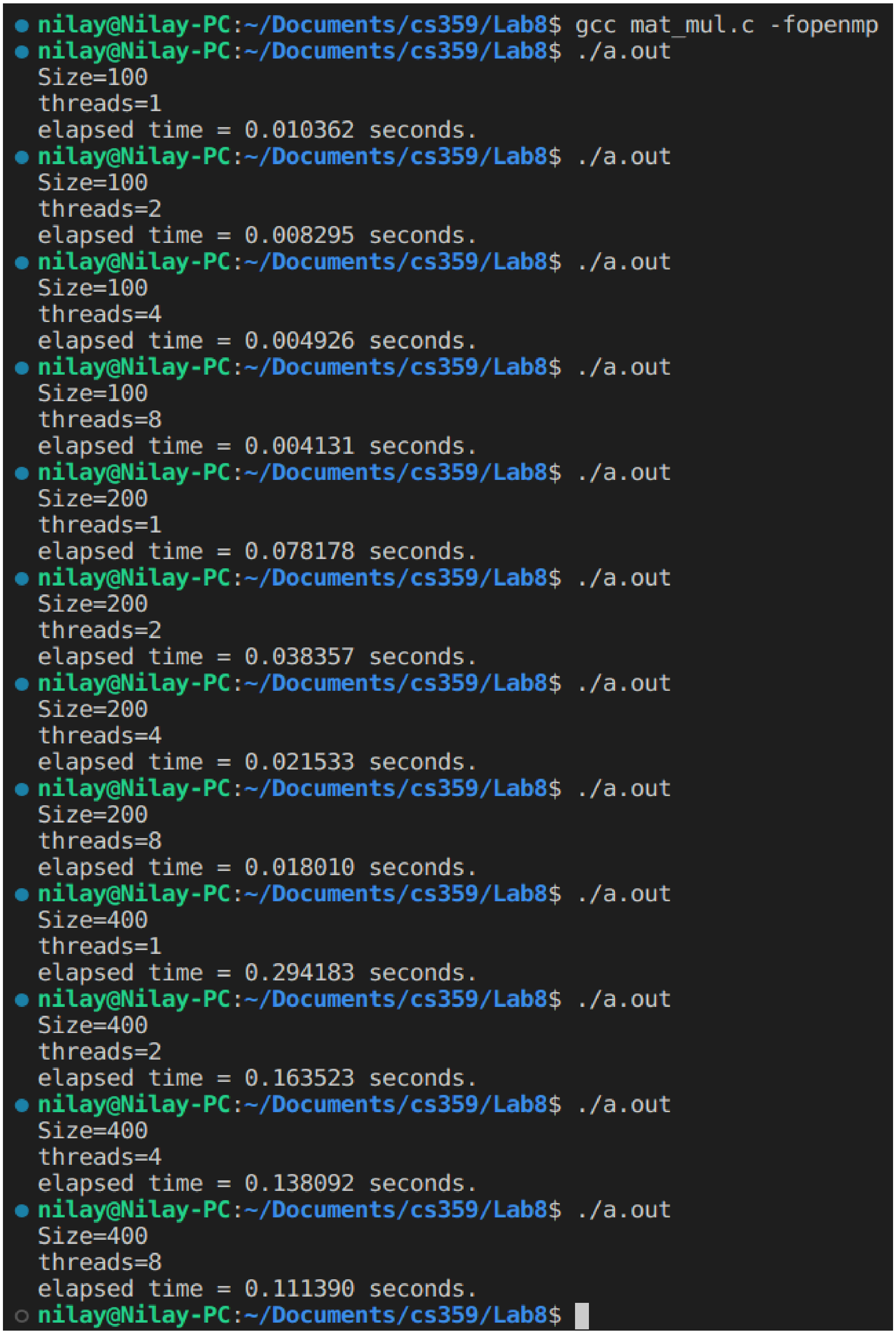
Parallel Computing Lab
Nilay Ganvit - 200001053
10th November 2022

Lab 8

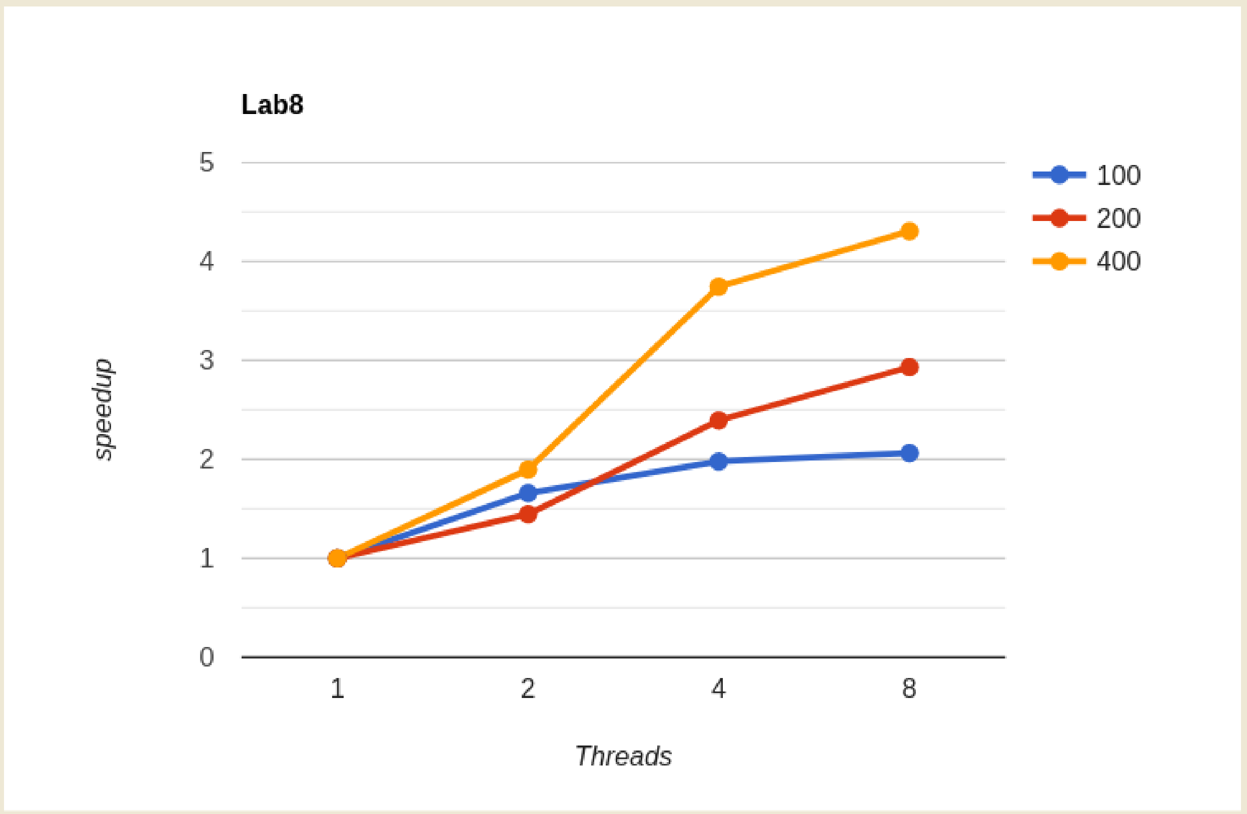
```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#include <sys/time.h>
#include <time.h>
#include <stdlib.h>

int main()
{
    int N;
    int threads;
    printf("Size=");
    scanf("%d", &N);
    printf("threads=");
    scanf("%d", &threads);

    int A[N][N];
    int B[N][N];
    int C[N][N];

    int i, j, k;
    struct timeval tv1, tv2;
    struct timezone tz;
    double elapsed;
    omp_set_num_threads(threads);
    for (i = 0; i < N; i++)
        for (j = 0; j < N; j++)
        {
            A[i][j] = rand() % 100;
            B[i][j] = rand() % 100;
        }
    gettimeofday(&tv1, &tz);
#pragma omp parallel for private(i, j, k) shared(A, B, C)
    for (i = 0; i < N; ++i)
    {
        for (j = 0; j < N; ++j)
        {
            for (k = 0; k < N; ++k)
            {
                C[i][j] += A[i][k] * B[k][j];
            }
        }
    }

    gettimeofday(&tv2, &tz);
    elapsed = (double)(tv2.tv_sec - tv1.tv_sec) + (double)(tv2.tv_usec -
tv1.tv_usec) * 1.e-6;
    printf("elapsed time = %f seconds.\n", elapsed);
}
```

Input/Output:



```
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ gcc mat_mul.c -fopenmp
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=100
  threads=1
  elapsed time = 0.010362 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=100
  threads=2
  elapsed time = 0.008295 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=100
  threads=4
  elapsed time = 0.004926 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=100
  threads=8
  elapsed time = 0.004131 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=200
  threads=1
  elapsed time = 0.078178 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=200
  threads=2
  elapsed time = 0.038357 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=200
  threads=4
  elapsed time = 0.021533 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=200
  threads=8
  elapsed time = 0.018010 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=400
  threads=1
  elapsed time = 0.294183 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=400
  threads=2
  elapsed time = 0.163523 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=400
  threads=4
  elapsed time = 0.138092 seconds.
● nilay@Nilay-PC:~/Documents/cs359/Lab8$ ./a.out
  Size=400
  threads=8
  elapsed time = 0.111390 seconds.
○ nilay@Nilay-PC:~/Documents/cs359/Lab8$
```

Graph:



Efficiency:

Size/Threads:	100	200	400
1	1	1	1
2	0.829	0.722	0.948
4	0.494	0.598	0.935
8	0.257	0.366	0.537

- The **scaling Behaviour** of the parallel program does **deviate from the Ideal** numbers it should have got from manual calculations.
- **But it does follow the trend**, that is increasing as the size of the matrix increases and decreasing as the number of cores increases.