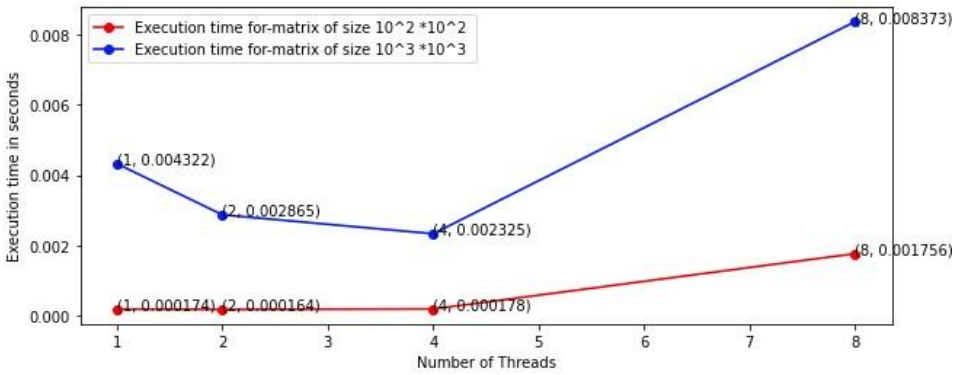
Problem2 on OpenMP:

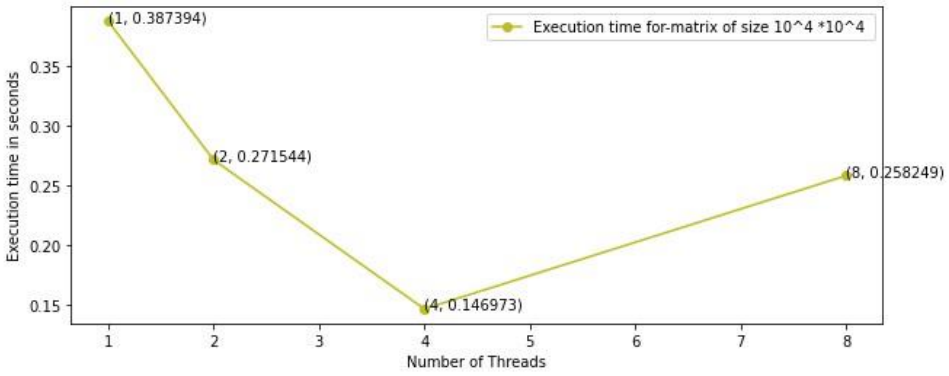
Name: BISWARUP KARMAKAR (SR:21055)

Execution time taken in seconds for different thread and N:

	Thread=1	Thread=2	Thread=4	Thread=8	L2 Norm of the error vector
N=100	0.000174	0.000164	0.000178	0.001756	5.387458
N=1000	0.004322	0.002865	0.002325	0.008373	18.379785
N=10000	0.387394	0.271544	0.146973	0.258249	58.132747

Plotting:





I have got all the output using CDS Turing Cluster.

It can be observed that with vector length N=100, we got the fastest execution time(0.000164 seconds) with number of threads=4 and with thread =8, it takes much more time. Again, for the length N=1000 with number of the threads=2 we got fastest time and for the thread =4,8 it takes much more time.

Now when the vector length N=10000 with the number of threads=4 we got the fastest computational time(0.146973 seconds).

.

I noticed that having more than 4 threads running for some code is much slower than having one thread.

(1) Because thread creation has overhead. If the task to be performed has only small computational cost, then the cost of creating multiple threads is more than the time saved by parallelism. Especially the case **when we are creating significantly more threads than there are CPU cores**.

(2) Many algorithms do not easily divide into independent sub-tasks. Dependencies on other threads requires synchronization, which has overhead that can in some cases be more than the time saved by parallelism. For a poorly designed programs, synchronization can cause all tasks to be processed sequentially even if they are in separate threads.

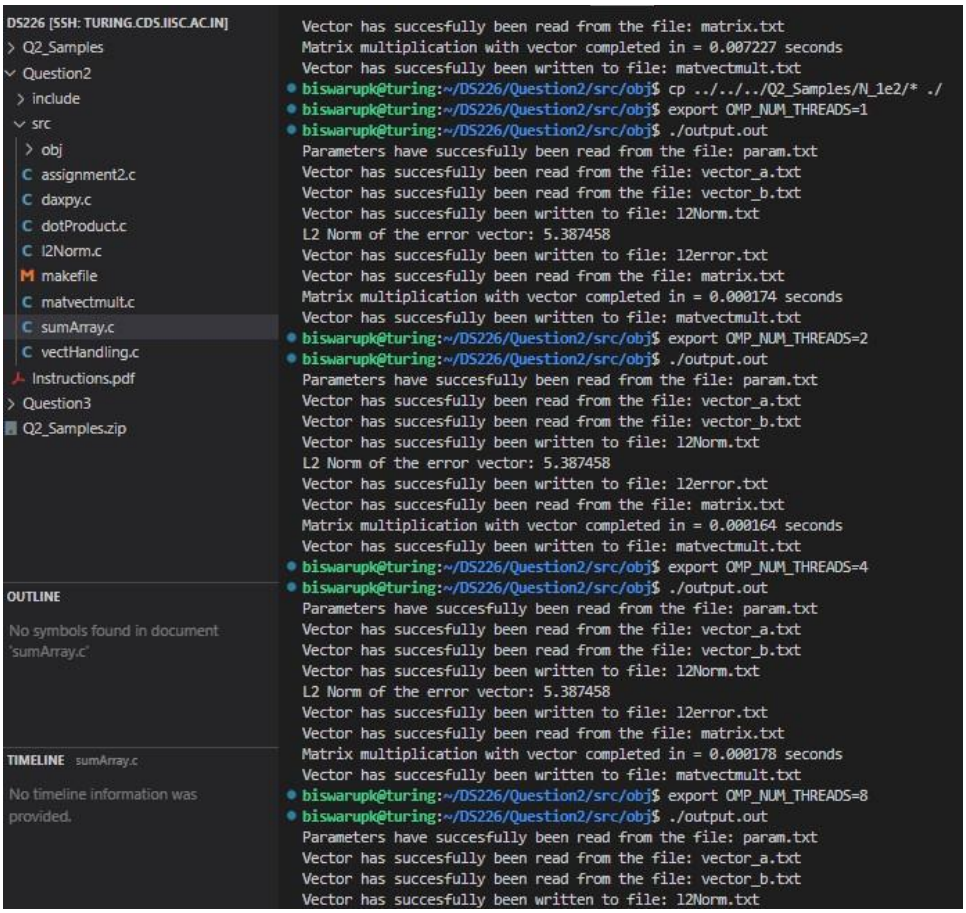
(3) When a thread writes into a piece of memory, all threads operating on the same cache line must synchronize (the CPU does this for you automatically) to remain consistent. The cost of cache misses is often much higher than the time saved by parallelism. This problem is called "false sharing".

In conclusion Threads are not automatically multiplies the performance of your program.

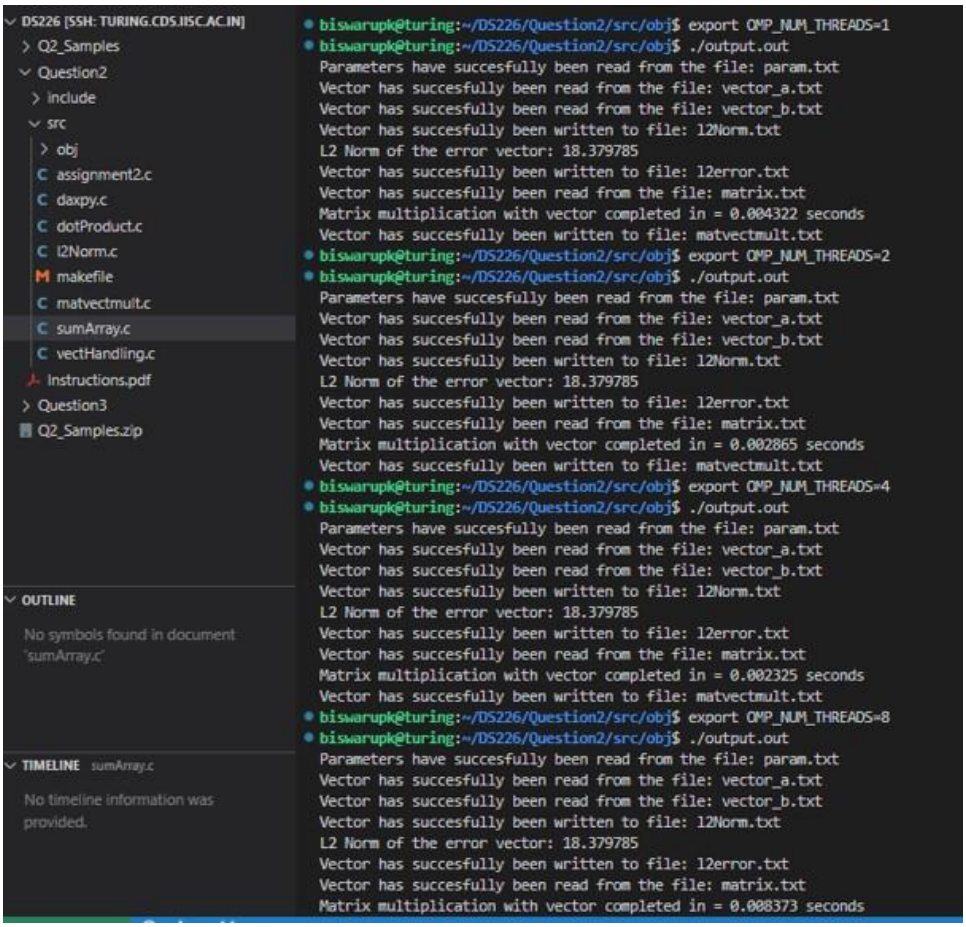
Python Code for plotting:

```
import numpy as np
import matplotlib.pyplot as plt
Thread=np.array([1,2,4,8])
Time_N_e2=np.array([0.000174,0.000164,0.000178,0.001756])
Time_N_e3=np.array([0.004322,0.002865,0.002325,0.008373])
Time_N_e4=np.array([0.387394, 0.271544,0.146973,0.258249])
fig,ax=plt.subplots(figsize=(10,4))
for i in range(len(Thread)):
  ax.text(Thread[i],Time_N_e2[i],(Thread[i],Time_N_e2[i]))
  ax.text(Thread[i],Time_N_e3[i],(Thread[i],Time_N_e3[i]))
plt.plot(Thread,Time_N_e2,color='r',marker='o',label="Execution time for-\
matrix of size 10^2 *10^2 ")
plt.plot(Thread,Time_N_e3,color='b',marker='o',label="Execution time for-\
matrix of size 10^3 *10^3 ")
plt.xlabel("Number of Threads")
plt.ylabel("Execution time in seconds")
plt.legend()
plt.show()

fig,ax=plt.subplots(figsize=(10,4))
plt.plot(Thread,Time_N_e4,color='y',marker='o',label="Execution time for-\
matrix of size 10^4 *10^4 ")

for i in range(len(Thread)):
 ax.text(Thread[i],Time_N_e4[i],(Thread[i],Time_N_e4[i]))
plt.xlabel("Number of Threads")
plt.ylabel("Execution time in seconds")
plt.legend()
plt.show()
```

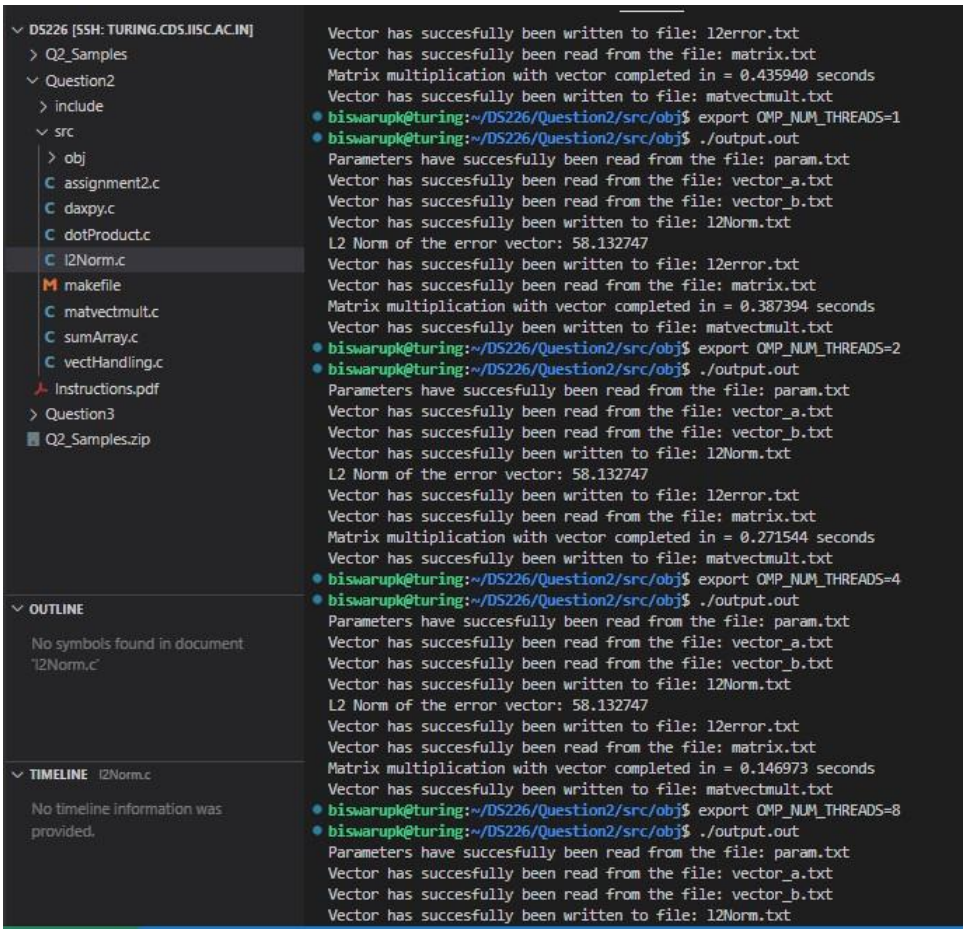
The Output for the vector length N=10^2



The output for the vector length N=10^3



The output for the vector length N=10^4


DS226 [SSH: TURING.CDS.IISC.AC.IN]
Q2_Samples
Question2
include
src
obj
assignment2.c
daxpy.c
dotProduct.c
l2Norm.c
makefile
matvectmult.c
sumArray.c
vectHandling.c
Instructions.pdf
Question3
Q2_Samples.zip
OUTLINE
No symbols found in document 'l2Norm.c'
TIMELINE l2Norm.c
No timeline information was provided.
Vector has succesfully been written to file: l2error.txt
Vector has succesfully been read from the file: matrix.txt
Matrix multiplication with vector completed in = 0.435940 seconds
Vector has succesfully been written to file: matvectmult.txt
biswarupk@turing:~/DS226/Question2/src/obj$ export OMP_NUM_THREADS=1
biswarupk@turing:~/DS226/Question2/src/obj$ ./output.out
Parameters have succesfully been read from the file: param.txt
Vector has succesfully been read from the file: vector_a.txt
Vector has succesfully been read from the file: vector_b.txt
Vector has succesfully been written to file: l2Norm.txt
L2 Norm of the error vector: 58.132747
Vector has succesfully been written to file: l2error.txt
Vector has succesfully been read from the file: matrix.txt
Matrix multiplication with vector completed in = 0.387394 seconds
Vector has succesfully been written to file: matvectmult.txt
biswarupk@turing:~/DS226/Question2/src/obj$ export OMP_NUM_THREADS=2
biswarupk@turing:~/DS226/Question2/src/obj$ ./output.out
Parameters have succesfully been read from the file: param.txt
Vector has succesfully been read from the file: vector_a.txt
Vector has succesfully been read from the file: vector_b.txt
Vector has succesfully been written to file: l2Norm.txt
L2 Norm of the error vector: 58.132747
Vector has succesfully been written to file: l2error.txt
Vector has succesfully been read from the file: matrix.txt
Matrix multiplication with vector completed in = 0.271544 seconds
Vector has succesfully been written to file: matvectmult.txt
biswarupk@turing:~/DS226/Question2/src/obj$ export OMP_NUM_THREADS=4
biswarupk@turing:~/DS226/Question2/src/obj$ ./output.out
Parameters have succesfully been read from the file: param.txt
Vector has succesfully been read from the file: vector_a.txt
Vector has succesfully been read from the file: vector_b.txt
Vector has succesfully been written to file: l2Norm.txt
L2 Norm of the error vector: 58.132747
Vector has succesfully been written to file: l2error.txt
Vector has succesfully been read from the file: matrix.txt
Matrix multiplication with vector completed in = 0.146973 seconds
Vector has succesfully been written to file: matvectmult.txt
biswarupk@turing:~/DS226/Question2/src/obj$ export OMP_NUM_THREADS=8
biswarupk@turing:~/DS226/Question2/src/obj$ ./output.out
Parameters have succesfully been read from the file: param.txt
Vector has succesfully been read from the file: vector_a.txt
Vector has succesfully been read from the file: vector_b.txt
Vector has succesfully been written to file: l2Norm.txt