Walchand College of Engineering, Sangli
Department of Computer Science and Engineering

Name: Meet Vipul Gandhi

PRN: 2020BTECS00112

Class: Final Year (Computer Science and Engineering)

Year: 2023-24 **Semester:** 1

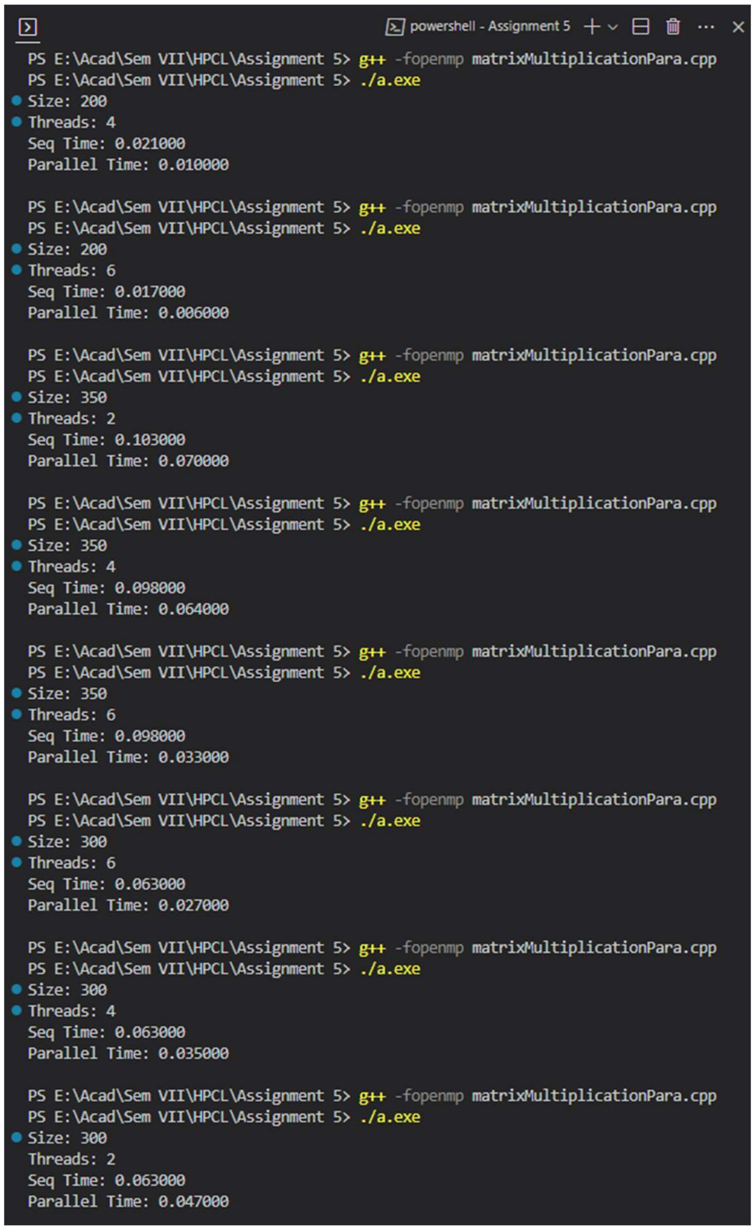
Course: High Performance Computing Lab

Practical No. 5

Title of practical: Implementation of OpenMP programs.

Problem Statement 1: Implementation of Matrix-Matrix Multiplication.

Screenshots:



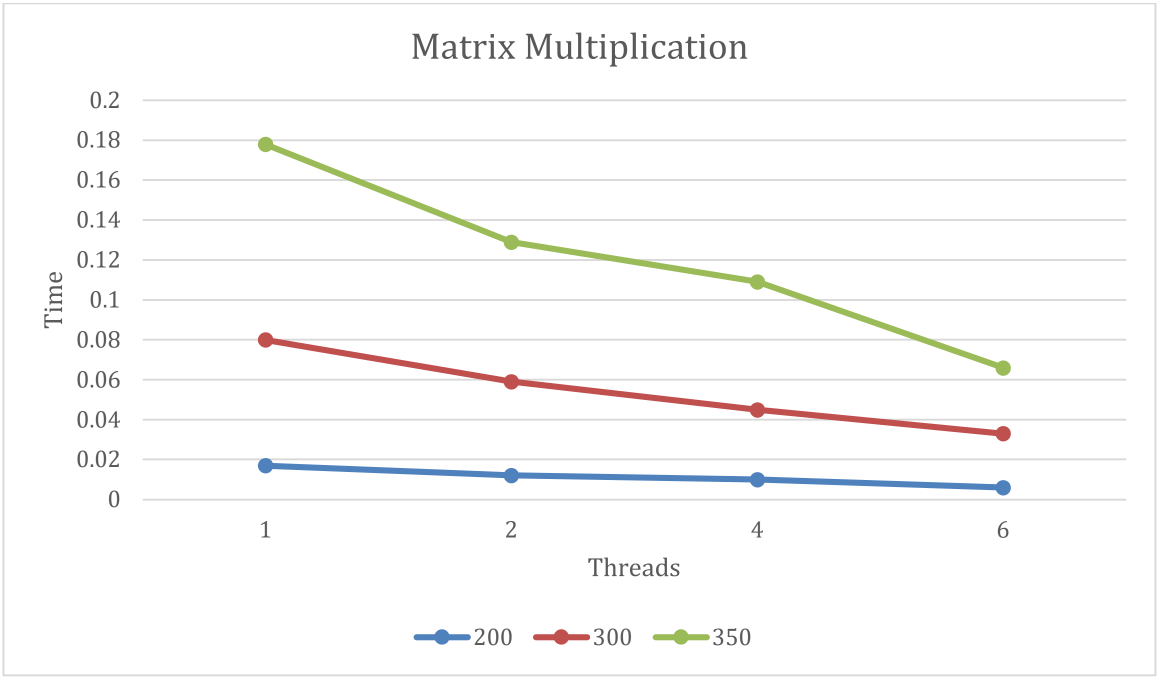
```
PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 200
Threads: 4
Seq Time: 0.021000
Parallel Time: 0.010000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 200
Threads: 6
Seq Time: 0.017000
Parallel Time: 0.006000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 350
Threads: 2
Seq Time: 0.103000
Parallel Time: 0.070000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 350
Threads: 4
Seq Time: 0.098000
Parallel Time: 0.064000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 350
Threads: 6
Seq Time: 0.098000
Parallel Time: 0.033000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 300
Threads: 6
Seq Time: 0.063000
Parallel Time: 0.027000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 300
Threads: 4
Seq Time: 0.063000
Parallel Time: 0.035000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp matrixMultiplicationPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 300
Threads: 2
Seq Time: 0.063000
Parallel Time: 0.047000
```

Information:

a. Matrix Multiplication has very high complexity of O(N^3) so on parallelizing we can reduce the time by a lot.
b. Few variables (I, j, k) are shared privately to each thread whereas the matrices A, B and C result matrix are shared.
c. Sizes are changed along with the number of threads.

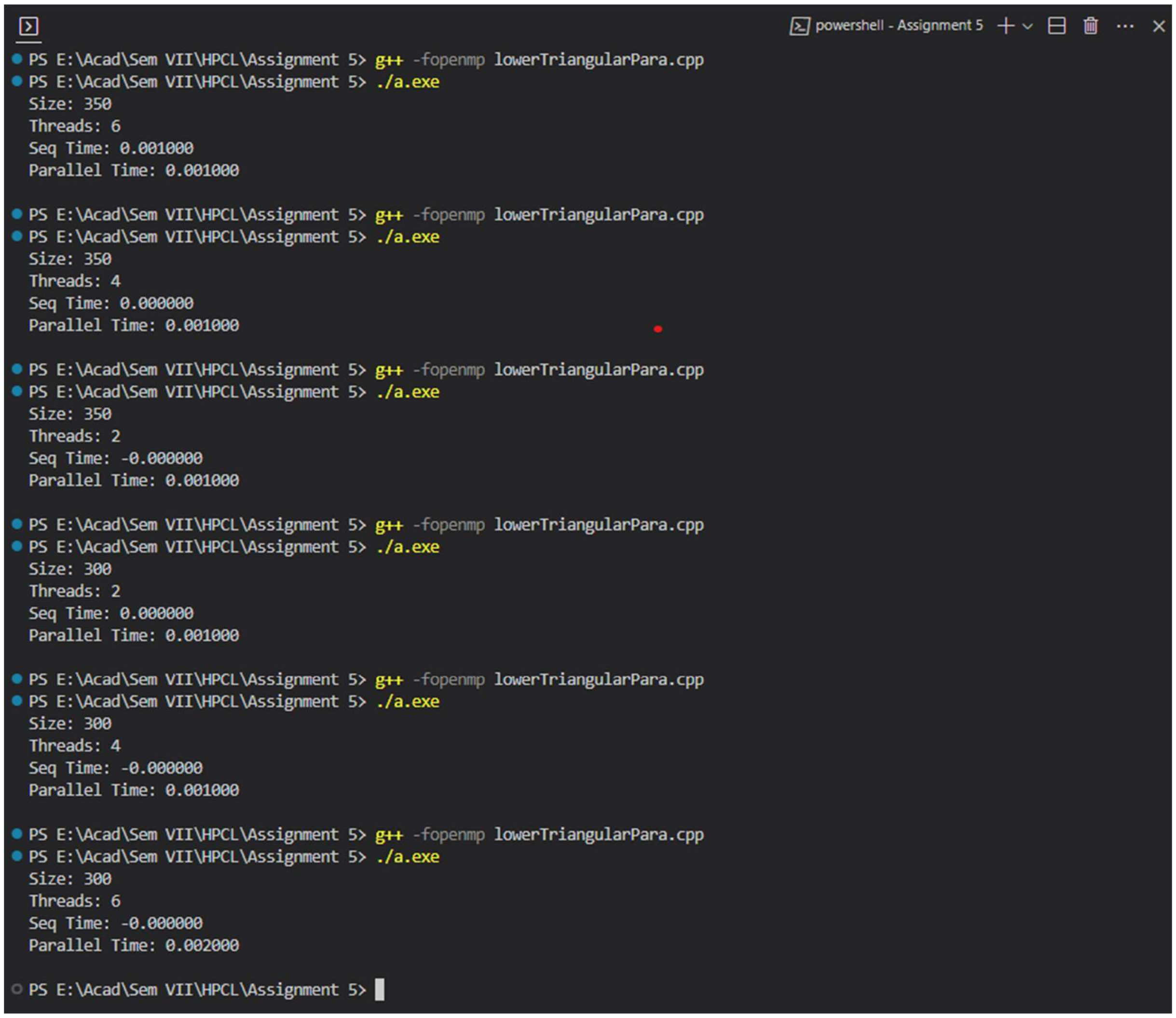
Analysis:

Size / Index	200	300	350
1	0.017	0.063	0.098
2	0.012	0.047	0.07
4	0.01	0.035	0.064
6	0.006	0.027	0.033



Walchand College of Engineering, Sangli
Department of Computer Science and Engineering
Problem Statement 2: Implementation of sum of two lower triangular matrices.

Screenshots:



```
PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp lowerTriangularPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 350
Threads: 6
Seq Time: 0.001000
Parallel Time: 0.001000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp lowerTriangularPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 350
Threads: 4
Seq Time: 0.000000
Parallel Time: 0.001000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp lowerTriangularPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 350
Threads: 2
Seq Time: -0.000000
Parallel Time: 0.001000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp lowerTriangularPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 300
Threads: 2
Seq Time: 0.000000
Parallel Time: 0.001000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp lowerTriangularPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 300
Threads: 4
Seq Time: -0.000000
Parallel Time: 0.001000

PS E:\Acad\Sem VII\HPCL\Assignment 5> g++ -fopenmp lowerTriangularPara.cpp
PS E:\Acad\Sem VII\HPCL\Assignment 5> ./a.exe
Size: 300
Threads: 6
Seq Time: -0.000000
Parallel Time: 0.002000

PS E:\Acad\Sem VII\HPCL\Assignment 5>
```

Information:

a. Lower triangular matrix is a matrix where all the cells above the diagonals are zero.
b. Cells below the diagonal of matrix can be zero as well as non-zero.
c. Just like above problem we used private sharing for (i, j, k) and shared for (A, B, C) for all the threads.
d. Paralleling the above problem doesn't affect much to the execution time.

Analysis: Here we can see that on increasing the matix size and number of threads doesn't result in decrease in execution time. So, for small matrix size parallelizing is not a good option.

GitHub Link: https://github.com/meetgandhi692/HPC-Lab/tree/52fd41746f81d008d559dd997530e81d3f6707d3/Assignment%205