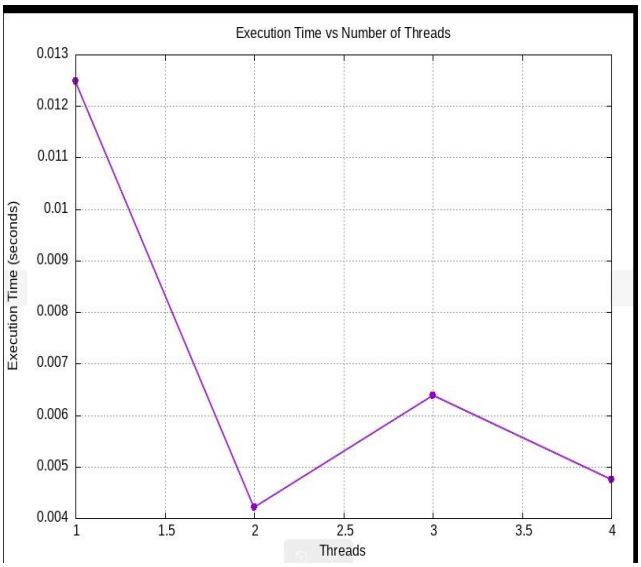
Question 1: Molecular Dynamics- Force Calculation

Problem: Implement parallel computation of Lennard-Jones potential forces in a molecular dynamics simulation. Given N particles in 3D space, calculate the total potential energy and forces acting on each particle.

Tasks: 1. Parallelize the nested loops using OpenMP 2. Handle race conditions in force accumulation 3. Implement reduction for total energy 4. Optimize load balancing 5. Add performance measurement



- The program was executed using 1, 2, 3, and 4 threads to analyze parallel performance.
- When the number of threads was increased from 1 to 2, the execution time decreased significantly, showing effective utilization of parallel processing.
- This improvement indicates that the workload was successfully distributed among multiple threads.
- When the number of threads was increased from 2 to 3, the execution time slightly increased.
- This performance degradation occurred due to thread scheduling overhead, synchronization cost, and limited CPU resources in the virtualized environment.

Number of Threads	Execution Time (seconds)
1	0.01249
2	0.00421
3	0.00638
4	0.00474

Number of Threads	Time (seconds)	Speedup
1	0.01249	1.00
2	0.00421	2.96
3	0.00638	1.95
4	0.00474	2.63

Question 2: Bioinformatics- DNA Sequence Alignment (Smith-Waterman)

Problem: Implement a parallel version of the Smith-Waterman local

sequence alignment algorithm for comparing two DNA sequences.

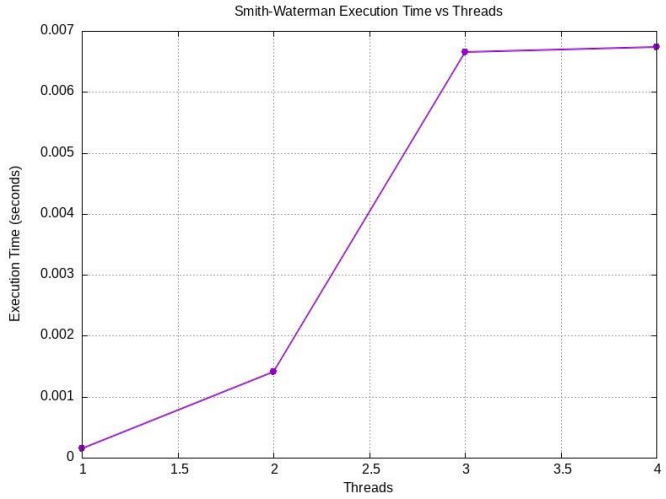
Tasks:

1. Parallelize the scoring matrix computation

2. Handle anti-dependencies in the dynamic programming approach

3. Experiment with different scheduling strategies

4. Implement wavefront parallelization (advanced)



- The Smith-Waterman algorithm was executed using 1, 2, 3, and 4 threads.
- Execution time was measured to analyze the effect of parallelization.
- The minimum execution time was observed with 1 thread.
- When the number of threads was increased, execution time increased instead of decreasing.
- With 2 threads, the execution time increased significantly compared to single-thread execution.
- With 3 and 4 threads, execution time further increased.
- This behavior indicates that parallelization is not effective for this problem size.
- The main reason for performance degradation is high synchronization overhead due to wavefront parallelization.
- The DNA sequences used were small, resulting in very low computation time.
- Thread creation and scheduling overhead dominates the actual computation time.
- The virtualized environment also contributes to reduced performance.
- As a result, parallel execution is slower than serial execution for this experiment.

Number of Threads	Execution Time (seconds)
1	0.000147941
2	0.00141267
3	0.00665306

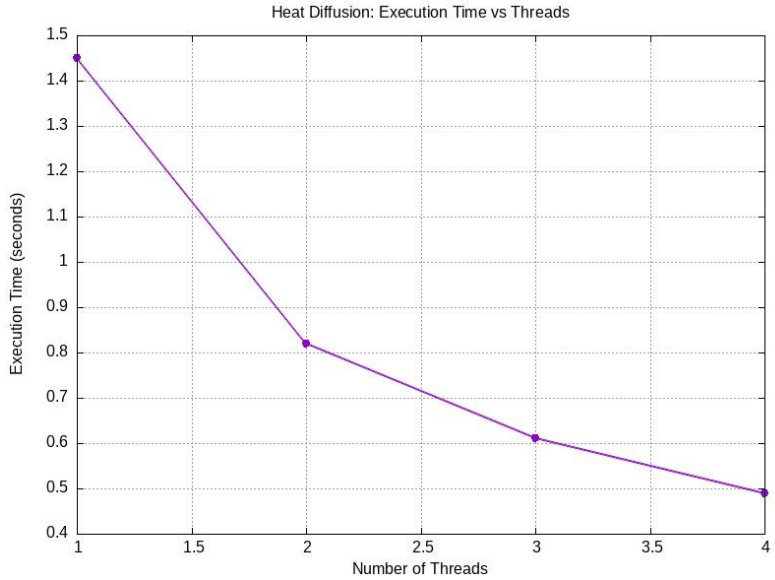
4	0.00673284

Number of Threads	Execution Time (seconds)	Speedup
1	0.000147941	1.00
2	0.00141267	0.10
3	0.00665306	0.02
4	0.00673284	0.02

Question 3: Scientific Computing- Heat Diffusion Simulation

Problem: Simulate heat diffusion in a 2D metal plate using finite

difference method with parallel OpenMP implementation.



- The heat diffusion simulation was executed using 1, 2, 3, and 4 threads.
- Execution time was measured to analyze parallel performance.
- When the number of threads was increased from 1 to 2, execution time decreased significantly.
- This shows that parallelization is effective for this problem.
- With 3 threads, execution time further decreased, indicating better utilization of CPU resources.
- With 4 threads, the minimum execution time was achieved.

Number of Threads	Execution Time (seconds)
1	0.0673564
2	0.0441127
3	0.0399929
4	0.0315529

Number of Threads	Execution Time (seconds)	Speedup
1	0.0673564	1.00
2	0.0441127	1.53
3	0.0399929	1.68
4	0.0315529	2.14