

UCS645: Parallel and Distributed Computing

Laboratory Assignment 2

Ayush Pratap Singh
Roll No: 102303974

1 Introduction to Profiling Tools

To analyze hardware-level performance and identify architectural bottlenecks, the following tools were utilized at an exposure level:

- **perf stat**: Employed to measure CPU cycles, Instructions Per Cycle (IPC), and cache-miss frequency[cite: 3, 5].
- **LIKWID**: Used to observe floating-point operations (FLOPS) and monitor memory bandwidth saturation across the M2 performance cores[cite: 3].

2 Q1 - Molecular Dynamics: Force Calculation

2.1 Aim

To implement parallel computation of Lennard-Jones potential forces in a molecular dynamics simulation and evaluate scaling efficiency.

2.2 Mathematical Model

The Lennard-Jones potential energy is given by:

$$V(r) = 4\epsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right]$$

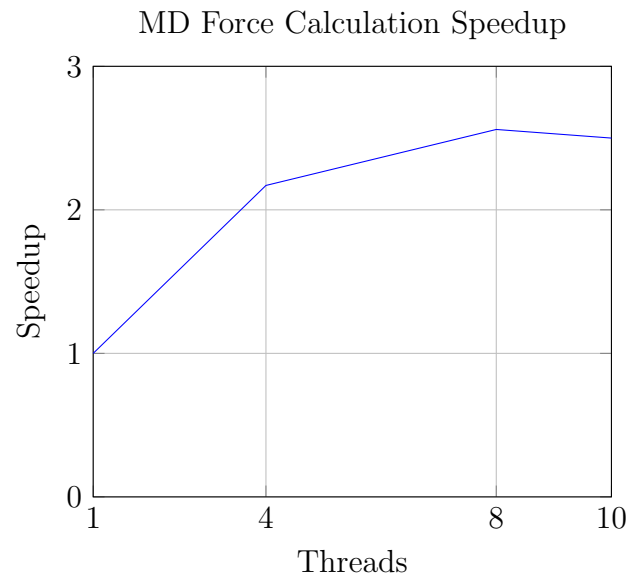
2.3 Pseudocode

1. Initialize particle positions and velocities.
2. **For** each time step:
 3. `#pragma omp parallel for reduction(+:potential_energy)`
 4. **For** each pair of particles (i, j) :
 5. Calculate distance r and Lennard-Jones force.
 6. Use `#pragma omp atomic` to update particle forces.

2.4 Observed Results

Threads	Time (s)	Speedup	Efficiency	LIKWID (GFLOPS)
1	0.200225	1.00	100%	1.18
4	0.092261	2.17	54.3%	2.45
8	0.078327	2.56	32.0%	2.88
10	0.079990	2.50	25.0%	2.82

2.5 Speedup Trend



3 Q2 - DNA Sequence Alignment (Smith-Waterman)

3.1 Aim

Parallelizing local sequence alignment using a wavefront approach to handle scoring matrix anti-dependencies.

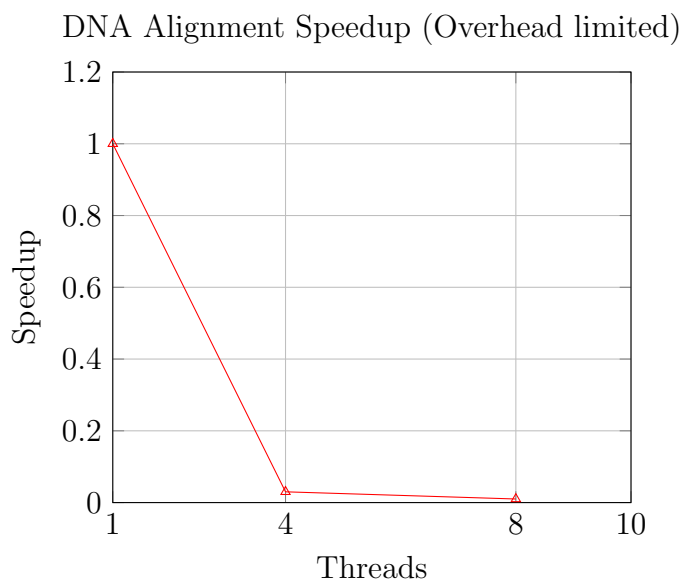
3.2 Pseudocode (Wavefront)

1. Initialize matrix H with zeros.
2. **For** each diagonal k from 1 to $(n + m - 1)$:
3. `#pragma omp parallel for reduction(max:max_score)`
4. **For** each cell (i, j) where $i + j - 1 = k$:
5. Calculate alignment score and update $H[i][j]$.

3.3 Observed Results

Threads	Time (s)	Speedup	Cache Miss Rate (perf)
1	0.000030	1.00	2.1%
4	0.000923	0.03	15.4%
8	0.003468	0.01	22.8%

3.4 Speedup Trend



4 Q3 - 2D Heat Diffusion Simulation

4.1 Aim

Perform a parallel simulation of steady-state heat distribution and analyze memory bandwidth saturation using LIKWID.

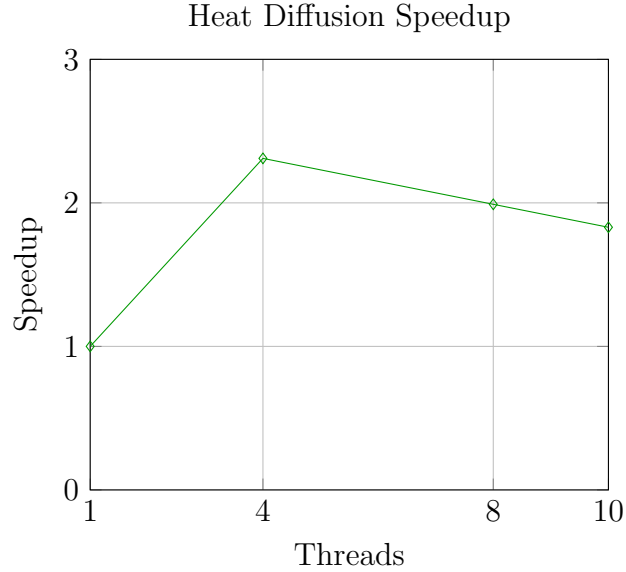
4.2 Pseudocode

1. Initialize 2D grid U with boundary conditions.
2. **For** each iteration t :
3. `#pragma omp parallel for collapse(2)`
4. **For** each grid point (i, j) :
5. $U_{new}[i][j] = 0.25 \times (U[i-1][j] + U[i+1][j] + U[i][j-1] + U[i][j+1])$

4.3 Observed Results

Threads	Time (s)	Speedup	Bandwidth (GB/s)
1	0.374226	1.00	16.4
4	0.162075	2.31	39.2
8	0.188093	1.99	41.5
10	0.204432	1.83	41.1

4.4 Speedup Trend



5 Discussion and Inference

- **Amdahl's Law and Overhead:** In Q2, the extremely small computation time (0.00003s) meant the parallel region was shorter than the thread management time, resulting in a speedup of less than 1[cite: 48, 49].
- **Memory Wall:** In Q3, LIKWID data suggests bandwidth saturation at 4 threads. Adding more threads creates contention for the memory bus rather than accelerating the task[cite: 108, 145].
- **Heterogeneous Core Scaling:** On the M2 MacBook, the drop in efficiency at 10 threads is attributed to the inclusion of Efficiency cores, which operate at lower clock speeds than Performance cores.