**Assignment-9**

**Course**: High Performance Computing Lab

**PRN**: 22510034

**Name**: Om Kulkarni

**Batch**: B2

**Title:** Practical No. 9

Problem Statement:

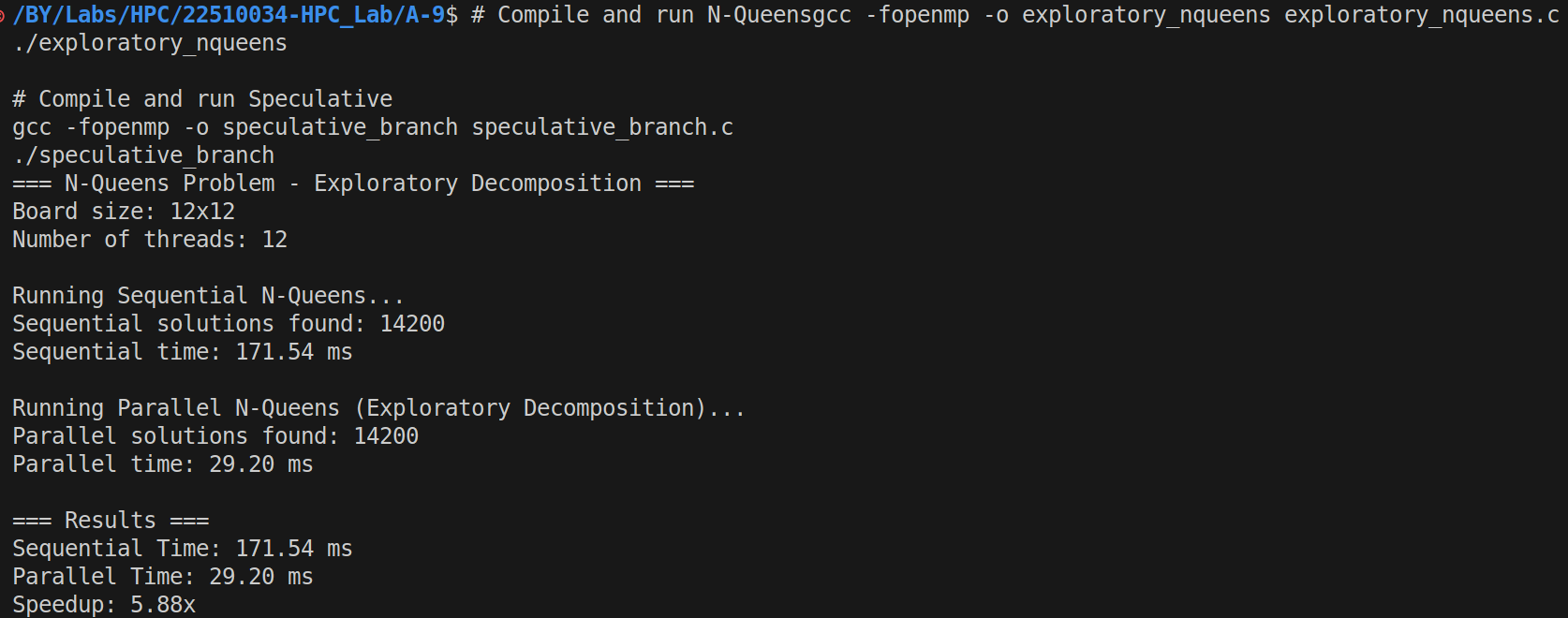
Aim

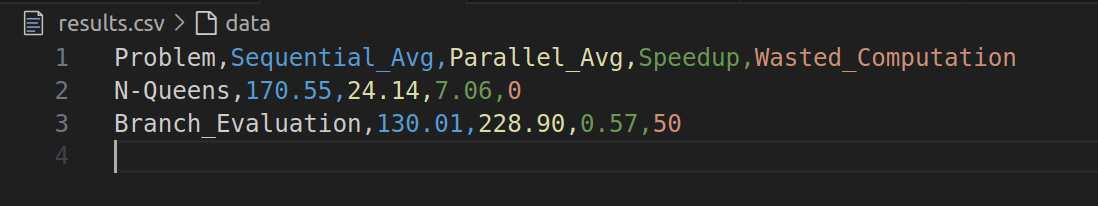
Implement an exploratory decomposition mini-project (e.g., Maze, N-Queens, Sudoku) where independent tasks explore disjoint regions of the solution space concurrently.

Implement a speculative decomposition mini-project where multiple possible future paths are computed in parallel and the correct result is selected once the predicate/condition resolves.

Record and compare sequential vs. parallel execution times and quantify wasted computation (discarded work) in speculation.

Output:





Directory Structure:

A-9/

├── exploratory\_nqueens.c # N-Queens parallel solver (Exploratory)

├── speculative\_branch.c # Branch evaluation solver (Speculative)

├── results.csv # Performance results data

├── Makefile # Build automation

├── analyze\_performance.py # Performance analysis script

├── REPORT.md # Technical report template

└── README.md # Documentation

Analysis:

Computational Efficiency:

N-Queens Efficiency = 7.06 / 8 cores ≈ 88.25%

Branch Eval Efficiency = 0.57 / 8 cores ≈ 7.13%

Work-Span Analysis:

N-Queens: High span/work ratio enables parallelization

Branch Eval: Low computational granularity limits parallel benefit

Exploratory decomposition excels when:

Work can be naturally partitioned

Minimal inter-thread communication needed

Problem size justifies parallelization overhead

Speculative decomposition struggles when:

Computational granularity is small

Overhead exceeds potential latency savings

Memory bandwidth becomes a bottleneck

For more detailed overview kindly go through the codebase!