Final Project Requirements: CS4612 Parallel & Distributed Computing

1) Scope & Goals

Each group completes two linked implementations on the same problem/dataset:
1) Parallel (shared memory) using OpenMP (CUDA optional if GPUs are available).
2) Distributed (message-passing) using MPI.

Focus on correctness → performance → tuning → analysis.
Deliver measurable speedup and efficiency, and explain trade-offs.

Each project group must consist of 4–5 students

2) Tech Stack & Constraints

- Languages: C (required).
- Parallel: OpenMP (required).
- Distributed: MPI (required).
- Optional: CUDA (only if GPUs exist).
- No high-level big data or ML frameworks (Hadoop, Spark, PyTorch, TensorFlow).

3) Dataset Requirements (Open-Source & "Large Enough")

- Use a public, open-source dataset with a permissive license.
- Size guidance:
  - Text: ≥100 MB total (e.g., e-books, Wikipedia subset).
  - Images: ≥50k images or ≥1–2 GB total.
  - Logs/CSV: ≥200 MB.
  - Graphs: ≥1M edges.
- Provide dataset citation/link in README.

Suggested sources:
- Text: Project Gutenberg, Wikipedia dumps.
- Images: CIFAR-10/100, Kaggle datasets.
- Logs: NASA HTTP logs, Airline delays, NOAA weather.
- Graphs: SNAP datasets (YouTube, LiveJournal, OpenFlights).

4) Functional Requirements

- **Implement serial baseline, parallel (OpenMP), and distributed (MPI) versions**.
- **Same problem solved twice: shared-memory vs. distributed.**
- **Program must accept command-line arguments** (dataset path, thread/process counts, tuning options).

5) Performance & Tuning Requirements

• OpenMP: vary threads, scheduling (static/dynamic/guided), chunk size; explore tiling, reductions.
• MPI: vary #processes, partitioning (row, block-cyclic), communication (blocking/non-blocking).
• GPU (optional): vary block/grid size, overlap transfers.
• Strong scaling (fixed size, more resources) and Weak scaling (grow size with resources).
• Metrics: runtime, speedup, efficiency, throughput.

6) Experimental Methodology (Reproducible)

• **Report hardware specs (CPU, cores, RAM, OS, compiler).**
• **≥3 trials per experiment; report mean ± std.**
• Provide scripts for builds and runs.

7) Deliverables

1) Code repository: serial/, parallel_omp/, distributed_mpi/, (gpu_cuda/ if used).
  - README with dataset link, build/run instructions.
  - Scripts to reproduce experiments.
2) Report (6–8 pages): Problem & Dataset, Design, Tuning, Results, Analysis.
3) Presentation (10–12 min): Problem & Dataset + graphs(results/tuning/analysis) + lessons learn.

8) Milestones & Timeline (suggested)

Week 3: **due on Sep 15 midnight**

- Submit a one-page **project proposal**:
  - Team members.
  - Chosen problem/application.
  - Selected open dataset (with link and size).
- Instructor feedback on feasibility.

Week 4

- Implement **serial baseline**.
- Verify dataset preprocessing/loading and correctness.

Week 5–6

- Develop **parallel (OpenMP)** version.
- Run first tests with different thread counts.
- Begin tuning (scheduling, chunk size, reductions).

Week 7–8

- Develop **distributed (MPI)** version.

- Validate correctness across multiple processes.
- Run initial scaling experiments.

 Week 9–10

- Conduct **advanced tuning**:
  - OpenMP: threads, schedule, chunk sizes.
  - MPI: process counts, blocking vs. non-blocking, collectives.
- Perform **strong scaling** experiments.

 Week 11–12

- Perform **weak scaling** experiments (dataset grows with processes).
- Compare OpenMP vs. MPI performance.
- Submit **progress report** with preliminary graphs.

 Week 13

- Draft **final report** (problem, dataset, design, experiments, results).
- Peer review / instructor feedback.

 Week 14

- Finalize code, results, and report.
- Prepare **presentation slides**.

 Week 15

- Submit **final code + report**.
- Deliver **presentations** in class.

9) Evaluation Rubric (100%)

• Problem/Dataset choice – 10%
• Correctness & code quality – 25%
• Experimental design & reproducibility – 20%
• Tuning & performance analysis – 30%
• Report & presentation – 15%

10) Academic Integrity, Ethics & Data Handling

• **Cite datasets and tools properly.**
• **Use only public, legally redistributable data.**
• **No personal/PII data.**

• If you use any external resources (e.g., tutorials, open-source snippets, libraries), you must:

1. **Cite the source properly in your report.**

2. **Clearly comment in the code which parts are adapted or inspired.**

Failure to properly cite and identify external code will result in a grade of ZERO for the project.

Example Project Problems:

1. Word Frequency Analysis

- **Problem**: Count the frequency of words across large text files.
- **Parallel (OpenMP)**: Split a file into chunks, use threads to count words, then combine.
- **Distributed (MPI)**: Each process handles a subset of files, then merges results via reduction.
- **Dataset**: Project Gutenberg e-books (~100MB+).

2. Image Filtering

- **Problem**: Apply filters (grayscale, blur, edge detection) to a large image dataset.
- **Parallel (OpenMP)**: Use threads to process image pixels in parallel.
- **Distributed (MPI)**: Divide images across processes, process locally, send results back to master.
- **Dataset**: CIFAR-10 or [Kaggle image sets] (~1–2GB).

3. **Flight Delay Statistics**

- **Problem**: Compute average delays by airline, airport, or route for a long duration of a year or two years.
- **Parallel (OpenMP)**: Threads parse different slices of a CSV file.
- **Distributed (MPI)**: Processes analyze different years/months of flight data.
- **Dataset**: U.S. Airline On-Time Performance (~500MB, can subset).