Project Proposal

CSC 505 (003) – Design and Analysis of Algorithms

Fall 2025

Katerina Vilkomir

Research Project (Reproduction Study)

Title: *Reproducing and Extending "Comparative Analysis of Sorting Algorithms: A Review" (IEEE ISCMI 2024)*

1. Motivation and Goal

Sorting is a foundational topic in algorithm design and analysis. Understanding how different sorting algorithms behave on real data is essential for evaluating theoretical and practical efficiency. This project aims to **reproduce** and **partially extend** the results from the IEEE paper *"Comparative Analysis of Sorting Algorithms: A Review"* by Ala'anzy et al. (2024). The original paper experimentally compares 13 sorting algorithms and concludes that Quick Sort, Merge Sort, and Radix Sort perform best for large datasets, while quadratic algorithms such as Bubble Sort and Selection Sort degrade sharply as input size grows. My goal is to **verify these findings through independent experimentation** and explore one small extension: how algorithm performance changes under different input distributions.

2. Scope and Objectives

- **Reproduce** runtime and scalability results for representative algorithms:
    - Quadratic sorts: Bubble, Insertion, Selection
    - Log-linear sorts: Quick, Merge, Heap, Shell
    - Linear sorts: Counting, Radix
- **Test datasets** of sizes n = 100, 1 000, 10 000, 100 000 integers (as in the paper).

- **Compare** results with the published CPU-time data (Table IV and Figures 1–4 in the paper).
- **Extend(if have time)** the analysis by measuring performance on:
    - random arrays
    - nearly sorted arrays (5 % elements shuffled)
    - reverse-sorted arrays
    - few-unique arrays (32 distinct values)
- **Possible extension:** include Python's built-in TimSort as a modern hybrid baseline.

3. Methodology

- **Implementation Language:** Python 3.12
- **Environment:**
    - Device: **MacBook Air (2025)**
    - **Chip:** Apple M4
    - **Memory:** 32 GB unified memory
    - **OS:** macOS Sequoia
- **Measurement Tools(from the paper):**
    - time.perf_counter_ns() for precise timing
    - Median of 10 runs per algorithm/size/distribution
    - Correctness check with assert sorted(output)==output
- **Data range:** integers from 0 to 10^6 (to suit Counting/Radix Sort).
- **Output:** tables and plots of runtime vs input size and distribution.

4. Expected Results

- Confirm that Quick Sort, Merge Sort, and Radix Sort exhibit the best scalability.

- Observe quadratic growth for Bubble, Selection, and Insertion Sorts.
- Validate the trade-off between time and space for Counting and Radix Sorts.
- Show how input order affects performance of Quick and Shell Sort.

5. References

Ala’anzy, M. A., Mazhit, Z., Ala’anzy, A. F., Algarni, A., Akhmedov, R., & Abdiyev, B. (2024). *Comparative Analysis of Sorting Algorithms: A Review.* In **11th International Conference on Soft Computing & Machine Intelligence (ISCMI)** (pp. 88–100). IEEE.