CSEE W4824 – Computer Architecture
Homework Assignment 4

Assigned: November 7 2024, Due: December 4 2024, 11:45 pm ET

Collaboration Guidelines: Students may work in groups of 4. Please follow all Columbia academic guidelines with respect to academic integrity. Please keep a log of what days/times you worked on the project, if both team members were present, and what specific work item was completed. Please submit this log along with your homework. Additionally, ensure that any code or data involved with this homework is also submitted.

Submission Guidelines: One submission per group of 4, submitted through Courseworks. Please attach any generated datasets or written code as part of your submission.

Objective

Your task is to implement an efficient, stable sorting algorithm for large in-memory datasets (typically in gigabytes) consisting of either 32 bit integers or floating-point numbers. The primary goal is to optimize for cost-efficiency, measured as the total cost of sorting per gigabyte.

You are expected to:

- Develop an efficient and stable sorting algorithm that operates on data pinned in memory (DRAM).
- Tailor your implementation to leverage specific features of the processor and memory system available on either a chosen Google Cloud Platform (GCP) instance or on hardware you have access to. Please include your GCP instance type in your report.
- Optimize the cost of sorting per GB by minimizing runtime, selecting appropriate hardware, and using architectural insights.

Assignment Details

Algorithm Selection and Data Structure Design

- Choose a stable sorting algorithm that meets the in-memory constraints of large data volumes. You may consider algorithms like Merge Sort, Timsort, Radix Sort, or other appropriate stable sorting methods.
- Select data structures that minimize memory footprint and optimize for cache locality, given the processor's cache hierarchy and DRAM characteristics.

Questions to consider:

- What are the tradeoffs of the algorithm(s) you selected in terms of speed, memory usage, and stability?
- How do the chosen data structures support efficient data access patterns for large datasets?

Processor and Memory System Optimization

Leverage architectural features of the processor and memory system to improve sorting performance. This may include:

- Optimizing for memory system configuration, cache usage, minimizing memory latency.
- Taking advantage of parallelism if multi-core processors.
- Using special-purpose instructions, etc.
- Utilizing any programming language, libraries, or even hand-optimized assembly code.
- Applying machine learning techniques to select the optimal processor instance for sorting based on data characteristics.
- Reimplementing algorithms from research papers for sorting.

Questions to consider:

- What architectural features are you exploiting to optimize performance? How do they enhance sorting speed or reduce memory usage?
- If using a GCP instance, what is your rationale for choosing a particular instance type, considering both performance and cost?

Performance Measurement and Cost Calculation

- Measure the runtime and memory usage of your sorting implementation.
- Based on these measurements, calculate the cost of sorting per GB using the pricing structure of your chosen hardware or GCP instance. For example, if an instance costs $25/month, use this rate to calculate the cost per GB based on your sorting time.

Questions to consider:

- How did you determine and calculate the total sorting cost per GB?
- What performance metrics did you gather (e.g., execution time, CPU cycles, memory usage), and how did they influence your final cost calculation?

Testing and Validation

- Test your sorting algorithm on sample datasets of varying sizes and characteristics to ensure scalability and correctness.
- Document any challenges or limitations encountered during testing and how you addressed them.

Analysis and Report

Write a report explaining:

- Your choice of algorithms and data structures.
- The optimizations applied based on processor and memory characteristics.
- A breakdown of performance measurements and final cost per GB calculations.
- A post-mortem analysis that reflects on the effectiveness of your implementation, highlights any potential areas for future improvement, and discusses any observed limitations.

Questions to consider:

- Why do you believe your algorithm and data structures were well-suited for this task?
- What insights did you gain from the post-implementation analysis regarding your algorithm's efficiency and cost-effectiveness?

Submission Requirements

- **Code:** Submit the source code (named sorting.c in your Courseworks submission) of your implementation and test cases. Please document your code and include comments for readability.
- **Report:** Provide a detailed report (3-5 pages) covering the points outlined above. Your report should be written in the format of a technical report and include at least the following sections: introduction, methodology, results, discussion, and conclusion. Do not exceed 5 pages.
- **Cost Analysis:** Include a clear calculation of the cost per GB for sorting based on your measurements and instance pricing.

Evaluation Criteria

- **Correctness and Stability (5 pts):** The implementation must produce a stable sort and correctly handle large datasets.
- **Cost-Efficiency (10 pts):** Minimization of the sorting cost per GB will be a primary metric for evaluation. Include graphs and data to show that your implementation is cost-effective for varying input sizes. Include your calculation of the cost per GB with and without optimizations.
- **Optimization and Innovation (10 pts):** Effective use of processor/memory architecture and any innovative techniques for performance improvement will be highly valued. Please answer the questions provided in the assignment in your report and explain any architectural design choices you made in your implementation. Your report must include a detailed discussion of at least three optimization strategies. Include data and graphs to support your design choices.
- **Report Quality (5 pts):** Clarity, depth of analysis, and thoroughness in the final report.