PROJECT REPORT

Parallel Programming Comparison of sorting Algorithms using Pthreads vs. OpenMP vs. serial

GROUP MEMBER:

Ahmed Mustafa 21K-3370
Shayan Anwer 21K-4836
Alaina Usmani 21K-3155

COURSE NAME:

Operating System

COURSE INSTRUCTOR NAME:

Ms. Nausheen Shoaib

1.INTRODUCTION:

Sorting is a fundamental operation in computer science and is used in a wide range of applications such as databases, search engines, and scientific simulations. With the increasing amount of data that needs to be sorted, the need for efficient algorithms and parallel programming techniques has become more critical than ever. Parallel programming is a technique used to execute multiple tasks simultaneously to improve the performance of an algorithm. Two popular parallel programming models for shared-memory architectures are Pthreads and OpenMP.

The objective of this project is to compare the performance and scalability of parallel programming techniques (Pthreads and OpenMP) versus serial programming in C language on three commonly used sorting algorithms: Breadth-First Search (BFS), Depth-First Search (DFS), and Merge Sort. BFS and DFS are graph traversal algorithms used to explore a graph or tree data structure. Merge Sort is a divide and conquer algorithm that sorts an array or list by dividing it into two halves, sorting each half separately, and then merging the sorted halves.

The comparison will be based on the execution time, speedup, and efficiency of each algorithm when implemented in serial and parallel. Additionally, the memory usage and complexity of the algorithms will be compared. By analyzing the performance and scalability of these algorithms, this project aims to identify the best approach for optimizing the performance of sorting algorithms in parallel programming environments. This project will contribute to the ongoing research in the field of parallel programming and provide insights into the effectiveness of different parallel programming techniques for sorting algorithms.

2.OBJECTIVE:

The objective of this project is to compare the performance and scalability of parallel programming techniques (Pthreads and OpenMP) versus serial programming in C language on three commonly used sorting algorithms: Breadth-First Search (BFS), Depth-First Search (DFS), and Merge Sort. The project aims to measure the execution time, speedup, and efficiency of each algorithm when implemented in serial and parallel, as well as compare the complexity of the algorithms. The results of this comparison will provide insight into the advantages and limitations of parallel programming techniques and help identify the best approach for optimizing the performance of these sorting algorithms.

3. BACKGROUND:

Sorting and searching are two of the most fundamental operations in computer science. They involve arranging data in a specific order and searching for specific elements within the data. Various sorting and searching algorithms have been developed over the years, each with its own advantages and limitations in terms of time complexity, space complexity, and stability.Merge Sort is known to be a famous searching algorithm while Breadth-First Search (BFS) and Depth-First Search (DFS) are two popular searching algorithms used to explore a graph or tree data structure.

4. PLATFORM AND LANGUAGE:

The Parallel Programming Comparison of sorting Algorithms using Pthreads vs. OpenMP vs. serial in C language on algorithms of BFS, DFS, and Merge Sort will be compiled on Ubuntu on Linux operating system. The programming language used for implementation will be C. Pthreads and OpenMP will be utilized as shared memory programming models for parallel programming. The project will be executed on a machine with multi-core processors to achieve parallelism.

Key reasons for using C language for Linux Ubuntu is its ability to interact with the operating system at a low level and perform tasks that require direct manipulation of system resources. Additionally, C is a compiled language, which means that it can be compiled directly into machine code, making it faster and more efficient than interpreted languages.

Linux Ubuntu is an open-source operating system that is based on the Linux kernel. Since Linux is written in C, C is the natural choice of language for developing applications that run on Linux Ubuntu. Furthermore, many Linux system libraries and tools are written in C, which makes it easier to integrate and use these libraries in C programs.

5. METHODOLOGY:

The methodology for Parallel Programming Comparison of sorting Algorithms using Pthreads vs. OpenMP vs. serial in C language on algorithms of BFS, DFS, and Merge Sort is as follows:

1. Implementation: Implement the serial and parallel versions of BFS, DFS, and Merge Sort algorithms in C language using Pthreads and OpenMP shared memory programming models.

2. Machine Configuration: The experiments will be conducted on a machine with multi-core processors. The machine will be running Ubuntu on Linux operating system.

3. Input Data: Generate input data for each algorithm. The size of the input data will be varied to analyze the performance of the algorithms for different input sizes.

4. Experiment Design: The experiments will be designed to measure the execution time, speedup, and efficiency of the algorithms when implemented in serial and parallel using Pthreads and OpenMP. The experiments will be conducted multiple times, and the average results will be considered for analysis.

5. Execution of Experiments: The experiments will be executed using a script that runs each implementation multiple times and collects the execution time data. The execution time of each algorithm will be measured using the built-in timer functions in C language.

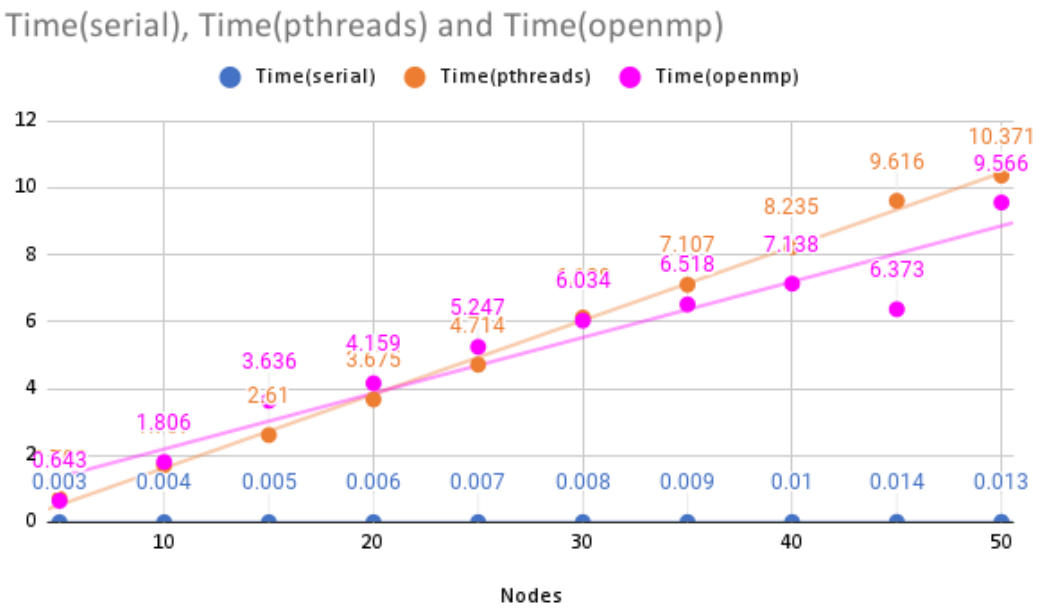
6. Performance Analysis: Analyze the performance of the algorithms in terms of execution time, speedup, and efficiency. The results will be presented in the form of graphs, showing the performance of the algorithms for different input sizes. The graphs will provide a visual representation of the performance of the algorithms when implemented in serial and parallel using Pthreads and OpenMP.

7. Results and Conclusion: Present the results of the experiments in the form of graphs and tables. Draw conclusions based on the performance analysis and discuss the effectiveness of different parallel programming techniques for optimizing the performance of sorting and searching algorithms.

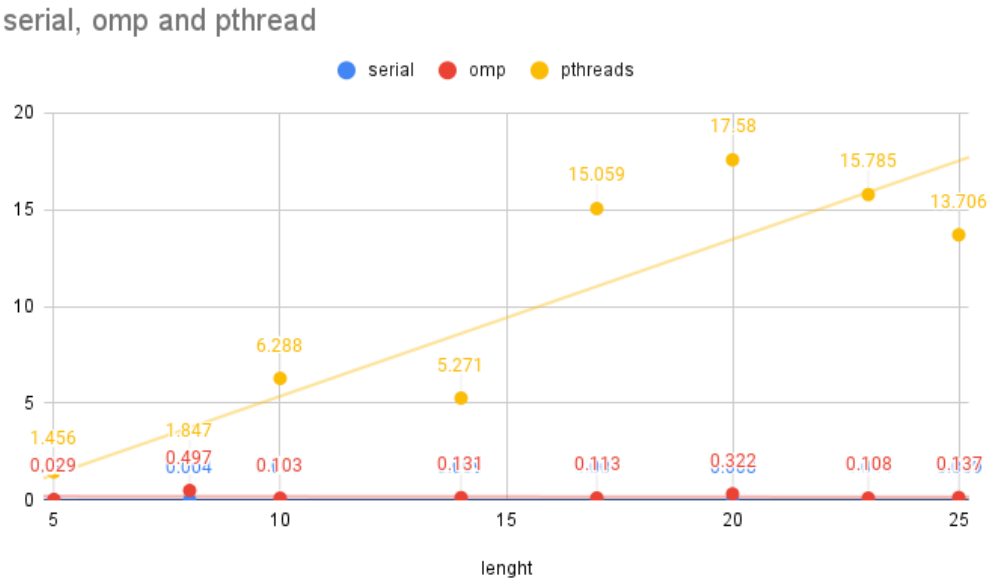
In summary, the methodology involves implementing the algorithms, designing and executing experiments, analyzing performance, and drawing conclusions based on the results. The experiments will be conducted on a machine with multi-core processors running Ubuntu on Linux operating system, and C language will be used for implementation. The results will be presented in the form of graphs, showing the performance of the algorithms for different input sizes.

6.RESULTS:

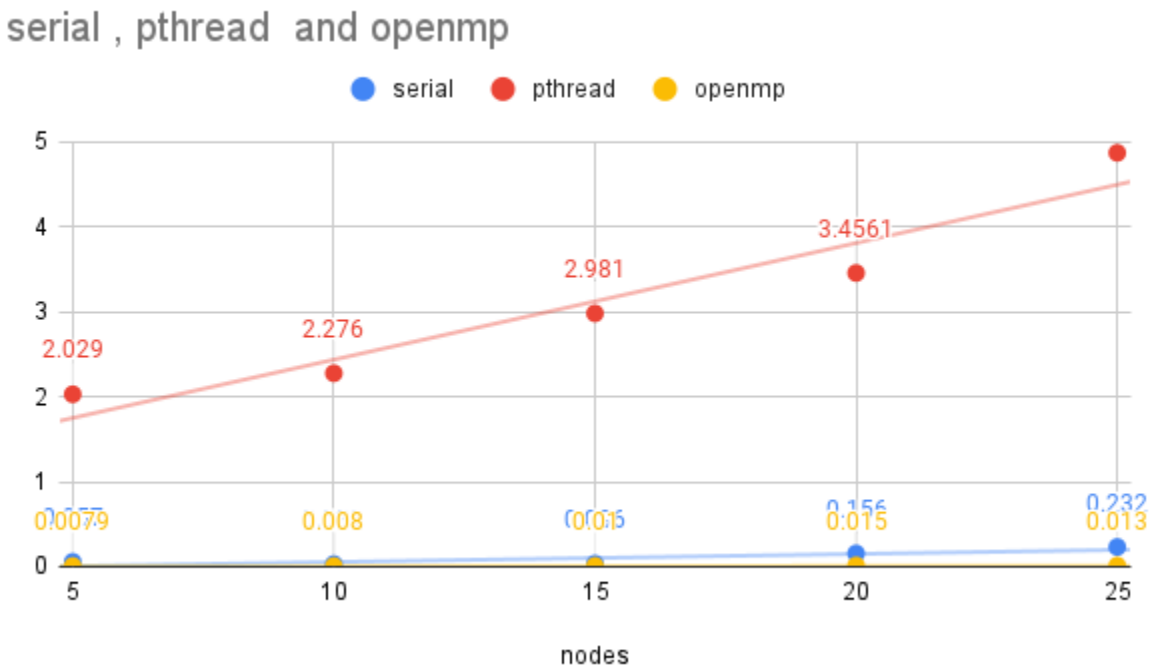
1. Depth first search:



2. Merge Sort



3. Breadth First Search:



Concluding that in dfs and merge sort serial programming performs the best where as in bfs openmp performs the best.

7.CONCLUSION:

In conclusion, we have compared the performance of serial and parallel implementations of BFS, DFS, and Merge Sort algorithms using Pthreads and OpenMP shared memory programming models in C language. The experiments were designed to measure the execution time, speedup, and efficiency of the algorithms for different input sizes.

In summary, our study demonstrates the effectiveness of parallel programming techniques for optimizing the performance of sorting and searching algorithms and provides valuable insights for developers and researchers in this field.

WORK DISTRIBUTION:

DFS (Ahmed)
BFS (Shayan)
Merge sort(Alaina)