**National University of Computer &  
Emerging Sciences Karachi Campus**  
  
  
***Parallel Programming Comparison of sorting Algorithms using Pthreads vs. OpenMP vs. serial***

**Project Proposal  
 Operating System  
Section: BSE-4B**

**21K-3839 Maham  
 21K-3836 Hoor  
 21K-3906 Aiman  
 21K-3907 Huda**  
  
**Introduction**

Sorting is a fundamental operation in computer science that is used extensively in various applications such as databases, search engines, and data analytics. Sorting refers to arranging a set of data elements in a particular order, such as ascending or descending order. The most common algorithms used for sorting are Quicksort, Mergesort, Heapsort, and Bubble sort. These algorithms have varying time complexities, but they all have a worst-case time complexity of O(n^2).

As the size of the data increases, the time required for sorting also increases, making it a computationally expensive task. Parallel programming is a technique that can be used to reduce the sorting time by utilizing multiple processors or cores simultaneously. Parallel programming refers to the process of breaking down a task into smaller subtasks that can be executed simultaneously by multiple processors or cores. Parallel programming can result in significant speedups for computationally intensive tasks, such as sorting.

The motivation behind this project is to investigate how parallel programming can be used to speed up the sorting process, and to determine which parallel programming paradigm is the most efficient for each algorithm. The results of this project can help in choosing the best parallel programming approach for sorting large datasets, which can lead to significant improvements in the performance of various applications that rely on sorting. The four sorting algorithms that are implemented in this project have varying time complexities and are suitable for different scenarios. Quicksort is a divide-and-conquer algorithm that recursively partitions the array into smaller sub-arrays based on a chosen pivot element. Mergesort is also a divide-and-conquer algorithm that recursively divides the array into smaller sub-arrays and then merges them back in sorted order. Bubble sort is a simple comparison-based sorting algorithm that repeatedly iterates through the list, compares adjacent elements and swaps them if they are in the wrong order. Both Quicksort and Mergesort have a worst-case time complexity of O(n log n), while Bubble sort has a worst-case time complexity of O(n^2).

**METHODOLOGY**

The project aims to compare the performance of four sorting algorithms (Quicksort by High, Quicksort by Low, Merge sort, Bubble sort) using three parallel programming paradigms (Pthreads, OpenMP, and Serial). The objective is to determine which parallel programming approach is the most efficient for each algorithm and to assess the impact of parallelization on sorting large datasets. This comparison will provide insights into the benefits and limitations of different parallel programming paradigms for sorting tasks. Implement the four sorting algorithms (Quicksort by High, Quicksort by Low,Bubble sort ,Merge sort) in each of the parallel programming paradigms (Pthreads, OpenMP, and Serial).That the sorting algorithms are correctly implemented and produce the expected sorted output. Handle edge cases and handle scenarios where the dataset size exceeds the available memory. We Set up a computer system with multiple processor cores/threads to execute the parallel programs. That the system is properly configured