*Project Proposal OS*

***Title:***

***Parallel Programming – Comparison of Sorting Algorithms using Pthreads vs. OpenMP vs. Serial, (6 algorithms)***



**Group Members:**

1. Mawiz Khan (19K-1408)

**Introduction:**

This project will compare the use of Pthreads, OpenMP and Serial by analyzing the performance for some algorithms written in C. Below are:

1. Quick Sort
2. Merge Sort
3. Insertion Sort
4. Bubble Sort
5. Selection Sort
6. Shell Sort

**Methodology:**

We will compare the three methods using these sorting algorithms:

1. **Quick Sort**

Quicksort is an in-place sorting algorithm. When implemented well, it can be somewhat faster than merge sort and about two or three times faster than heapsort.

1. **Merge Sort**

Merge Sort is an efficient, general-purpose, and comparison-based sorting algorithm.

1. **Insertion Sort**

Insertion sort is a simple sorting algorithm that builds the final sorted array one item at a time.

1. **Bubble Sort**

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order.

1. **Selection Sort**

The Selection Sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from the unsorted part and putting it at the beginning.

1. **Shell Sort**

Shell Sort is mainly a variation of Insertion Sort. In insertion sort, we move elements only one position ahead. When an element has to be moved far ahead, many movements are involved. The idea of shellSort is to allow exchange of far items.

**Description:**

In this work we supply several approaches of parallelization of sorting algorithms. There are a lot of sorting algorithms that differ from each other by their effectiveness, resource usage etc. We show three different types of parallelization for sorting algorithms-parallelization through algorithm modification, multithreading, OpenMP. In this work as an example we represent and develop parallel versions of quick sort, insertion sort and merge sort, bubble sort, selection sort and shell sort algorithms. We also compare the speedup for each parallel version and find out a more efficient parallel algorithm. In this work we use C programming language on Ubuntu Linux OS for developing our algorithms. We will apply these algorithms on large data.