Computer Systems Engineering - I

Course Assignment 04

Naren Akash, R J
2018111020

Problem 01: Concurrent Quicksort

Quicksort is a *Divide and Conquer* sorting algorithm. It is one of the most efficient sorting algorithms and is based on the idea of splitting an array into smaller ones. The algorithm first picks an element called a pivot and partitions the array into a low subarray, elements smaller than the pivot and a high subarray, elements greater than the pivot. This step is called the partitioning step.

Quicksort algorithm recursively applies the above step to the sub-arrays, resulting in sorting the array. In this task, you are given the number of elements in the array N and the elements of the array. You are required to sort the numbers using a Concurrent version of Quicksort algorithm.

Tasks

1. Implement a Concurrent version of Quicksort algorithm
2. Make a detailed report
    a. Explains your implementation for the Concurrent Quicksort algorithm
    b. Compares the performance of Concurrent Quicksort with normal Quicksort

Bonus Task

1. Implement a variant of the *Concurrent Quicksort* algorithm, which uses threads instead of processes.
2. Compare the performance of this variant of the algorithm with the ones implemented as a part of the above-mentioned tasks.

Instructions

- The median of the subarray should be chosen as the pivot. It is also allowed to choose a random element from the subarray as your pivot.
- Partition the array around the pivot such that all the elements with a value less than the pivot are positioned before it, while all the elements with a value greater than the pivot are positioned after. In case of equality, they can go on either side of the partition.
- Recursively make two child processes. One of which will sort the low subarray and the other will sort the high subarray.
- When the number of elements in the array for a process is less than 5, perform an Insertion sort to sort the elements of that array.
- Use the shmget, shmat functions for accessing the shared memory.