# Michael Chillemi

# 07/02/2023

# CS 590 - Algorithms

# M7.B1: Module 7 Sorting and Selection Reinforcement Exercises

Problem 8.5.4

Suppose we modify the deterministic version of the quick-sort algorithm so that, instead of selecting the last element in an n-element sequence as the pivot, we choose the element at index [n/2], that is, an element in the middle of the sequence. What is the running time of this version of quick-sort on a sequence that is already sorted?

Answer:

The quicksort algorithm's worst time complexity occurs when the elements are sorted and is . The elements must be sorted for this to operate because the center element is being used as a pivot point. To finish this, follow these steps: Pick a pivot point first. The pivot value will therefore be the value of the middle element, albeit it can be any value that falls within the range of the sorted values, even if it isn't present in the array. Then, reorder the components so that all elements smaller than the pivot are placed on the left side of the array, and all elements larger than the pivot are placed on the right side. Any location in the array can retain values that are equal to the pivot. The array may be partitioned into non-equal pieces, as you can see. Finally, recursively apply the quicksort method to the left and right components.

Problem 9.5.1

Which, if any, of the algorithms bubble-sort, heap-sort, merge-sort, and quick-sort are stable?

Answer:

Maintain the relative order of records with equal keys in a stable sorting algorithm. That example, if two records R and S share the same key and R appeared before S in the original list, R will appear before S in the sorted list, indicating that the sorting algorithm is stable. Insertion sort, merge sort, and bubble sort are stable algorithms that fall in the aforementioned list of algorithms. However, the algorithms heap sort and quicksort are equally unstable.