Flori Kusari
Algorithms and Data Structures (Week 7)

Assignment 7

20th March 2024

Problem 7.1:

Implement 3 versions of the Quicksort algorithm with 3 different versions of the partition algorithm:

(a) (2 points) Lomoto partition as on the lecture slides (Lecture 16).

(b) (2 points) Hoare partition (use Wikipedia or other sources).

(c) (2 points) ”Median-of-three” partition (use Wikipedia or other sources).

(d) (4 points) Measure the running times of the 3 Quicksort versions from above for the same

100000 randomly generated sequences of fixed length 1000, compute the average running

times for each of the 3 versions and compare them. Explain the behavior of the 3 versions

and your observations.

Answers

A. For a see the respective files attached.
B. For b see the respective files attached.
C. For c see the respective files attached.
D. Median Version: This version theoretically chooses the most appropriate pivot such that the algorithm is more efficient (thus avoiding the case where the pivot is the smallest or largest element), it seems to fall flat due to how much time it takes to select that pivot, which is why it performs the worst out of the three. Overall, it has a great logic and if the pivot was to be chosen quicker it would potentially be the best, however, since it does not we know that it is the worst.

   Lomoto Version: This version doesn't focus on avoiding the selection of a bad pivot, however, it is still pretty efficient, outperforming the Median Version since it doesn't spend any time trying to calculate the best element for being the pivot. In essence, less focus on being perfect makes it outperform the Median Version.

Hoare Version: Among the three versions evaluated, the Hoare partitioning method emerges as the most efficient. This efficiency can be directly linked to its pivot selection process, which is both rapid and effective at averting undesirable pivot choices. In essence, requires minimal computation which allows it to be the best out of the three.

Problem 7.2:

(a) (6 points) Implement a modified version of the Quicksort algorithm, where the sequence of elements is always split into three subsequences by simultaneously using the first two elements as pivots.

(b) (4 points) Determine and prove the best-case and worst-case running time for the modified Quicksort from subpoint (a).

(c) (2 points) Implement a modified version of the Randomized Quicksort algorithm, where the sequence of elements is always split into three subsequences by simultaneously using two random elements as pivots.

Answers

A. For A see quicksort_modified.cpp

B. The best case for this modified version would be one where the array is split into three equal parts which means that the recurrence formula is:

   $$T(n) = 3T(n/3) + O(n)$$

   The worst case would be when the pivots are the smallest and largest element, which means that the recurrence formula for this case is:

   $$T(n) = T(n-2) + 0(n)$$

   See the attached TimeComplexity.pdf for the time complexity calculations

C. For A see quicksort_random.cpp