CS 61B Spring 2024

More Sorting

Discussion 12: April 15, 2024

1 Quicksort

(a) Sort the following unordered list using Quicksort. Assume that we always choose first element as the pivot and that we use the 3-way merge partitioning process described in lecture. Show the steps taken at each partitioning step.

```
18, 7, 22, 34, 99, 18, 11, 4
```

(b) What is the best and worst case running time of Quicksort with Hoare Partitioning on N elements? Given the two lists [4, 4, 4, 4, 4] and [1, 2, 3, 4, 5], assuming we pick the first element as the pivot every time, which list would result in better runtime?

(c) What are two techniques that can be used to reduce the probability of Quicksort taking the worst case running time?

2 Radix Sorts

(a) Sort the following list using LSD Radix Sort with counting sort. Show the steps taken after each round of counting sort. The first row is the original list and the last two rounds are already filled for you.

	30395	30326	43092	30315
1				
2				
3				
4	30315	30326	30395	43092
5	30315	30326	30395	43092

(b) Sort the following list using MSD Radix Sort with counting sort. Show the steps taken after each round of counting sort. The first row is the original list and the first round is already filled for you.

	21295	22316	30753	21248	30751
1	21295	22316	21248	| 30753	30751
2					
3					
4					
5					

(c) Give the best case runtime, worst case runtime, and whether or not the sort is stable for both LSD and MSD radix sort. Assume we have N elements, a radix R, and a maximum number of digits in an element W.

(d) We saw in part (c) that radix sort has good runtime with respect to the number of elements in the list. Given this fact, can we say that radix sort is the best sort to use?

3 Sort Identification

Match the sorting algorithms to the sequences, each of which represents several intermediate steps in the sorting of an array of integers. Assume that for quicksort, the pivot is always the first item in the sublist being sorted. Note that these steps are not necessarily the first few intermediate steps and there may be steps which are skipped.

Algorithms: *Quicksort, Merge Sort, Heapsort, MSD Radix Sort, Insertion Sort*

(a) `12, 7, 8, 4, 10, 2, 5, 34, 14`
`7, 8, 4, 10, 2, 5, 12, 34, 14`
`4, 2, 5, 7, 8, 10, 12, 14, 34`

(b) `23, 45, 12, 4, 65, 34, 20, 43`
`4, 12, 23, 45, 65, 34, 20, 43`

(c) `12, 32, 14, 11, 17, 38, 23, 34`
`12, 14, 11, 17, 23, 32, 38, 34`

(d) `45, 23, 5, 65, 34, 3, 76, 25`
`23, 45, 5, 65, 3, 34, 25, 76`
`5, 23, 45, 65, 3, 25, 34, 76`

(e) `23, 44, 12, 11, 54, 33, 1, 41`
`54, 44, 33, 41, 23, 12, 1, 11`
`44, 41, 33, 11, 23, 12, 1, 54`

4 Conceptual Comparison Sorts *Extra*

Answer the following questions regarding various sorting algorithms that we've discussed in class. If the question is T/F and the statement is true, provide an explanation. If the statement is false, provide a counterexample.

(a) Give a 5 integer array that elicits the worst case runtime for insertion sort.

(b) Give some reasons as to why someone would use mergesort over quicksort.

(c) You are given the following options:

(A) Quicksort (in-place using Hoare partitioning and choose the leftmost item as the pivot)

(B) Mergesort

(C) Selection Sort

(D) Insertion Sort

(E) Heapsort

(F) (None of the above)

For each of the statements below, list all letters that apply. Each option may be used multiple times or not at all. Note that all answers refer to the entire sorting process, not a single step of the sorting process, and assume that N indicates the number of elements being sorted.

_____ Bounded by $\Omega(N \log N)$ lower bound.

_____ Worst case runtime that is asymptotically better than Quicksort's worst case runtime.

_____ In the worst case, performs $\Theta(N)$ pairwise swaps of elements.

_____ Never compares the same two elements twice.

_____ Runs in best case $\Theta(\log N)$ time for certain inputs.