QuickSort Recap

This is another example of a pure “divide and conquer” algorithm.

**Step 1 (divide)**
Select a “pivot” value and logically partition the list into two sub-lists:
- L1: values less than the pivot
- L2: values greater than the pivot

Your list is now: \([L1, pivot, L2]\)

**Step 2 (conquer)**
Sort L1 and L2

SORTED!
QuickSort Pseudocode

Algorithm
Let’s assume that our list L is held in an array and that we want to use as little extra space as possible.

QuickSort(array L, int first, int last) {
    if (first < last) {
        pivotpos = Partition(L, first, last)
        QuickSort(L, first, pivotpos-1)
        QuickSort(L, pivotpos+1, last);
    }
}

NOTE: We would still need to write the Partition algorithm. The easiest thing to code would probably be to pick the last value in the list as the pivot and then partition based on that.

Trace some examples if needed/useful…

5, 7, 6, 1, 3, 2, 4
1, 2, 3, 4, 5
Partition’s runtime…

There are many ways to implement the partition algorithm, but in terms of the number of data comparisons, it should be accomplished using n-1.

QuickSort’s runtime…

Start with \( T(0) = T(1) = 0 \)

For the recurrence relation we want to consider three cases:

- With the worst case split.
  \[ T(n) = (n-1) + T(0) + T(n-1) \]

- With the best case split.
  \[ T(n) = (n-1) + T(n/2-1) + T(n/2) \]

- With the average/expected split….
Average Case Analysis

We return to the idea of expected values…
Let’s assume that every “division situation” around the pivot is equally likely.

If we let $i$ represent the position where $L2$ starts, then we could represent the expected runtime as being:

$$T(n) = (n-1) + \frac{\sum_{i=1}^{n}[T(i-1)+T(n-i)]}{n}$$

What about that worst case?

Recall that regardless of the “average” case, that if we expect mostly-sorted inputs, then the runtime will be bad.

How could we alter our approach to try to address (ie: decrease the likelihood of) the issue of sorted lists leading to $n^2$ runtime with the pivot/partitioning algorithm that I originally presented?