Readings Chapter 7

Self test 7.1-1, 7.2-2

Lecture 11

The following algorithm sorts an input sequence S in non-decreasing order.

```
QuickSort(S):
  if |S| ≤ 1:
    return S
  else:
    select pivot p ∈ S
    parition elements of S into
      L = elements of S < p
      E = elements of S = p
      G = elements of S > p
    return[QuickSort(L), E, QuickSort(G)]
```

For the purposes of determinism, select the first item in S as pivot.

To prove worst/best case:

- a worst/best case input
- the time complexity for this input
- an argument that this input must be a worst or best case (no other case can take longer/shorter)

Worst Case

To simplifiy analysis, count only comparisons between elements of S. Where are these comparisons performed? → During partition step.

Upper bound:

- every element in S is pivot at most once
- every pair of elements are compared at most once (at most all other elements of S are compared to pivot)
- there are $\binom{n}{2}$ pairs of elements if $|S| = n$, $T(n) \leq \binom{n}{2}$ is in $O(n^2)$

Lower bound:
Want to find S of size n for which `QuickSort(S)` does at least cn^2 comparisons.
Let $C(n) = \#$ of comparisons performed on $[n, n-1, n-2, \ldots, 2, 1]$.
n will be pivot $\rightarrow E = [n]$
$n - 1$ comparisons $\rightarrow L = [\,], G = [n-1, n-2, \ldots, 2, 1]$

$$\begin{aligned} C(n) &= n - 1 + C(n-1), C(1) = 0 \\ &= (n-1) + (n-2) + (n-3) + \cdots + 1 \\ &= \sum_{i=o}^{n-1} i = \frac{n(n-1)}{2} \in O(n^2) \end{aligned}$$

By definition, $T(n) \geq C(n) = \binom{n}{2}$. Hence $T(n) \in \Omega(n^2)$. Then $T(n) \in \Theta(n^2)$.