CS110
Data Structures and Algorithms

Lecture 07

Sorting

“Orderliness is beauty”

Application of Sorting : Duplicate Finding

- Recap: Given an array / list of numbers, it take $O(n^2)$ comparisons to find duplicates by the naive approach of comparing every element with the other

- If the same is array / list is known to be sorted, then the solution to duplicate finding takes only $O(n)$ comparisons

- Given that an optimized sorting algorithm itself takes $O(nlogn)$ time [to be studied shortly], the duplicate finding problem is of complexity $O(nlogn)$

So Sorting is after all useful: How to Sort?

- There are 10 players wearing numbered jerkins 1 through 10 standing in a line. Note player A's jerkin number is denoted as #A. They were asked to rearrange their lineup in the increasing order of their jerkin number following the rules
- Step 1: The first person in the lineup is handed a baton. Whoever has the baton does the following.
- Step 2: Say, Player A in position j has the baton and the right neighbor at j+1. Is B
- Step 3: If #A > #B then A and B exchange their positions and A retains the baton. Else A hands the baton over to B. Repeat Steps 2,3 till baton reaches last in the lineup
- Step 4: If at least one exchange happened then repeat from Step 1

Modeling it as a computational problem

- Sorting a list of numbers is the computational equivalent of the earlier real world problem
- Players with numbered jerkins lined up, is modeled as a list of numbers
- Players exchanging places is swapping of two adjacent elements in the list
- The baton moving from the first to the last in the lineup is setting up Iteration
- Given a list of n numbers, justify that the it takes not more than n-1 iterations to arrive at the sorted order

Describing Complexity

- An Algorithm of complexity g(n) has a constant cost operation X executed g(n) number of times.
- We have seen before that the operation may be one of comparison, arithmetic operation, indexing, function call etc.
- The naive algorithm of finding duplicates in an ordered list has $O(n^2)$ comparisons.
- To improve the solution number of described operation (comparison in the above case) should be reduced.
- Using the ordering property total number of comparisons dropped to O(n) comparisons justifying the new improved complexity measure.