Lecture 10, Oct 13, 2023

Randomized Quicksort

- We've seen algorithms that rely on the input being random to achieve a good average runtime; what if we make random choices inside the algorithm ourselves, so that even if the input is not random, we still get good performance?
- Suppose we want to sort a set of S distinct keys in increasing order; we can use recursive quick sort (RQS):
  1. If S is empty or has only one element, then return it
  2. Select a "pivot" key p uniformly at random among S (i.e. each key is equally likely to be selected as pivot)
  3. Compare the pivot with every element of S; split into two subsets $S_<$ which are keys less than p, and $S_>$ which are keys greater than p
  4. Recursively sort $S_<$ and $S_>$; output $S_<, p, S_>$ in order
- Note:
  - Two keys are compared only if one of them is selected as a pivot
  - Two keys are compared at most once, since a pivot cannot compare with keys after partitioning
  - If two keys are split by a pivot, they will never be compared
- Consider fixing some input S with n keys; let C be the number of pairwise key comparisons done by RQS
  - In the worst case, we choose the biggest or smallest element of the set as the pivot each time, so each recursive call reduces n by 1
    - $C = (n-1) + (n-2) + \cdots + 2 + 1 = \Theta(n^2)$
  - What is the expected value of C? i.e. in the average case, over all the possible pivot selections, how many comparisons do we get?
  - Let $z_1 < z_2 < \cdots < z_i < \cdots < z_j < \cdots < z_n$ be the keys of S in ascending order
    - Let $c_{ij} = 1$ if RQS compares z_i, z_j, or 0 otherwise
    - We call this an *indicator random variable*
    - We have $C = \displaystyle\sum_{1 \le i < j \le n} c_{ij}$
  - So what is $\mathbb{E}[c_{ij}]$?
    - $\mathbb{E}[c_{ij}] = 1 \cdot P(c_{ij} = 1) + 0 \cdot P(c_{ij} = 0) = 0$
    - We will later prove $c_{ij} = \dfrac{2}{j - i + 1}$
    - Substituting this back in and expanding the sum, we get that $H_n = 1 + \dfrac{1}{2} + \dfrac{1}{3} + \cdots \in O(\log n)$
- Consider the special case $i = 1, j = n$, then for $i = 1, j = n$, we have $\dfrac{2}{n}$ of them being compared
  - These two keys will only be compared if the first or last element s chosen as pivot
  - For any i and $j = i + 1$, the probability of comparison is guaranteed, since they are right next to each other
    - The only way these elements are separated is if one of them becomes a pivot
  - Consider the set of keys $Z_{ij} = \{ z_i, \dots, z_j \}$
    - For this set to stay on the same side, the pivot must be its range
    - This set has size $j - i + 1$
    - Initially Z_{ij} is entirely contained in S
    - RQS keeps splitting S until it gets to subsets of size one or zero; as long as the pivot is not in the set, then the set will stay together and be untouched
    - Consider the first time RQS selects a pivot in Z_{ij}
      - If z_i or z_j is selected as the pivot, they will be compared once and never again
      - If $z_i < p < z_j$ then z_i or z_j will never be compared
    - Therefore the probability of z_i, z_j being compared is the probability of selecting z_i or z_j as the pivot, given $p \in Z_{ij}$

  - The size of Z_{ij} is $\dfrac{1}{j-i+1}$ and we have two possibilities
  - So given any two keys, the probability of comparison is $\dfrac{2}{j-i+1}$
- Hence, C is $O(n \log n)$