Zijian Gao
26319560
2/29/2016
CS311 HW2

9.3-1.)

With groups of 7, the number of elements greater than the median is at least $4(\lceil \frac{1}{2}\lceil \frac{n}{7} \rceil \rceil - 2)$, similar to select with 5 groups, half of the groups have medians that are greater than or equal to the median-of-medians x such that they may contribute at least 4 (including their median) elements to x. The same unfilled groups (one group that may not be fully filled if n mod 7 is not 0, and one that includes x and so will not contribute 4 elements). Following the pattern for step 5, SELECT will be recursively called on at most $\frac{5n}{7} + 8$ elements. Following the inductive proof done on SELECT, we have the recurrence $T(n) \le T(\lceil \frac{n}{7} \rceil) + T(\frac{5n}{7} + 8) + O(n)$ for all n greater than some constant value. Choose a such that $O(n)$ is bounded above by an for all n. Assume $T(n) \le cn$ for some suitably large constant c and all n less than some constant bound. So, substituting into the recurrence, $T(n) \le \frac{cn}{7} + \frac{5cn}{7} + 8c + an$. By associativity, $T(n) \le cn + (-\frac{cn}{7} + 8c + an)$ and so $T(n)$ is at most cn if $-\frac{cn}{7} + 8c + an \le 0$ and the algorithm still works in linear time if divided into groups of 7.

For groups of 3, an nlog(n) lower bound can be obtained by running analysis similar to before. In this case, at least half of the groups of 3 will be greater than the median of medians x, in which at least 2 elements will be greater than x. The group including x and the group previously subtracted for having fewer than 3 elements do not need to be removed this time since these groups contribute less than 2 elements that are greater than x, and so fall into the base case for SELECT. It follows that the number of elements that are greater than x is bounded above by $2(\lfloor \frac{1}{2}\lfloor \frac{n}{3} \rfloor \rfloor)$ so SELECT will be recursively called on at least $\frac{2n}{3}$ elements. $T(n) \ge T(\lceil \frac{n}{3} \rceil) + T(\frac{2n}{3}) + \Omega(n)$. Taking $T(n) \ge cn\log(n)$ and choosing a such that $\Omega(n)$ is bounded below by a for all n,

$T(n) \ge T(\lceil \frac{n}{3} \rceil) + T(\frac{2n}{3}) + \Omega(n) \ge cn\log(\frac{n}{3}) + c\frac{2n}{3}\log(\frac{2n}{3}) + an$.

$T(n) \ge cn\log(n) - cn\log(3) + \frac{2cn}{3}\log(2n) - \frac{2cn}{3}\log(3) + an = cn\log(n) - cn\log(3) + \frac{2cn}{3}\log(2) + \frac{2cn}{3}\log(n) - \frac{2cn}{3}\log(3) + an$

By expanding logarithms. It is trivially true that $\frac{2cn}{3} \le \frac{2cn}{3}\log(2) + \frac{2cn}{3}\log(n) - \frac{2cn}{3}\log(3)$ for all n as linear growth is bounded above by n log(n). Substituting,

$T(n) \ge cn\log(n) - cn\log(3) + \frac{2cn}{3} + an$ and so $T(n) \ge cn\log(n)$ if $-cn\log(3) + \frac{2cn}{3} + an \ge 0$ and so SELECT with groups of 3 does not work in linear time.

9.3-3.)

Quicksort can be made to run in O(n log n) time by modifying the means by which partitions are made using the SELECT algorithm to always select the $\lfloor \frac{n}{2} \rfloor$ smallest median pivot x such that the best case partitioning is always achieved where the partitioning produces two subproblems, each of size no more than n/2. The runtime for this is $T(n) = 2T(\frac{n}{2}) + \theta(n\log n)$, which replaces the runtime $\theta(n)$ for partitioning with n log n for using SELECT to partition. This trivially simplifies to

$T(n) = \theta(n\log n) + \theta(n\log n) = \theta(n\log n)$