Illinois Institute of Technology
Department of Computer Science

Solutions to Homework Assignment 5

CS 430 Introduction to Algorithms
Spring Semester, 2016

Solution:

1. 19.3-1 on page 522

   A root x is marked if (1) one of his child is removed for any reason (a child being promoted as a new root also counts) or (2) x used to be a marked child of the $H.min$, and $H.min$ is extracted by FIB-HEAP-EXTRACT-MIN. If x were a marked non-root and lost one child, it would have promoted as a new root and unmarked at then, therefore whether x has been a marked non-root is irrelevant to analyzing how it came to be a marked root.

2. 19-3 on page 529

   (a) i. $k > x.key$: It is possible that k is larger than keys of some children of x. Therefore, update the key $x.key \leftarrow k$ and then push x down until the min heap property is preserved. The worst actual cost is $O(\log n)$, and the potential does not change, therefore the amortized cost is $O(\log n)$.

   ii. $k = x.key$: Does nothing, therefore the potential does not change, and the amortized cost is equal to the actual cost (of the comparison) which is $O(1)$.

   iii. $k < x.key$: Simply call FIB-HEAP-DECREASE-KEY(H, x, k), whose amortized cost is $O(1)$.

   (b) The amortized cost of deleting a given node is $O(\log n)$, but that does not mean the answer to this problem is $O(q \log n)$ because we are allowed to delete any q nodes we choose. Intuitively, we try deleting q leaves. In the worst case, every deleted leaf will incur cascading cut until the root (in each individual tree), which implies the actual cost of FIB-HEAP-PRUNE(H, r) is $c = q \log n$. Now we use the potential function (as in CLRS page 509) $\Phi(H) = t(H) + 2m(H)$ to see what is the potential change. Note that I ignored all ±1's during the calculation for better understanding, which does not affect the final amortized cost.

   $$t(H_{\text{after}}) - t(H_{\text{before}}) = q \log n$$

   since in the worst case $q \log n$ new trees are created due to the promotion during the cascading cut, and H_{after} has $q \log n$ more trees.

   $$m(H_{\text{after}}) - m(H_{\text{before}}) = -q \log n \Rightarrow 2m(H_{\text{after}}) - 2m(H_{\text{before}}) = -2q \log n$$

   because in the worst case $q \log n$ marked nodes are promoted as new roots, which means there are $q \log n$ less marked nodes in H_{after}. Then,

   $$\hat{c} = c + \Phi(H_{\text{after}}) - \Phi(H_{\text{before}}) = q \log n + q \log n - 2q \log n = O(1)$$

   Therefore, removing leaf nodes has an amortized cost $O(1)$.