Homework HW8: 9-29

0110761 邱盟竣

A binary heap is a heap data structure created using a binary tree. The optimal method starts by arbitrarily putting the elements on a binary tree, respecting the shape property (the tree could be represented by an array, see below). Then starting from the lowest level and moving upwards, shift the root of each subtree downward as in the deletion algorithm until the heap property is restored. More specifically if all the subtrees starting at some height h (measured from the bottom) have already been "heapified", the trees at height $h+1$ can be heapified by sending their root down along the path of maximum valued children when building a max-heap, or minimum valued children when building a min-heap. This process takes $O(h)$ operations (swaps) per node. In this method most of the heapification takes place in the lower levels. Since the height of the heap is $\lfloor \log(n) \rfloor$, the number of nodes at height h is $\leq \left\lceil 2^{(\log n - h)-1} \right\rceil = \left\lceil \frac{2^{\log n}}{2^{h+1}} \right\rceil = \left\lceil \frac{n}{2^{h+1}} \right\rceil$.

Therefore, the cost of heapifying all subtrees is:

$$\begin{aligned}\sum_{h=0}^{\lceil \log n \rceil} \frac{n}{2^{h+1}} O(h) &= O\left(n \sum_{h=0}^{\lceil \log n \rceil} \frac{h}{2^{h+1}} \right) \\ &\leq O\left(n \sum_{h=0}^{\infty} \frac{h}{2^h} \right) \\ &= O(n)\end{aligned}$$ [1]

[1] http://www.wikiwand.com/en/Binary_heap