ALGORITHMIC DESIGN - HOMEWORK - BINARY HEAPS

Keerthana C J

September 2020

1 Binary Heaps Implementation

The array implementation of Binary Heaps has been done with the functions HEAP MIN, REMOVE MIN, HEAPIFY, BUILD HEAP, DECREASE KEY, and INSERT VALUE. The following table compares extracting the minimum value from a heap to that of an array.

Size of instance	Heaps	Arrays
0	0.000009	0.000002
1820	0.079779	0.617346
3640	0.120430	2.815553
5461	0.216084	6.410724
7281	0.273035	10.976252
9102	0.346384	21.220686
10922	0.391069	24.661298
12743	0.565134	33.472749
14563	0.600841	43.524406
16384	0.700654	57.064713

Table 1: Time needed to extract the minimum value from a heap vs from an array

2 Theoretical exercises

1. Show that, with the array representation, the leaves of a binary heap containing n nodes are indexed by $\lfloor \frac{n}{2} \rfloor + 1$, $\lfloor \frac{n}{2} \rfloor + 2$, ..., n.
   Answer: Let us consider the left child of the node with index $\lfloor \frac{n}{2} \rfloor + 1$.

$$LEFT\_CHILD(\lfloor \frac{n}{2} \rfloor + 1) = 2(\lfloor \frac{n}{2} \rfloor + 1) > 2(\frac{n}{2} - 1 + 1) > n$$

Since the total number of nodes is n and the left child of the node with index $\lfloor \frac{n}{2} \rfloor + 1$ is greater than n, it should be a leaf.
Coming to the node with index $\lfloor \frac{n}{2} \rfloor$, it cannot be a leaf as it will have a left child with index n if n is even and if n is odd, it contains a left child with index $n - 1$ and a right child with index n.

2. Show that the worst-case running time of HEAPIFY on a binary heap of size n is $\Omega(log_2 n)$.
   Answer: Consider a heap resulting from an array $A[n] = 1$ and $A[i] = 2$ for $1 \leq i < n$. Since 1 is the smallest element, it must be swapped through each level until it becomes the root. The height of heap is given by $\lfloor log_2 n \rfloor$. Hence the number of swaps performed is in the order of $\Omega(log_2 n)$.

3. Show that there are at most $\lceil \frac{n}{2^{h+1}} \rceil$ nodes of height h in any n element heap.
   Answer: From question 1, we can understand that the number of leaves in any heap of size n is $\lceil \frac{n}{2} \rceil$. Let n_h be the number of nodes at height h and n_{h-1} at height $h - 1$.
   Hence, $n_h = \lceil \frac{n_{h-1}}{2} \rceil \leq \lceil \frac{1}{2} \frac{n}{2^{(h-1)+1}} \rceil = \lceil \frac{n}{2^{h+1}} \rceil$