Binary Heap

Binary Heap is a complete binary tree with heap-order property. A complete binary tree is a binary tree that is completely filled with the possible exception of bottom level which is filled from left to right. Heap-order property maintains that each node be smaller than (or equal to) its children.

Since binary heap is a complete binary tree, it can implemented using a simple array. For any element in i position, its left child is in $2i$ position, its rght child is in $(2i+1)$ position and its parent is in $\lfloor i/2 \rfloor$ position.

d-Heap is a heap with each node having at most d children. A similar strategy can be used for its array based implementation. For a 3-heap A, its root is at $A[1]$ and the children of $A[i]$ is in positions $A[3i-1]$, $A[3i]$ and $A[3i+1]$. For increasing d value, the heap becomes shallower.

Height of a binary heap is $O(logN)$

Consider a binary heap H of height h containing N elements.

Minimum number of elements: 2^h
Maximum number of elements: $2^{h+1}-1$

Thus bounds on height of the binary heap H is

$$log(N+1)-1 \leq h \leq log(N)$$

Height of a binary heap = $O(logN)$

Build heap from an array in $O(N)$

The idea is to start at index $N/2$ and percolate down for each indices above it. Thus,

$$i = N/2, (N/2)-1, ..., 1$$

Each element below $N/2$ index is a child of a parent which will be percolated down.

Assume a perfect binary heap H (complete at the bottom level as well) of height h containing $N = 2^{h+1}-1$ elements. Each indices undergoes 2 $comparisons$, 1 $swapping$ followed by a down $percolation$. Maximum depth to which each element percolates to is exactly the height of that element.

Max Operations for each index = (2 $comparison$ + $swapping$) $\times$ height of index
Total Max Operations = (2 $comparison$ + $swapping$) $\times$ Sum of heights of all indices

Total Complexity = $O(Sum\ of\ heights\ of\ all\ indices)$

Sum of heights of level j = Number of items in level $j \times$ height of j

$$S_j = 2^j \times (h - j)$$

$S_0 = 2^0 \times (h - 0) = h$
$S_1 = 2^1 \times (h - 1) = 2(h - 1)$
$S_2 = 2^2 \times (h - 2) = 4(h - 2)$
...
Thus,

$$S = S_0 + S_1 + S_2 + ... + S_{h-1}$$

$$S = h + 2(h-1) + 4(h-2) + 8(h-3) + ... + 2^{h-1}(1)$$

$$2S = 2(h) + 4(h-1) + 8(h-2) + 16(h-3) + ... + 2^h(1)$$

Subtracting above two equations we get,

$$S = -h(2 + 4 + 8 + ... + 2^h) = -h - 1 + (1 + 2 + 4 + ... + 2^h)$$

Using geometric series sum, the total sum is given as,

$$S = 2^{h+1} - 1 - (h+1) = N - (h+1)$$

Similarly for binary heap with 2^h elements, the sum is

$$S = 2^h - 1$$

Thus for a given height h of binary heap, bounds on S is given by:

$$2^h - 1 \leq S \leq 2^{h+1} - 1 - (h+1)$$

The upper bound on S validates that the complexity of building heap from an array of items is $O(N)$.

Another representation of the sum is given as,

$$S = N - b(N)$$

where, $b(N)$ is 1's in binary representation of N