http://www.cs.unh.edu/~ruml/cs758

Heaps

Problems

1. Finding the min

Problems

1. Finding the min
2. Finding the min with insertions

Problems

1. Finding the min
2. Finding the min with insertions
3. Finding the min with insertions and deletions

Heaps

Invariant: parent comes before (or equal to) children

Implementation

parent $i = \frac{(child\ \ i)-1}{2}$

left child of $i = 2i + 1$

right child of $i = 2i + 2$

automatic balance!

Insertion

1. insert at end

Insertion

1. insert at end
2. re-establish invariant by

Insertion

1. insert at end
2. re-establish invariant by pulling up if necessary

Pull Up

assume heap except possibly between i and parent:
$A[i]$ might be too small

so consider pulling $A[i]$ up

pullup(i)
1. if $A[i]$ comes before $A[parent]$
2. exchange $A[i]$ with $A[parent]$
3. pullup(*parent*)

invariant: initialization, maintenance, termination

Extract Min

1. remove first elt
2. copy last into first

Extract Min

1. remove first elt
2. copy last into first
3. re-establish invariant by

Extract Min

1. remove first elt
2. copy last into first
3. re-establish invariant by pushing down if necessary

Extract Min

1. remove first elt
2. copy last into first
3. re-establish invariant by pushing down if necessary

heapsort

Push Down

assume heap except possibly between i and parent:
$A[i]$ might be too large

so consider pushing $A[i]$ down

pushdown(i)
1. *smallesti* $\leftarrow$ index of smallest among i and children
2. if *smallesti* $\neq i$ then
3. exchange $A[i]$ with $A[smallesti\,]$
4. pushdown(*smallesti*)

invariant: initialization, maintenance, termination

Correctness

What's the space complexity?

What's the time complexity?

Break

- asst 1
- asst 2
- asst 3

More Heaps

Creation

Given array, how to form heap?

Creation

Given array, how to form heap?

Can we do better than $\Theta(\frac{n}{2}\lg\frac{n}{2}) = \Theta(n\lg n - n) = \Theta(n\lg n)$?

Creation

Given array, how to form heap?

Can we do better than $\Theta(\frac{n}{2}\lg\frac{n}{2}) = \Theta(n\lg n - n) = \Theta(n\lg n)$?

bottom up:

1. for i from $\frac{length}{2} - 1$ to 0
2.   pushdown(i)

how long does this take?

Creation Time

height of a node is (longest) distance to a leaf

$$\sum_{h=0}^{\lg n} (O(h) \times \#\text{-}\textit{nodes-with-height-h})$$

Creation Time

height of a node is (longest) distance to a leaf

$$\sum_{h=0}^{\lg n} (O(h) \times \textit{\#-nodes-with-height-h})$$

We will see $\frac{n}{2^{h+1}}$ nodes with height h.

$$\sum_{h=0}^{\lg n} O(h) \frac{n}{2^{h+1}} \quad = \quad O(n \sum_{h=0}^{\lg n} \frac{h}{2^{h+1}})$$

Creation Time

height of a node is (longest) distance to a leaf

$$\sum_{h=0}^{\lg n} (O(h) \times \textit{\#-nodes-with-height-h})$$

We will see $\frac{n}{2^{h+1}}$ nodes with height h.

$$\sum_{h=0}^{\lg n} O(h) \frac{n}{2^{h+1}} \quad = \quad O(n \sum_{h=0}^{\lg n} \frac{h}{2^{h+1}})$$

$\sum_{h=0}^{\infty} \frac{h}{2^h} = 2$

Creation Time

height of a node is (longest) distance to a leaf

$$\sum_{h=0}^{\lg n} (O(h) \times \textit{\#-nodes-with-height-h})$$

We will see $\frac{n}{2^{h+1}}$ nodes with height h.

$$\sum_{h=0}^{\lg n} O(h)\frac{n}{2^{h+1}} \quad = \quad O(n\sum_{h=0}^{\lg n}\frac{h}{2^{h+1}})$$

$\sum_{h=0}^{\infty} \frac{h}{2^h} = 2$

$$\begin{aligned} O(n\sum_{h=0}^{\lg n}\frac{h}{2^{h+1}}) \quad &= \quad O(n\sum_{h=0}^{\infty}\frac{h}{2^h}) \\ &= \quad O(n) \end{aligned}$$

resize by doubling! how expensive?

resize by doubling! how expensive?

'amortized' analysis: the 'accounting method'

1. start half full, with zero credit
2. each insertion costs 3:
   (a) insert self now
   (b) eventually move self when full
   (c) eventually move an existing elt when full
3. when full, have credit for each item
4. now half full, with zero credit

Amortization 2

'amortized' analysis: the 'aggregate method'

Let $c_i = i$ if $i - 1$ is a power of 2, 1 otherwise.

$$\sum_{i=1}^{n} c_i$$

Amortization 2

'amortized' analysis: the 'aggregate method'

Let $c_i = i$ if $i - 1$ is a power of 2, 1 otherwise.

$$\begin{aligned}
\sum_{i=1}^{n} c_i &\leq n + \sum_{j=0}^{\lg n} 2^j \\
&< n + 2n \\
&< 3n
\end{aligned}$$

Problems

1. Finding the min
2. Finding the min with insertions
3. Finding the min with insertions and deletions
4. Finding the kth largest

EOLQs

- What's still confusing?
- What question didn't you get to ask today?
- What would you like to hear more about?

Please write down your most pressing question about algorithms and put it in the box on your way out.
Thanks!