

CSCI 429 Assign 2 Sol 1017

1. Binary Rep Heap, for ADT Priority Queue

1.1. (4) Given n , the shape of the Bin Rep Heap reflects the binary representation of n .

in that $X[i] = \begin{cases} \text{an array of size } 2^i \text{ if} \\ \text{the } i^{\text{th}} \text{ bit of } n \text{ is } 1 \\ \text{an empty array otherwise} \end{cases}$

1.2. (4) - Max time for insert:

If all lists are full, an insertion will cost $\Theta\left(\sum_{i=0}^{\lg n} 2^i\right)$, to merge lists of size $0, 1, 2, \dots, \lg n$.

ie running time in worst case is $\Theta(n)$

- max time for findMin is $\Theta(\lg n)$ as need to look at one element in each of $\lg n$ lists to find the smallest.

1.3 (8) Amortized running time of Insert-only

Claim: m inserts, starting with empty heap, take $\Theta(\lg n)$ amortized time per operation.

Proof: Using the Accounting method, upon insertion we give the inserted element $\lfloor \lg n \rfloor + 2$ credits to pay for work, where

1 credit can pay for — creation of a list of size 1
or — an element comparison

Each insert therefore costs $\lfloor \lg n \rfloor + 2$ credits, and each credit pays for a constant amount of work.

◦ it only remains to show that the credits in the system are sufficient to pay for all the work done.


Insert (x)

— creating a new list with just x in it costs 1 credit

The remaining credits can be called $x_0, x_1, \dots, x_{\lfloor \lg n \rfloor}$

— if $A[0]$ is empty, $A[0]$ will now point to x 's list, and no merges need be paid for — x 's credits $x_0 \dots x_{\lfloor \lg n \rfloor}$ are intact.

- As x is merged into larger lists, it will contribute its credit x_i to the merge of the 2^i -sized list x is in with another 2^i -sized list. Since $2 \cdot 2^i$ elements contribute to the merge, all the comparisons in this merge are paid for.

- As a result, any element y in the list of size 2^{i+1} has its credits $y_{i+1}, y_{i+2}, \dots, y_{\lfloor \lg n \rfloor + 1}$ unspent, and can pay for future merges. 

1.4 (4) This should read "findMin", not "find". Those who answer for "find" will not have marks deducted.

findmin runs in $\Theta(\lg n)$ time, which is also $\lg n$ amortized time, and does not change the data structure.

Insert runs in $\Theta(\lg n)$ amortized time

Consequently, interleaving these operations

will still be in $\Theta(\lg n)$ amortized time.

2. (4) Lemma 2.3 of CLRS

$\text{rank}(x) = 0$ right after $\text{makeSet}(x)$ is called

$\text{rank}(x)$ increases by 1 if x is made parent of y when y has same rank as x

Nothing else changes the rank of x .

Claim: $\text{size}(x) \geq 2^{\text{rank}(x)}$

Proof: By induction on r .

Basis: if $r=0$, then x has no children
and $\text{size}(x) = 1 = 2^0 = 2^{\text{rank}(x)}$

Let r be a rank > 0 .

Ind Hyp: Assume that it is always the case in union find forests using union-by-rank that $\forall y$ of rank $r-1$, $\text{size}(y) \geq 2^{r-1}$

Now consider an x that has just achieved rank r by a Link operation; $\text{Link}(x, y)$

Then before the link, y and x both had

rank $r-1$.

Then before the Link, $\text{size}(y) \geq 2^{r-1}$ and
 $\text{size}(x) \geq 2^{r-1}$.

Then after the Link, y is a child of x

and $\text{size}(x) \geq 2^{r-1} \cdot 2 = 2^r$. 