CS5800: Algorithms — Virgil Pavlu

Homework 9

Name: Mumuksha Pant
Collaborators:

Instructions:

- Make sure to put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- Please review the grading policy outlined in the course information page.
- You must also write down with whom you worked on the assignment. If this changes from problem to problem, then you should write down this information separately with each problem.
- Problem numbers (like Exercise 3.1-1) are corresponding to CLRS 3^{rd} edition. While the 2^{nd} edition has similar problems with similar numbers, the actual exercises and their solutions are different, so make sure you are using the 3^{rd} edition.

1. (25 points) Exercise 17.3-3. (Hint: a reasonable potential function to use is $\phi(D_i) = kn_i \cdot \ln n_i$ where n_i is the number of elements in the binary heap, and k is a big enough constant. You can use this function and just show the change in potential for each of the two operations.)

Solution:

Insert -

Using Potential Function Method :

$\hat{c} = c + \phi(D_i) + \phi(D_{i-1})$

$$\phi(D_i) = n_i * \log n_i \tag{1}$$
$$\phi(D_{i-1}) = n_{i+1} * \log n_{i+1} \tag{2}$$

$$\Delta\phi = 1 \tag{3}$$

equation (3) : since only one element is being inserted, size of heap is increased by 1

Putting the value of (3) in Potential Function AND using c= log.n (which is the true cost of insert in a binary min- heap)

$\hat{c} = \log n + 1$

$\hat{c} = O(\log n)$

Extract MIN

Using Potential Function Method :

$\hat{c} = c + \phi(D_i) + \phi(D_{i-1})$

Extract-MIN reduces the size of the heap by 1. True Cost is the change in potential ($\Delta\phi$) of the same order, so constant amount of work is required , ie, O(1) .

$\phi_i = k * [(n-1)\log(n-1)]$
$\phi_{i-1} = k * [n.\log n]$

$\Delta\phi = \phi_i - \phi_{i-1}$

$\Delta\phi = k[(n-1)\log(n-1) - n\log n]$

$\Delta\phi = k[\log(n-1)^{n-1} - \log n^n]$

$\Delta\phi = k[\log \frac{n-1^{n-1}}{n^n}]$

We know, $(1 - 1/n)^n = 1/e$

$\Delta\phi = k[\log \frac{n-1}{n}]$

$\Delta\phi = k[\log 1 - \frac{1}{n}] \leq -log2$

$\hat{c} = truecost + k.(-\log 2)$

$\hat{c} = logn - k.log2$

$\hat{c} = O(1)$

---

2. (25 points) Exercise 17.3-6.

Solution:
Accounting method can be used here for analysis of Enque and Deque operarion. In accounting method, when an operation's amortized cost exceeds its actual cost, we assign the difference to specific objects in the data structure as credit. **Credit can be used to pay for later operations** when amortized cost is less than actual cost.

```
class myQueue
{
Stack S1,
Stack S2,


void Push(int data )
 { S1. push (data ) }

int pop (x)
{
   while ( ! S1. empty () )
     S2. push ( S1. pop () )

int ans = S2. pop ()

while ( ! S2.empty () )
    S1.push( S2.pop () )

return ans
}

}
```

Enqueue operation : 3 coins
Deque operation : 0 coins

We assign cost (ammortized) 3 coins to each enqueue operation . we enqueue by pushing data in Stack 1.

For deque operation,
1. We pop every element from S1 and push it to S2. (for each item we are pushing in the stack S2, we are charging its cost as 2 coins)

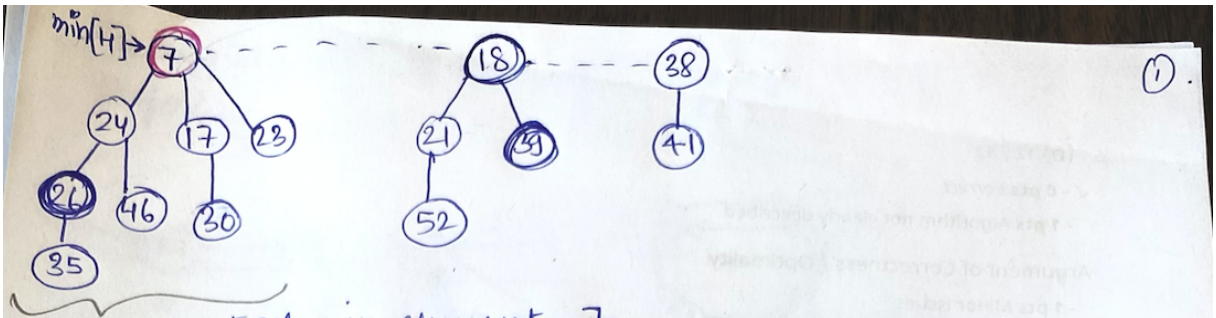
2.Then we pop the top element from S2. (this would cost 0 coins because we have charged a credit on previous push operation)

3.Lastly, we pop all elements from S2 and push it back to S1 (cost is 1 coin for pop from S2)
Overall , we can say that, :
every Enqueue takes O(1) AND every deque takes O(1) time.

---

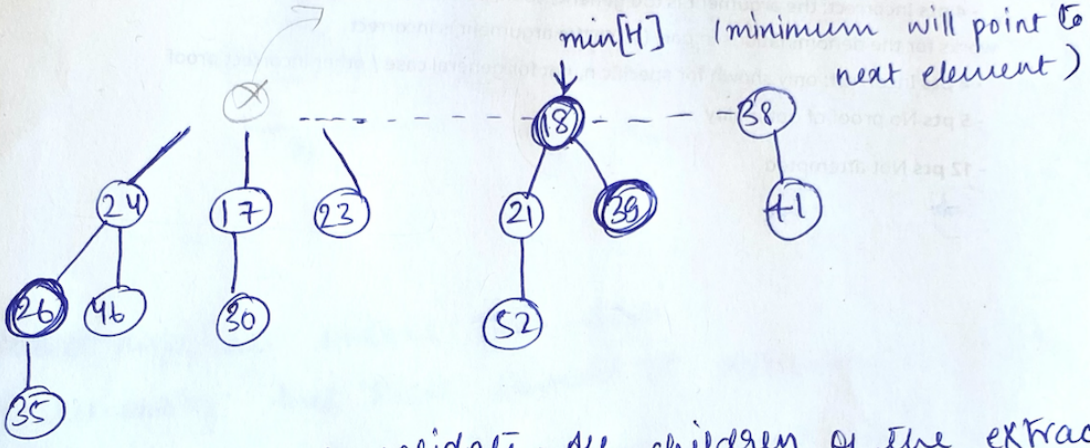
3. (25 points) Exercise 19.2-1.

Solution:
img attached :

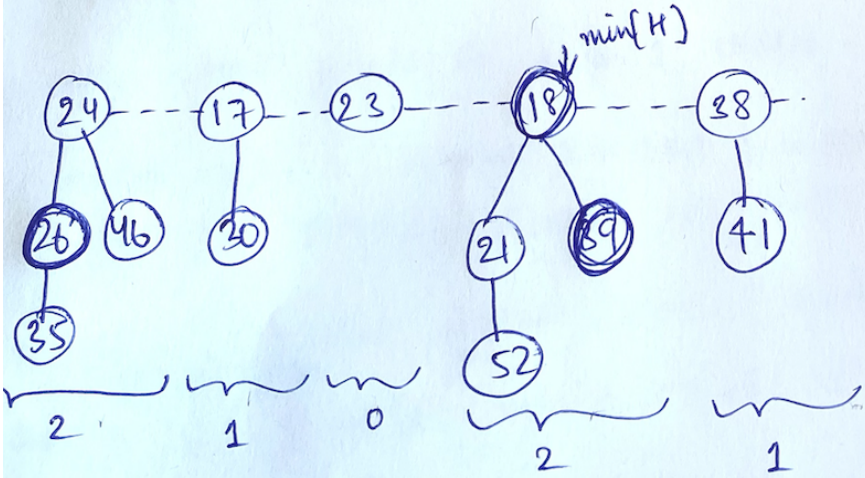


(1).

3 extract min element 7

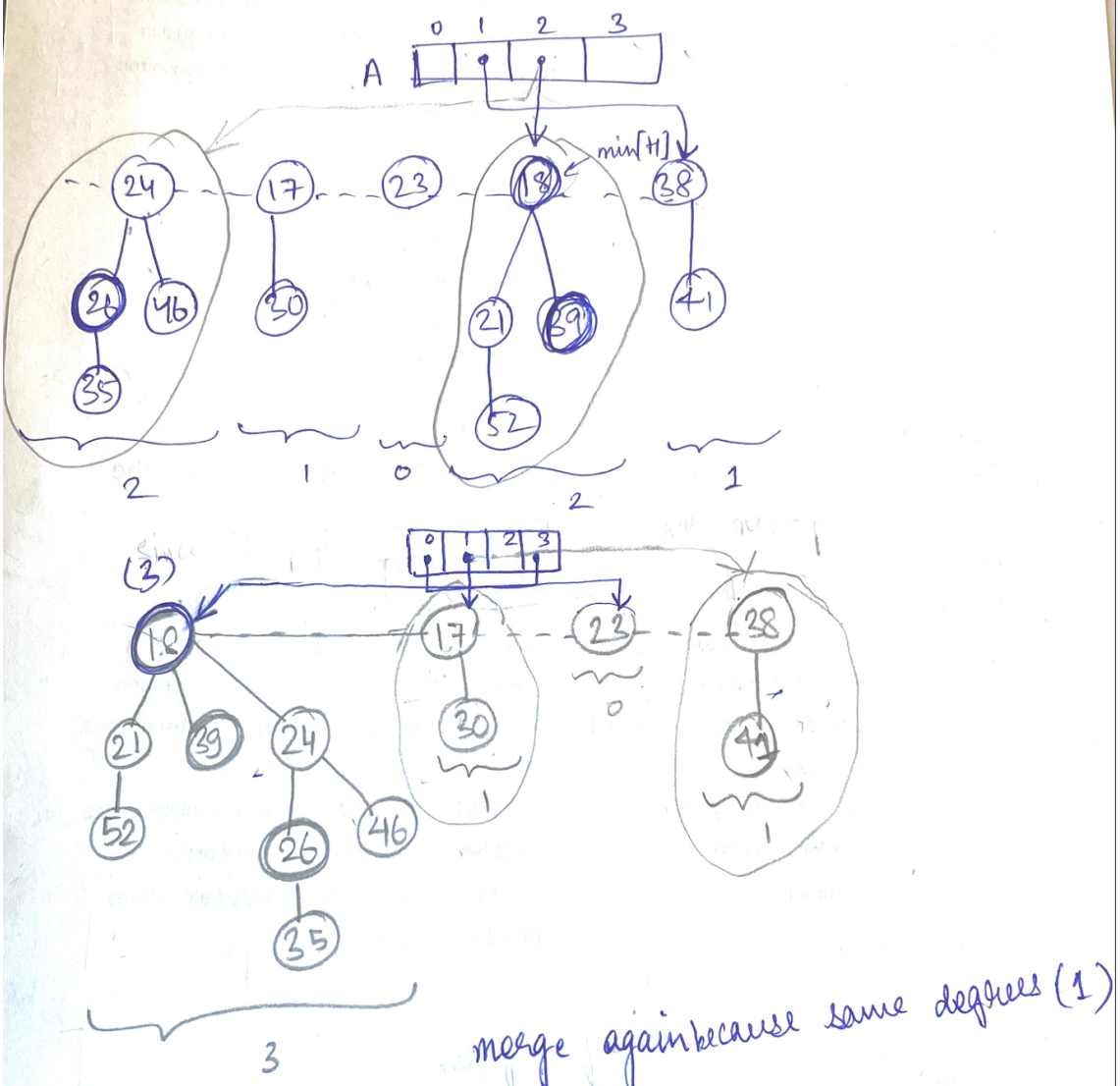


now we consolidate. All children of the extract min[H] become the root.

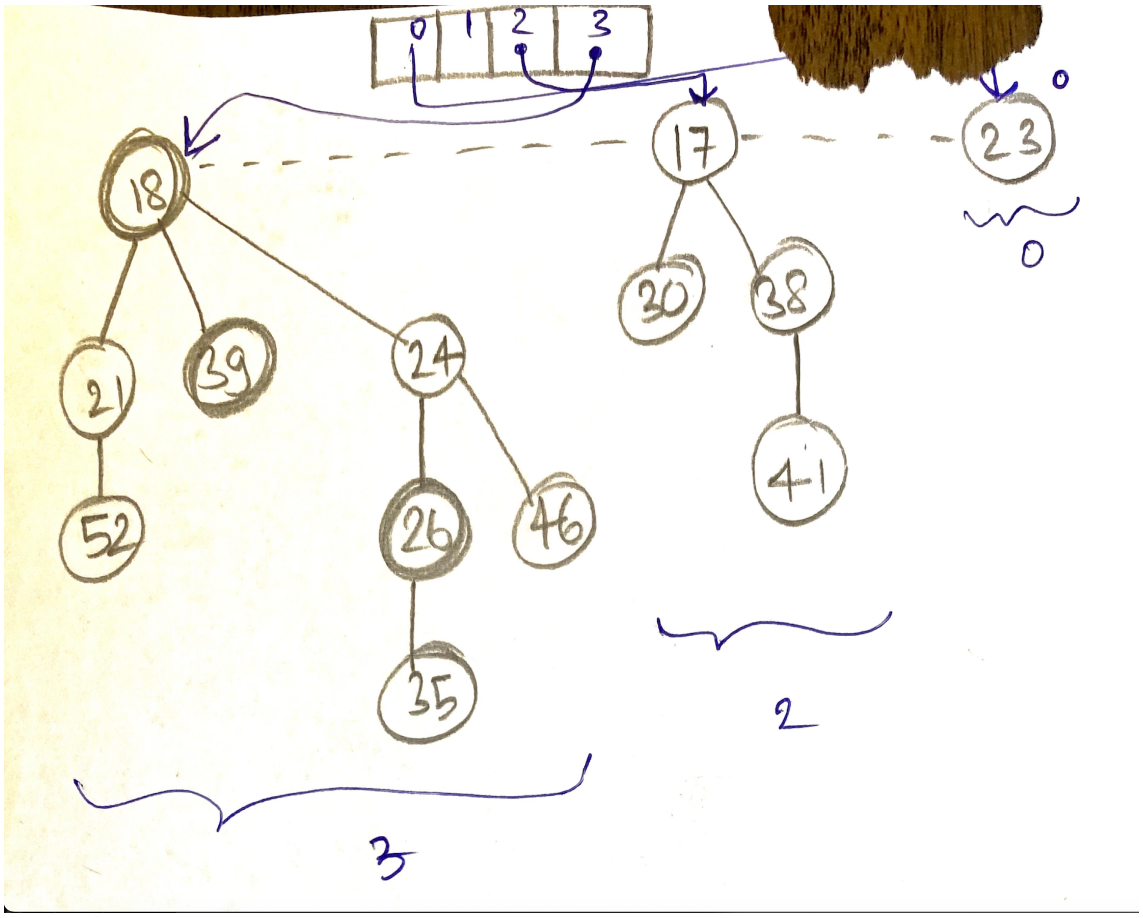


now we have to consolidate till there are DISTINCT degree trees. currently, we have duplicate degree.

create an array with size as max degree of the original heap (max degree = 3) (2)



merge again because same degrees (1)



---

4. (50 points) Implement binomial heaps as described in class and in the book. You should use links (pointers) to implement the structure as shown in the figure. Your implementation should include the operations: Make-heap, Insert, Minimum, Extract-Min, Union, Decrease-Key, Delete.

Make sure to preserve the characteristics of binomial heaps at all times:

(1) each component should be a binomial tree with children-keys bigger than the parent-key;

(2) the binomial trees should be in order of size from left to right. Test your code several arrays set of random generated integers (keys).

Solution:

zip folder attached

---

5. (Extra Credit) Find a way to nicely draw the binomial heap created from input, like in the figure.

6. (Extra Credit) Write code to implement Fibonacci Heaps, with discussed operations:ExtractMin, Union, Consolidate, DecreaseKey, Delete.

7. (Extra Credit) Figure out what are the marked nodes on Fibonacci Heaps. In particular explain how the potential function works for FIB-HEAP-EXTRACT-MEAN and FIB-HEAP-DECREASE-KEY operations.