Data Structure & Algorithms

Assignment 3. Innopolis University, Spring 2019

Name: Hany Hamed
Group Number: BS18-06

Section 2: Coding Part

(a) **Submission Number**: 52924609
Submission Link: https://codeforces.com/group/lk8Ud0ZeBu/contest/242929/submission/52924609.

1. *Proofs of* $O(\log n)$ ***for all operations:***

- ***insert()***
  Full proof is inside the code
  inserting is just put it in the correct order
  My algorithm is just: insert and use other function called: $merge_Tree$
  which order, organize and merge the trees in the heap
  $merge_Tree$ *is just traverse on the binomial heap then it is reordering the tree's roots by swapping them and merge if they have equal order*
  As the traverse on the heap acts only on the heads of the trees in the binomial heap and they are always $\leq \log(n)$
  Then $merge_tree$ *is always* $\log(n)$ *and insert is always* $\log(n)$

- ***max()***
  Full proof is inside the code
  Just traverse the heads, and compare them and get the most important one

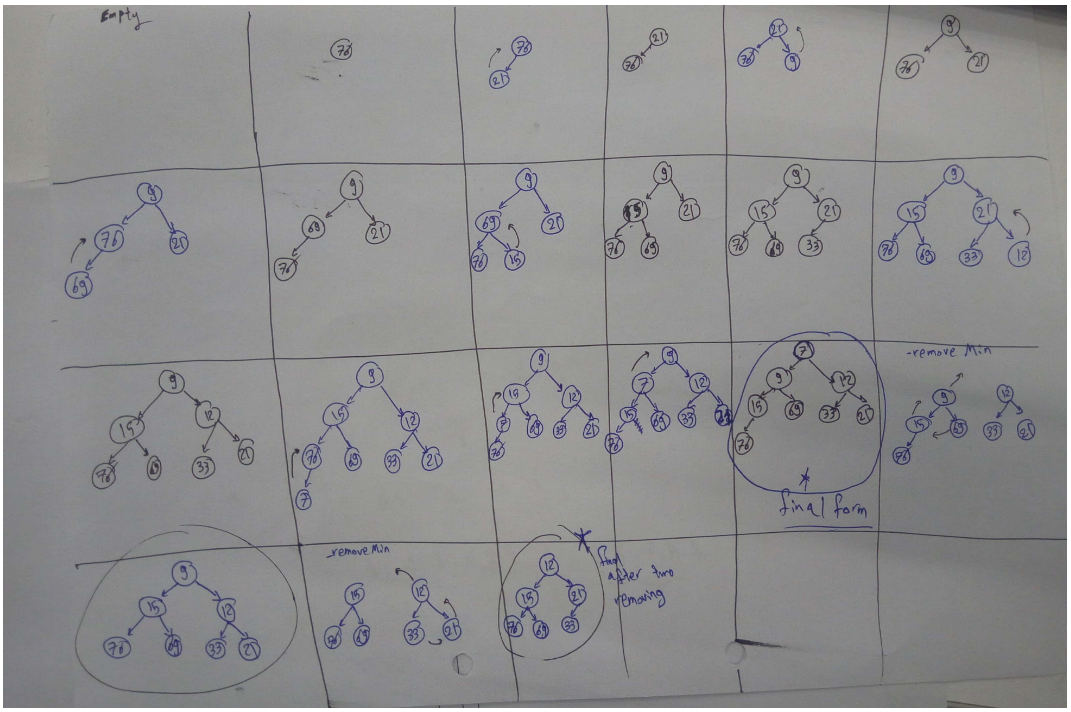
- ***removeMax()***
  Full proof is inside the code
  use max to get the maximum root, extract their children and insert them back in the heap.
  T(n) = O(log(n) + log(n)) = O(log(n))

- ***merge()***
  Full proof is inside the code
  It is similar to removeMax, but it has no removing or get the maximum

root, it is just extract them, then insert usthem back.
T(n) = O(logn+logn)

Section 3: Theoretical part

- **3.1 - Binary Heaps**
  - **a & b**



Link for the image: image:https://drive.google.com/file/d/10q0I2gkQHSPrc-sMI4YYfwNf1Jy3PFD0/view?usp=sharing.

  - **c.**
    The proof:
    1^{st} way:
    $\sum_{i=1}^{h} 2^i$ = maximum number of nodes in each level
    where h = height of the heap and n is the number of nodes in the heap

Then, $n \leq \sum_{i=1}^{h} 2^i = \frac{2^{h+1}-1}{2-1}$
$n \leq 2^{h+1} - 1$
$n + 1 \leq 2^{h+1}$
$\log(n+1) \leq h + 1$
$\log(n+1) - 1 \leq h$
as h is Natural number and if we represent $\log(n+1) - 1$ on number lines as in the lecture, then we can say that: (other reason: using the fact that $log(n+1) = floor(\log(n) + 1)$
then, $h = floor(\log(n))$
Q.E.D
2^{nd} way:
where h = height of the heap and n is the number of nodes in the heap
then,
$2h \leq n \leq 2^{h+1} - 1$
taking log for the inequality
then, $h \leq \log(n)$ and $log(n+1) \leq h + 1$
then, $\log(n+1) - 1 \leq h \leq log(n)$
as h is Natural number, then
then, $h = floor(\log(n))$
Q.E.D

- **3.2 - Binomial trees and binomial heaps**

  - **a.**
    We march x instead of merging the first two in order to have a correct order of the trees as all the trees should present in the heap in ascending order according to the degree of the tree.
    So, if we merge the first two of order n, there will be tree of order n+1 then tree of order n, then it will not merge correctly as maybe the tree after the third tree is of order n+1, so, it will not merged correctly.
    So, we march x in order to merge the last two tree, so it will have the correct order of the trees: n then n+1.

  - **b.**
    False statement
    As it is true that it will be equal to 2^d but if it is only complete binomial tree of order/height d
    but it is false that it will not exceed it.
    So, the correct that it is at most 2^d not at least.
    As the order of the binomial tree is the same as the height of the tree, then we can prove that complete binomial tree of height d has

exactly 2^d nodes using induction. Then it will follow that at most it will be 2^d not at least

The proof:

Base case: number of nodes in $B_0 = 1$ only single node
and $2^0 = 1$
then the base case is correct.

Hypothesis:

Assume that: number of nodes in $B_n = (num.B_n) = 2^d$ where d is the height of the tree which is the order of the tree.

Inductive step:

as $B_{n+1} = (B_n + B_{n-1}.... + B_0)$

then, $num.B_{n+1} = 1 + \sum_{i=1}^{n} 2^i$

$num.B_{n+1} = 1 + \frac{2^{h+1} - 1}{2 - 1}$

$num.B_{n+1} = 1 + 2^{h+1} - 1$

$num.B_{n+1} = 2^{h+1}$

Then it is proofed that if it is complete Binomial tree of height/order d it will be at most 2^d which will make our original proposition is false.

And we can say that always Binomial trees are always complete and we use Binomial Heap in order to have more than one Binomial Trees and not to have not Complete Binomial Trees.

Which means that Binomial Tree always has exactly 2^h not at least. Q.E.D

- **3.3 - Graph representation**

- **1.**

Nodes	in-degree	out-degree	degree
c	1	1	2
f	2	2	4

- **2.**
  No, it is not strongly connected as not every node is reachable from the the other nodes for example: a and g.

- **3.**

	a	b	c	d	e	f	g	h	i
a	0	1	0	0	0	0	0	0	0
b	0	0	0	1	0	1	0	0	0
c	0	1	0	0	0	0	0	0	0
d	0	0	1	0	0	0	0	0	0
e	0	1	0	0	0	0	0	0	0
f	0	0	0	0	1	1	0	0	0
g	0	0	0	0	1	0	0	0	0
h	0	0	0	0	0	0	0	0	1
i	0	0	0	0	0	0	1	0	0

- **The proof:**
  proof of Handshack Theorem which is degree sum formula which says that there is odd number of nodes which have odd degree according to Wikipedia.
  1^{st} **way: Induction**
  Base case: number of nodes (n) = 2
  then the possible graphs is empty or K_2 is a complete graph.
  In empty there will 0 nodes with odd degree where 0 is even.
  In complete, there will be 2 nodes with odd degree where 2 is even.
  Then the base case is correct Inductive Hypothesis: Assume that there is even number of odd degree nodes in any undirected graph which have n nodes is true.
  Inductive Step:
  if we have extra node then if want to connect it to the graph at least with one edge.
  If we connected it with even degree node, it will be odd degree node and our new node will be odd degree node. As the number of odd degree nodes were even before the addition then num.odd += 2 then it will even too.
  If we connected to an odd node, it will be even and the the added node will be odd. num.odd += -1 +1 = num.odd which will remain Even number of odd degree nodes in any undirected graph.
  And adding one edge it is the same as two edge between the added node and the graph so, it will remain having this property.
  Q.E.D
  2^{nd} **way: Contradiction**
  If we assume that there will be a graph that will have odd number of odd degree nodes
  As the sum of the degree of the nodes is always even $\sum_{v} degv = 2|E|$ according to Handshake Theorem, then if we have odd number of nodes which have odd degree the sum will be odd number but it should be an even, so, it contradict with Handshaking theorem, then it is false the assumption.
  Q.E.D