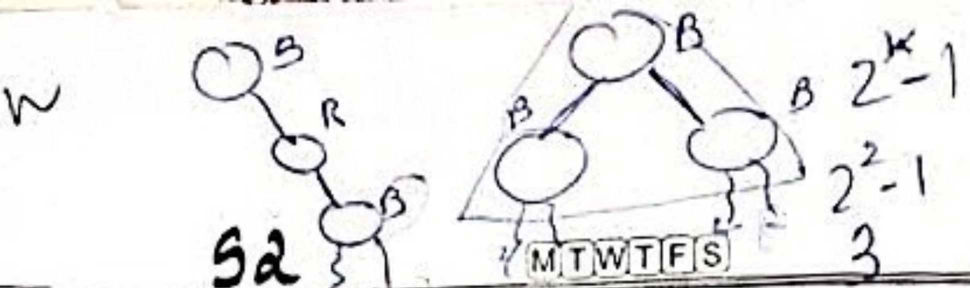

Q In Tree representation of a 'n' element heap the height is $\Theta \lg n$ or $O \lg n$. justify your answer
Let $h = \Theta(\lg n)$
Since heap is always a balanced tree becz it holds the property of CBT. We know that CBT grows from left to right and top to bottom. So there is no chance for the imbalaced tree. There is no possibility to decrease hight from $\log n$ and increase from $\log n$. So its height will always be $\log n$

Q largest and smallest number of internal node in read-black tree?

The largest possible number of internal node in RBT is $2^{k+1}-1$. Because in binary tree maximum number of nodes ar $2^{h+1}-1$ and in RBT with black height k has at most k+1 level including root and leaf levels.

The smallest is 2^k-1 because when the internal node have 2 black children so ~~as~~ ~~th~~ so its value will 2^k-1.

Q An array is filled with n elements representing piority values. Figure out whether it is piroriy queu or not
we will deal with this array $T(n) = n$ so since half of array is already heap so,
$n/2 \rightarrow$ to check max heap
$n/2 \rightarrow$ to check min heap
$T(n) = n/2 + n/2$
$= \Theta(n)$

Date______ M T W T F S

Q When does worst-case running time of Quick sort happen?

~~Worst case of quick Sort will happen when q=1 or q=n where q is random index.~~

$T(n) = T(n-2) + T(1) + \Theta(n)$

~~$\Theta(n)$ = partining time~~

base condition ~~$T(1) = \Theta(1)$~~

So

$T(n) = T(n-2) + (n+1) \quad — (i)$

$T(n-2) = T(n-2-2) + (n-2+1)$

$= T(n-4) + (n-1) \rightarrow$ put in (i)

$= T(n-4) + (n-1) + (n+1)$

$T(n-k) + \sum_{i=0}^{k-3} (n-i) + 1$

$n-k = 1$

$n = k$

$T(k-k) + \sum^{k-3} (n-i) + 1$

~~$T(0) + \sum_{i=0}^{k-3} (n-i) + 1$~~

so total running time is

$\Theta(n^2)$

Date MTWTFS

Q If black-height of any node in RBT is x. what we can conclude about the black height of it both children.

In RBT, every path from root to leaf node have same number of black nodes. If a node has black height of x then its children must also have a black height of x. If one children have black height less than x it would violate the req. Therefore the black height of a node must have same number of black height for both children

Q Suppose you want to sort n number each of values 0s and 1s in array. Write names of two asymptotically effient algorith and their time

→ ~~Counting Sort: works by counting number of occurence of each value in the array and then placing~~ each value in its correct position in output array. Time complexity of counting sort is $O(n)$.

→ Quick Sort: This algorithm can be used to sort array in $O(n\log n)$ time in avg. Worst case is $O(n^2)$ if pivot is not chosen optimal. Best case is $O(n)$ in this if pivot value is used 0.

Q Show that longest path from node x in RBT to any descendant leaf has lenght at most twice that of shortest path from node x to any descendant leaf.

In longest path atleast every other node is black. In the shortest path, at most every node is black. Therefore two path contains equal number of black nodes, the length of longest path is at most twice

Scanned with CamScanner

ps://chat.openai.com/chat

given an unsorted array of n integers, a max heap of elements of A can be performed asympotically faster than building a RBT out of elements of A?

In general, building a max heap of n elements can be done in O(n) time, while building a red-black tree (RBT) from n elements takes O(n log n) time in the worst case. Therefore, for large values of n, building a max heap is asymptotically faster than building an RBT.

The reason for this difference in running time is that building a max heap can be done in linear time using the bottom-up heap construction algorithm, which starts from the leaves of the heap and works its way up to the root, adjusting the heap property as needed. On the other hand, building an RBT involves inserting elements one at a time into the tree, which can cause tree rotations and color changes to maintain the RBT properties, leading to a worst-case running time of O(n log n).

However, it's w Regenerate response ce between a max heap and an RB c use case and the

Send a message...

ChatGPT Mar 23 Version. Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts

Therefore, to ensure that the dynamic programming algorithm produces a globally optimal solution, we need to compare the solutions obtained by combining the optimal solutions to all subproblems and choose the one that maximizes or minimizes the objective function, depending on the problem. This step is typically performed using a recurrence relation or a table lookup in the dynamic programming algorithm. reduce in 2 to 4 lines

To ensure a globally optimal solution in a dynamic programming problem, we need to:

1. Use the optimal substructure property to break the problem into subproblems
2. Solve each subproblem optimally
3. Combine the optimal solutions to all subproblems to obtain the globally optimal solution
4. This step is typically performed using a recurrence relation or a table lookup

Regenerate response

Send a message...

ChatGPT Mar 23 Version. Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts

In a complete binary tree, the number of nodes in each level doubles as we move down the tree. Therefore, the height of a complete binary tree with n nodes is log2(n+1) - 1, rounded down to the nearest integer.

A heap is a complete binary tree, so its height can be determined using the same formula. Therefore, the height of a heap with n nodes is log2(n+1) - 1, rounded down to the nearest integer.

In asymptotic notation, the height of a heap can be expressed as O(log n), where n is the number of nodes in the heap. This means that the height of the heap grows logarithmically with the number of nodes in the heap.

Regenerate response

Send a message...