Time Complexity of building a heap

Consider the following algorithm for building a Heap of an input array A.

```
BUILD-HEAP(A)
    heapsize := size(A);
    for i := floor(heapsize/2) downto 1
        do HEAPIFY(A, i);
    end for
END
```

What is the worst case time complexity of the above algo?
Although the worst case complexity looks like O(nLogn), upper bound of time complexity is O(n). See following links for the proof of time complexity.

http://www.cse.iitk.ac.in/users/sbaswana/Courses/ESO211/heap.pdf/
http://www.cs.sfu.ca/CourseCentral/307/petra/2009/SLN_2.pdf

Related Topics:

- Linearity of Expectation
- Iterative Tower of Hanoi
- Count possible ways to construct buildings
- Build Lowest Number by Removing n digits from a given number
- Set Cover Problem | Set 1 (Greedy Approximate Algorithm)
- Find number of days between two given dates
- How to print maximum number of A's using given four keys
- Write an iterative O(Log y) function for pow(x, y)

Writing code in comment? Please use **ideone.com** and share the link here.

13 Comments **GeeksforGeeks** 1 Login

Recommend Share Sort by Newest

Join the discussion…

Tyrion • 2 months ago
http://www.cs.umd.edu/~meesh/3...
Reply • Share ›

karna • 4 months ago
Relevent: http://stackoverflow.com/quest...
Reply • Share ›

Ankit • 4 months ago
How is a upper bound different from worst case time complexity and when one makes more sense over other.

And is tight upper bound any different ?
Reply • Share ›

Ankit Kapur → Ankit • 3 months ago
They're nearly the same. The tightest upper bound is exactly the same thing as worst case time complexity
Reply • Share ›

karna → Ankit • 4 months ago

karna → Ankit • 4 months ago

http://stackoverflow.com/quest...

• Reply • Share ›

Sriram Ganesh • 10 months ago

http://www.cse.iitk.ac.in/user...

This link is not working.

• Reply • Share ›

groomnestle • a year ago

It is easier to write heapsort with first heap index as 1 instead of 0, in this case you can correlate parent and children with a simple formula:
parent = k, left child = 2k, right child= 2k+1.

1 • Reply • Share ›

Venki • 4 years ago

The complexity O(n log n) is an upper bound. The upper bound is calculated assuming that every node is of height log n. But in reality it is not the case. Height of node varies in the heap. By definition, height of a node is the longest path length from that node to leaf node. This way the root is at maximum height, followed by next level subtrees.

The build-heap time complexity is function of node's height f(h). How can we express f(h)? Given a binary-heap having N elements, we can observe that there will be maximum of f(h) = [N/2^(h+1)] nodes at height h. For example, given N = 7 i.e. a full binary tree of height 2, the root is at height of 2 and f(2) = 7/8 = 0, f(1) = 7/4 = 1, f(0) = 7/2 = 3, all satisfying our assumption. Infact, it is an approximation and as the height of tree increases our approximation reach exact limit.

Now, it is easy to check the tighter bound on building the heap. To build heap, we call heapify() whose complexity is O(h) and we call it on nodes from nodes n/2 to 1. These nodes form internal nodes of binary heap. To find exact computational cost we need to sum the cost of heapifying each node. Yet we are interested in asymptotic bound. We need to find the cost of *a node* at each level and integrate (sum) them to find the complexity.

Mathematically T(n) = summation of [h x f(h)] in the interval 0 to logN (i.e. height of tree). After

see more

6 • Reply • Share ›

Sandeep → Venki • 4 years ago

@Venki: Both O(n) and O(nLogn) are upper bounds for build heap. O(n) is tighter upper bound.

3 • Reply • Share ›

tk • 4 years ago

Ineresting fact. Good to know.

• Reply • Share ›

Vinay • 4 years ago

one conceptual question from Cormen:why the loop goes down from heapsize/2 to 1 and not increment from 1 to heapsize/2?

• Reply • Share ›

Amit → Vinay • 4 years ago

If you index i=1 to n/2 you can observe that you will need to modify your Heapify function to work (try to apply this and you will able to see the differnece).... so in Cormen to avoid this modification we start with i=n/2 to 1

Both indexing will work but the efficient way will be the latter case....

• Reply • Share ›

kartik → Vinay • 4 years ago

I think its because the way Heapify process works. When we Heapify a node at index i, we assume that all the subtrees of i are heapified. Also, the Heapify process must go in upward direction to make sure that the maximum (or minimum) element is at the top.

4 • Reply • Share ›

-
-
-
-
  - Interview Experiences
  - Advanced Data Structures
  - Dynamic Programming
  - Greedy Algorithms
  - Backtracking
  - Pattern Searching
  - Divide & Conquer
  - Mathematical Algorithms
  - Recursion
  - Geometric Algorithms
-

- ## Popular Posts

  - All permutations of a given string
  - Memory Layout of C Programs
  - Understanding “extern” keyword in C
  - Median of two sorted arrays
  - Tree traversal without recursion and without stack!
  - Structure Member Alignment, Padding and Data Packing
  - Intersection point of two Linked Lists
  - Lowest Common Ancestor in a BST.
  - Check if a binary tree is BST or not
  - Sorted Linked List to Balanced BST

-
- ## Recent Comments
  - lt_k

    i need help for coding this function in java...

    Java Programming Language · 1 hour ago
  - Piyush

    What is the purpose of else if (recStack[*i])...

    Detect Cycle in a Directed Graph · 1 hour ago
  - Andy Toh

    My compile-time solution, which agrees with the...

    Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago
  - lucy

    because we first fill zero in first col and...

    Dynamic Programming | Set 29 (Longest Common Substring) · 1 hour ago
  - lucy

    @GeeksforGeeks i don't n know what is this long...

    Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 2 hours ago
  - manish

    Because TAN is not a subsequence of RANT. ANT...

    Given two strings, find if first string is a subsequence of second · 2 hours ago

-