Modify the algorithm of max-heap such that two elements are pushed in each iteration

Arun Kumar Reddy
IRM2014005
Email:irm2014005@iiita.ac.in
IIIT Allahabad

Swarnima
IWM2014003
Email:iwm2014003@iiita.ac.in
IIIT Allahabad

Akhila Jetty
IRM2014006
Email:irm2014006@iiita.ac.in
IIIT Allahabad

***Abstract*—The objective of this paper is to modify the implementation of heap to insert two elements into the heap in a single operation.**

Keywords*—*Max-heap, Implementation.

I. Introduction

Heap is a very important data structure in computer science. It is used for an efficient implementation of a priority queue. Heap sort which makes use of heaps is one of the best sorting algorithms as it is in place and has no worst case quadratic scenarios. A Heap can be defined as a partially ordered tree. It is ordered because the every node in the heap satisfies a property. There are mainly two types of heaps - Minheap and Maxheap. In a minheap, the value of the parent is lesser than the value of its children and vice-versa for the maxheap. Heaps are used to store data as they have a logarithmic run time for both the insert and delete operations.Heaps are also used to reduce run time in graph algorithms.

II. Motivation

Heap is a very significant data structure as priority queues have many applications in scheduling etc. Heap sort is one of the best sorting algorithms due to its different properties. The motivation was to understand and implement the heap data structure. The traditional heap data structure can be modified to make it more efficient for certain operations. The idea was to modify the insert operation of the heap such that two elements are inserted into the heap in the same iteration.

III. Implementation

The traditional max-heap implementation was slightly modified in order to insert two elements in the same iteration. The idea is to insert two elements into the heap at the same iteration. Two new nodes are added in the heap and the nodes are recursively compared with their parents until the root is reached. Each node is compared with its parent and if greater their values are exchanged and this process is recursively repeated for the parent until root node is reached. The program was implemented in C++ and the time complexity of the insert operation is of the order O(log n) where n is the number of nodes in the heap. As the heap is implemented as a balanced binary tree, The height of the tree will never exceed log n and therefore the operation has a logarithmic complexity.

A. Approach

```
1. MAX-HEAP-INSERT(A, key1, key2)
2.         heap-size[A] = heap-size[A]+2
3.         index = heap-size[A]
4.         A[index-1] = key1
5.         A[index] = key2
6.         i = index-1
7.         while(i > 1 and A[parent(i)] < A[i])
8.                 exchange A[parent(i)] and A[i]
9.                 i = parent(i)
10.        i = index
11.        while(i > 1 and A[parent(i)] < A[i])
12.                exchange A[parent(i)] and A[i]
13.                i = parent(i)
```

IV. Conclusion

Therefore the insert operation of the max-heap data structure was modified to insert two elements into the max-heap in the same operation. The plot of the run time of the operation against the number of nodes in the heap is shown below.

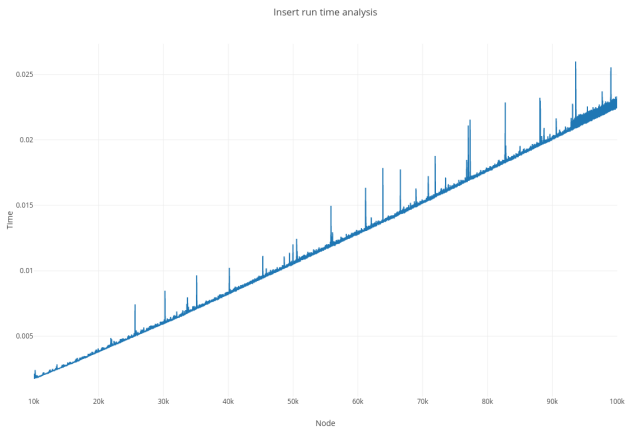

Fig. 1. Run time analysis of the Insert operation

References

[1] Heap(data structure) - $https://en.wikipedia.org/wiki/Heap$