CSI33 Data Structures

Sharon Persinger

Fall 2019

Day 25 December 2

Topics

Priority Queue

Binary heap:

- definitions
- implementation

Heapsort

Priority queue with heap

Priority queue

A container abstract data type. Every item has data and also a priority.

The item on the queue with highest priority is dequeued first.

Some specifications provide that items with the same priority are dequeued in the unusual FIFO order for a queue.

uses: printer queue, operating systems in general, medical systems - emergency room

Specification of Priority Queue ADT in Python

```
Class PQueue(Object):
    def enqueue(self, item, priority):
        '''post: item is inserted with specified priority in the PriorityQueue'''
    def first(self):
        '''post: returns but does not remove highest priority item from the PriorityQueue'''
    def dequeue(self):
        '''post: removes and returns the highest priority item from the PriorityQueue'''
    def size(self):
        ''' post: returns the number of items in the PriorityQueue'''
```

Implementation ideas:

Simplest implementation is to insert the items into a list in order sorted by priority.

It is possible to do this in an order that maintains the FIFO property when items are dequeued.

Enqueue requires \Theta(n) time in the worst case, since inserting at the beginning - high priority - requires shifting all the items down.

Dequeue requires \Theta(n) time in the worst case, since removing at the beginning - high priority - requires shifting all the items up.

Linked list implementation takes \Theta(n) time in the worst case to find the correct location for insertion.

Binary Heap Data Structure

Can be used to implement a priority queue,

Also can be used to implement a sorting algorithm

Based on a complete binary tree- a binary tree in which every level, except possibly the last, is completely filled, and in the lowest level, every node is as far to the left as possible.

Binary Heap Data Structure

heap definition:

A complete binary tree in which the data at each node satisfies a simple order property- the value at a node is greater than or equal to the value at each of its children.

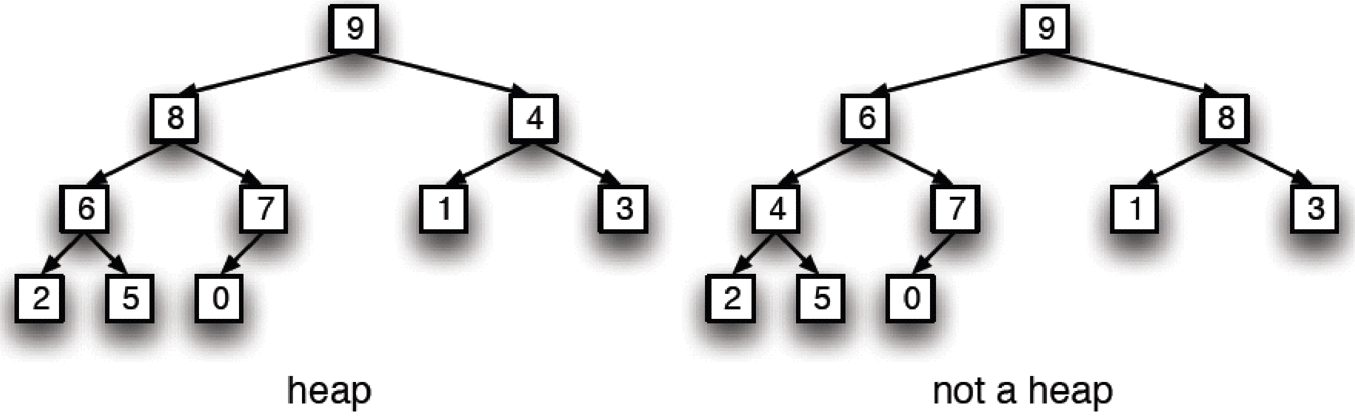

Figure 13.1: Two complete trees with only the left one satisfying the heap property

Heap operations – for priority queue

Dequeue:

- highest priority item is at the root, so easy to find it to remove.
- How to put the correct item into its place and maintain the heap property?
- Obvious item to put in its place is the larger of its two children.
- Continue this process on down, but not assured to produce a complete binary tree.
- Could end up with a gap at the lowest level.
- Think of the process this way: Remove the root and put the rightmost item from the lowest level in its place. Move that item now at the root down by swapping it with the larger of its two children. Continue until the heap property is satisfied.

Dequeue

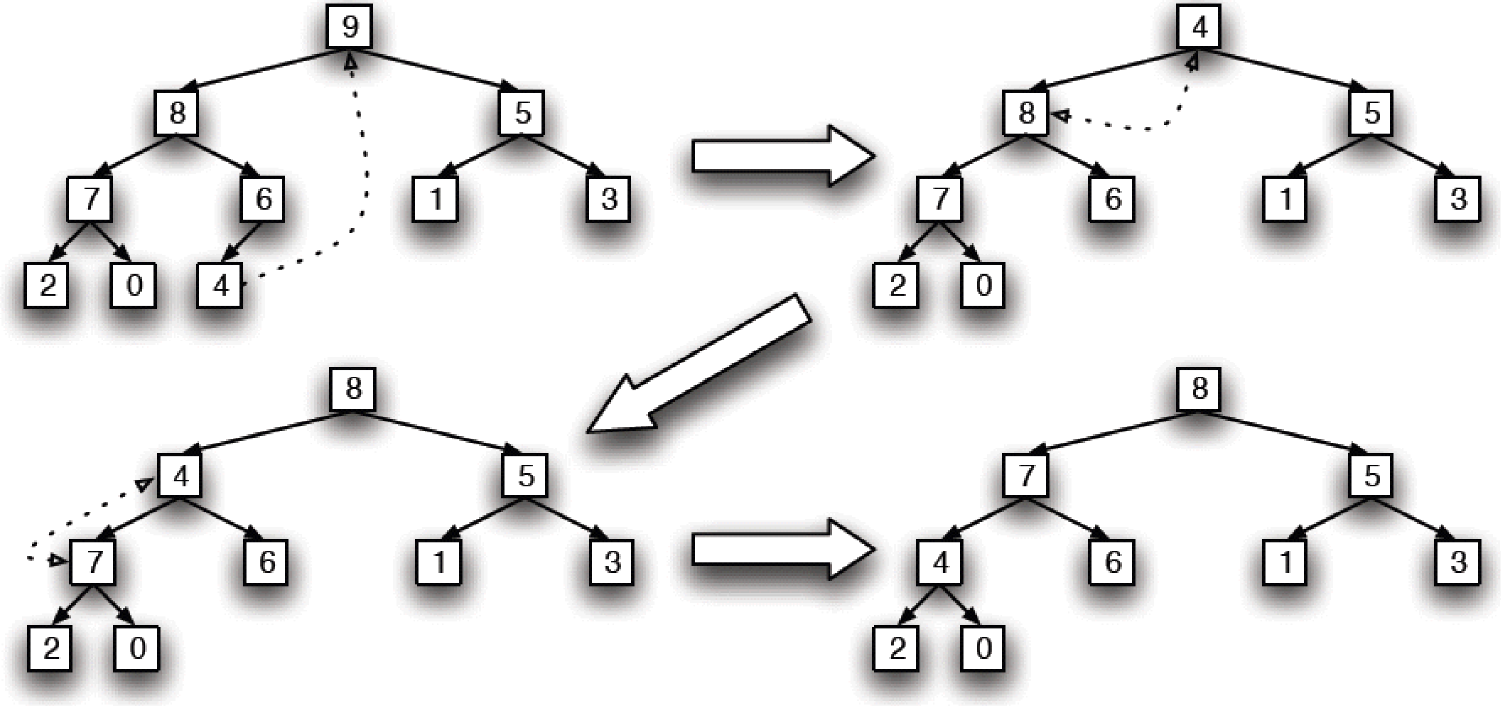

Figure 13.2: Removing 9 and reorganizing the heap

Enqueue similarly

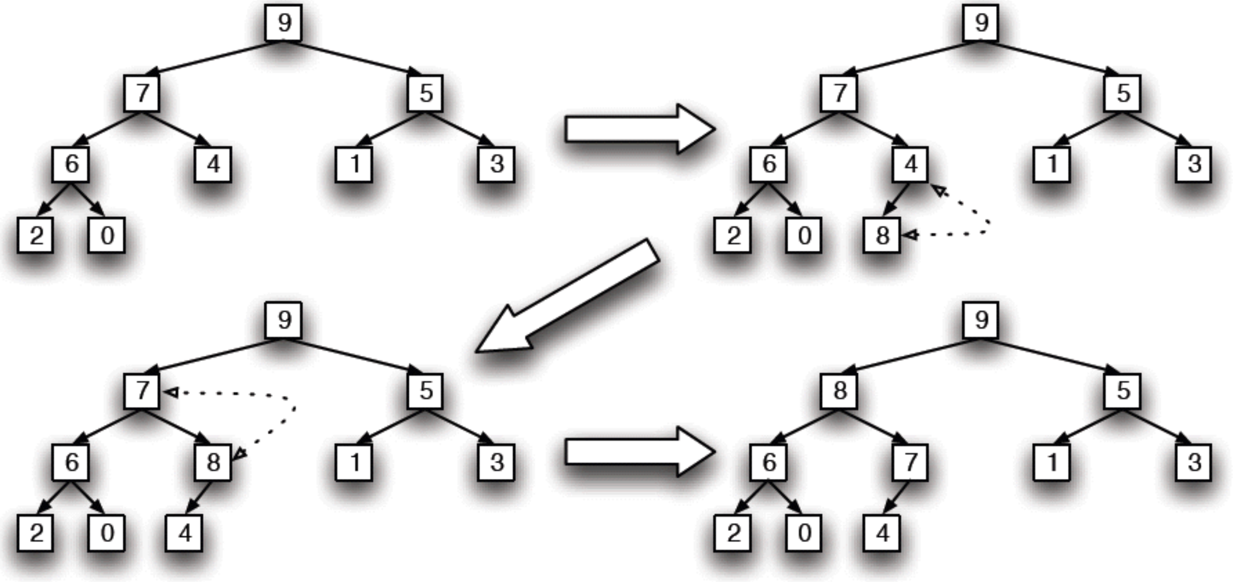

Figure 13.3: Inserting the 8 and reorganizing the heap

Place the new item in the left-most available position.

If the item is larger than its parent, swap them. Continue until the item is less than or equal to its parent.

Binary heap visualizer

http://btv.melezinek.cz/binary-heap.html

Time analysis for enqueue, dequeue

$\Theta(log_2 n)$ since the number of steps is less than the depth of the tree.

Implementation of heap

- First, remember that a complete binary tree can be sored in a Python list (or C++ array).
- We will store the root node at index 1, since that makes the computations of the index of parent or child node easier.
- For any node of index k,
  - the left child has index 2*k and
  - the right child has index 2*k+1.
  - The parent of node k has index k//2.

Implementation of Heap class

- A heap object has
  - Member variables: heap, a list that stores the items of the object, and heap_size, an integer that stores the number of items in the heap.
  - Member methods:
    - Constructor -- makes empty heap by default, or builds a heap from a list of items
    - size method -- returns the number of items
    - delete_max -- removes and returns the maximum item, restores the heap property afterward. $\Theta(log_2 n)$ time.

Implementation of Heap class

- Member methods, continued:
  - insert – inserts a new item into the heap in the proper location by moving it up the tree $\Theta(log_2 n)$ time
  - _heapify – helper function that restores the heap property by moving an item down the tree to the correct position. $\Theta(log_2 n)$ time
  - _build_heap – takes values stored in self.heap and arranges them to satisfy heap property. $\Theta(n\ log_2 n)$ time
  - heapsort – rearranges items in self.heap into a list in increasing order. Heapsort is an in-place $\Theta(n\ log_2 n)$ sorting algorithm.

Review the implementation

Heap.py

Priority queue operations

Insert is enqueue operation: Time is $\Theta(\log n)$

delete_max is dequeue operation: Time is $\Theta(\log n)$