问题与讨论

2014/6/12

6.1-2

Show that an n-element heap has height $\lfloor \lg n \rfloor$.

设树高为h

$$2^h \le n \le 2^{h+1} - 1$$

$$\Rightarrow h \le \lg n < h + 1$$

$$\Rightarrow \lg n - 1 < h \le \lg n$$

$$h = \lfloor \lg n \rfloor.$$

6.1-7

Show that, with the array representation for storing an n-element heap, the leaves are the nodes indexed by $\lfloor n/2 \rfloor + 1, \lfloor n/2 \rfloor + 2, \ldots, n$.

对于结点i，如果 2i<= n，说明结点i有左孩子，因此i不是叶子结点。

如果结点i是叶子结点，则2i>n，即 $i \geq \left\lfloor \frac{n}{2} \right\rfloor + 1$

因此，叶子结点是 $\lfloor n/2 \rfloor + 1, \lfloor n/2 \rfloor + 2, \ldots, n.$

6.2-5

The code for MAX-HEAPIFY is quite efficient in terms of constant factors, except possibly for the recursive call in line 10, which might cause some compilers to produce inefficient code. Write an efficient MAX-HEAPIFY that uses an iterative control construct (a loop) instead of recursion.

```
1  Max-Heapify(A, i)
2      while true
3          l = Left(A, i)
4          r = Right(A, i)
5          if l <= A.heap-size and A[l] > A[i]
6              largest = l
7          else
8              largest = i
9          if r <= A.heap-size and A[r] > A[largest]
10             largest = r
11         if largest != i
12             swap A[i] with A[largest]
13             i = largest
14         else
15             break
```

6.3-3

Show that there are at most $\lceil n/2^{h+1} \rceil$ nodes of height h in any n-element heap.

当h=0时，叶子结点数量是 $\lceil n/2 \rceil = \lceil n/2^{0+1} \rceil$ (Exercise6.1-7)

假设当高度为h-1时命题成立，证明当高度为h时也成立。

n_h 表示高度为h的结点的个数

n'_h 表示去掉叶子结点后，高度为h的结点的个数

由于去掉叶子结点后，所有结点的高度减1，所以有 $n_h = n'_{h-1}$

去掉叶子结点后，树中所有结点的个数为：$n - \lceil n/2 \rceil = \lfloor n/2 \rfloor$

$$n_h = n'_{h-1} \le \lceil \frac{n'}{2^h} \rceil = \lceil \frac{\lfloor n/2 \rfloor}{2^h} \rceil \le \lceil \frac{n/2}{2^h} \rceil = \lceil \frac{n}{2^{h+1}} \rceil$$

6.5-9

Give an $O(n \lg k)$-time algorithm to merge k sorted lists into one sorted list, where n is the total number of elements in all the input lists. (*Hint:* Use a min-heap for k-way merging.)

建一个大小为k的堆，堆中的每个元素代表一个List，元素的key为List当前最小元素的值。

每次取出堆顶的元素，存入结果数组。然后插入与堆顶元素对应List中下一个元素的值，如果该List没有下一个元素则删除该结点重新建堆，直到n个元素归并完毕。