Heapsort

Chapter 6

Algorithm Design Techniques

1.

2.

3.

Trees

- What's the difference between a graph and a tree?
- Free Tree:
- Forest:
- Rooted Tree:
- Node vs. Vertex

Tree Properties

- Binary tree?
- Depth of the node?
- Height of a node?
- Height of the tree?
- **Full** binary tree?
- **Complete** binary tree?
- **Perfect** binary tree?

Tree Terminology

- Ancestor / Descendant
- Proper Ancestor / Proper Descendant
- Parent / Child / Siblings
- Leaf / External Node / Internal Node
- Degree of Node x

Facts about Perfect Binary Trees

Complete Binary Trees

- Nodes at depth h (the lowest level) are as far left as possible
- What is the relationship between the height and the number of nodes?

Heaps

- A ***heap*** is a complete binary tree
- Extra nodes go from left to right at the lowest level
  - Where the value at each node is ≥ the values at its children (if any)
  - This is called the ***heap property*** for max-heaps
- Max or Min Heap

Example

Heap vs. Heap

- Where have you seen the word heap before?

Storing Heaps

- As arrays!

- Root of tree is:
- Parent of A[i] is:
- Left child of A[i] is:
- Right child of A[i] is:

Example

- n = 13

92 85 73 81 44 59 64 13 23 36 32 18 54

Functions on Heaps

- MAX-HEAPIFY
- BUILD-MAX-HEAP
- HEAPSORT
- MAX-HEAP-INSERT
- HEAP-EXTRACT-MAX
- HEAP-INCREASE-KEY
- HEAP-MAXIMUM

MAX-HEAPIFY, p 154

Max_Heapify(A, i) // A: Array, i: int

```
1   int L = left(i)
2   int r = right(i)
3   if (L <= A.heap_size and A[L] > A[i] )
4     largest = L
5   else Largest = i
6   if (Right <= A.heap_size and A[R]> A[largest])
7     largest = R
8   if largest != i
9     swap ( A[i], A[largest] )
10    Max_Heapify(A, largest)
```

Example

- 15 6 4 8 5 3 1 2 7 i = 2

Build_Max_Heap, p 157

Build_Max_Heap (A) // A: Array

```
1   A.heap_size = A.length
2   for i = floor ( A.length/2) to 1
3     Max_Heapify(A,i)
```

Example

- 4 3 7 13 1 20 12 16 2 18

HeapSort, p 160

HeapSort(A) // A: Array

1	`Build_Max_Heap(A)`
2	`for i = A.length to 2`
3	`  swap(A[1], A[i])`
4	`  A.heap_size = A.heap_size - 1`
5	`  Max_Heapify(A, 1)`

Example

- 20 18 12 16 3 7 4 13 2 1