CS 241
Data Organization
Heapsort

Brooke Chenoweth

University of New Mexico

Fall 2014

Heapsort algorithm

- Make heap
- While heap is not empty
    - Remove largest item
    - Restore heap property

What is a heap?

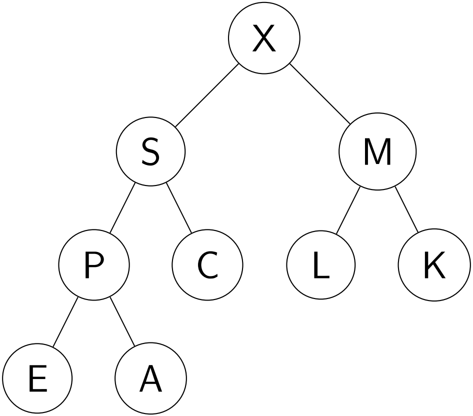
- Complete Binary Tree
  - Binary Tree: Each node has at most 2 children
  - Complete: All levels of tree are full (except maybe last)
- Satisfies *heap property* for all nodes
  - Heap Property: Parent $\geq$ Child
- Largest value is at the root.
- Subtrees are also heaps.

Complete Binary Tree as Array

We can represent a complete binary tree as an array.

- Root is at index zero
- For a node at index i:
    - Left child is at index $2i + 1$
    - Right child is at index $2i + 2$

Example Heap



X	S	M	P	C	L	K	E	A

Heapsort: swap

```c
#include <stdio.h>

void swap(char a[], int i, int j)
{
  char tmp = a[i];
  a[i] = a[j];
  a[j] = tmp;
}
```

Heapsort: siftDown

```c
void siftDown(char a[], int i, int n)
{
  int left = 2*i+1;
  int right = 2*i+2;
  int largest = i;
  /* Is a child larger than this node? */
  if(left < n && a[left] > a[largest])
  { largest = left;
  }
  if(right < n && a[right] > a[largest])
  { largest = right;
  }
  /* if child is larger, swap and fix subtree */
  if(largest != i)
  { swap(a, i, largest);
    siftDown(a, largest, n);
  }
}
```

Heapsort: heapify

```c
void heapify(char a[], int n)
{
  int i;
  for(i = (n-2)/2; i >= 0; i--)
  {
    siftDown(a, i, n);
  }
}
```

Heapsort: heapsort

```c
void heapsort(char a[], int n)
{
  int end;
  heapify(a, n);
  for(end = n-1; end > 0; end--)
  {
    printf(" Sorting: %s, end=%d\n", a, end);
    swap(a, 0, end);
    siftDown(a, 0, end);
  }
}
```

Heapsort: main

```c
void main(void)
{
  char data[] = "CELKMSPXA";

  printf("Original: %s\n", data);
  heapsort(data, 9);
  printf("  Sorted: %s\n", data);
}
```

Heapsort output

```
Original: CELKMSPXA
 Sorting: XMSKCLPEA, end=8
 Sorting: SMPKCLAEX, end=7
 Sorting: PMLKCEASX, end=6
 Sorting: MKLACEPSX, end=5
 Sorting: LKEACMPSX, end=4
 Sorting: KCEALMPSX, end=3
 Sorting: ECAKLMPSX, end=2
 Sorting: CAEKLMPSX, end=1
  Sorted: ACEKLMPSX
```

Analysis

- Heapsort has worst case and average performance of $O(n \log n)$.