CS 180 - Heaps

Announcements

- Midterms graded: lowest problem
- HW due tonight
- No lab tomorrow - lecture instead
- Office hours: tomorrow 11-12
- Boisy scholarships (104 Ritter)
Each node (except \( v \)) has a unique parent.

In CS, the relationship in a parent-child hierarchy is a tree. A tree is a set of nodes structured as a hierarchical tree.

Last time: Trees
- descendant/ancestor
- rooted subtree
- internal nodes
- leaves
- siblings
- ch. 12
- Wre dls
Complete: all levels filled

Full: exactly 2 children

Binary Tree.
\[ \text{height}(v) = \max \text{ (height of children)} \]

\[ \text{height} : \text{height}(\text{left}) = 0 \]

\[ \text{depth}(v) = \text{depth(\text{parent}(v))) + 1} \]

\[ \text{depth} : \text{depth}(r) = 0 \]

Depth + Height = defined recursively
Can be pointers or array based!

Nice trick
How big?

ergy:

Potential downside (of array)
Data Structure:

Priority Queue: supports the following operations

insert(e): adds element e to the data structure

removeMax(): removes maximum element

getMax(): returns maximum element

How to build?
Maintaining with list or vector:

How?

Repeated access is highest weight item in structure.

Good if you need limited

Why?
Vector: \( \frac{1}{2} \)

\[ \begin{pmatrix} 3 \\ 3 \\ \vdots \\ 3 \end{pmatrix} \]

Insert: \( 0 \cdot (\cdot) \)

Sort vector: \( \max \) at \( y \)

\[ \frac{1}{2} \]

Remove Max

Insert: \( \frac{1}{2} \)
(a): Get Max: find max new max
(b): remove Max: remove max from vector
(c): insert: put at end

Another (vector)

1 2 3 4 5 6 7 8 9
10
is scaled in left to right order

The tree is complete at level 0.

A binary tree where:

\[
\begin{align*}
\text{The level of node } v \text{ is parent of node } w \\
\text{if and only if } v \text{ is a leaf node and } w \text{ is not.}
\end{align*}
\]
Heap

Not a best

insert

getMax: return root
\[ \frac{1}{2} \cdot 3 \cdot 5 \cdot 7 \cdot \ldots \cdot n = 2^n - 1 \]
Next thing

the many Companies/Services

Running things
Amy. Because why?

Cool for this class.