CS 225

#27: Hashing Analysis

October 25, 2021 · *G Carl Evans*

Running Time:

Linear Probing:

- Successful: $\frac{1}{2}(1 + 1/(1-\alpha))$
- Unsuccessful: $\frac{1}{2}(1 + 1/(1-\alpha))^2$

Double Hashing:

- Successful: $1/\alpha * \ln(1/(1-\alpha))$
- Unsuccessful: $1/(1-\alpha)$

Separate Chaining:

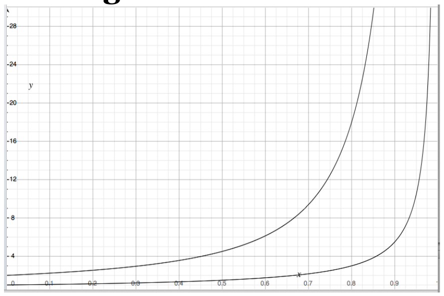
- Successful: $1 + \alpha/2$
- Unsuccessful: $1 + \alpha$

Running Time Observations:

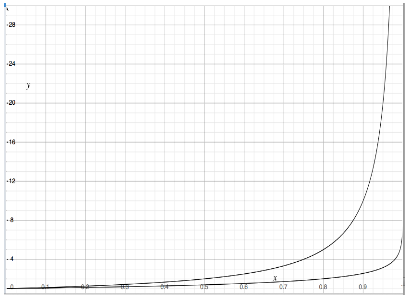
1. As α increases:

2. If α is held constant:

Running Time Observations:



Linear Probing:
Successful: $\frac{1}{2}(1 + 1/(1-\alpha))$
Unsuccessful: $\frac{1}{2}(1 + 1/(1-\alpha))^2$



Double Hashing:
Successful: $1/\alpha * \ln(1/(1-\alpha))$
Unsuccessful: $1/(1-\alpha)$

ReHashing:

What happens when the array fills?

Algorithm:

Which collision resolution strategy is better?

- Big Records:

- Structure Speed:

What structure do hash tables replace?

What constraint exists on hashing that doesn't exist with BSTs?

Why talk about BSTs at all?

Analysis of Dictionary-based Data Structures

	Hash Table		AVL	List
	SUHA	Worst Case		
Find				
Insert				
Storage Space				

A Secret, Mystery Data Structure:

ADT:
insert

remove

isEmpty

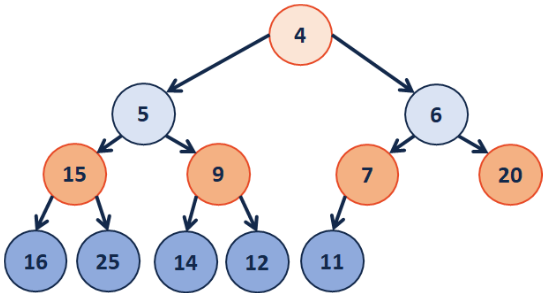
Implementation of ________________

insert	removeMin	Implementation
O(n)	O(n)	Unsorted Array
O(1)	O(n)	Unsorted List
O(lg(n))	O(1)	Sorted Array
O(lg(n))	O(1)	Sorted List

Q1: What errors exist in this table? (Fix them!)

Q2: Which algorithm would we use?

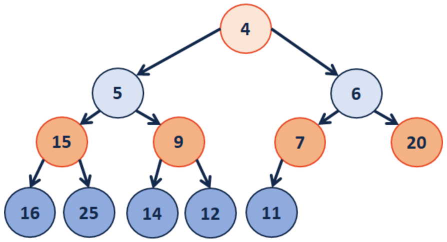
A New Tree-like Structure:



A complete binary tree T is a min-heap if:

-
-

Implementing a (min)Heap as an Array



4	5	6	15	9	7	20	16	25	14	12	11			

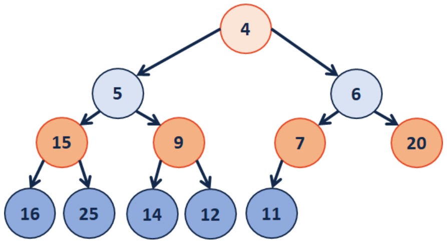
Operations:

leftChild(index) :=

rightChild(index) :=

parent(index) :=

Insert:



–	4	5	6	15	9	7	20	16	25	14	12	11			

CS 225 – Things To Be Doing:

1. Exam 2 starts tomorrow.
2. mp_mosaics EC deadline is today – earn the extra credit!
3. lab_hash released Thursday
4. Daily POTDs are ongoing!