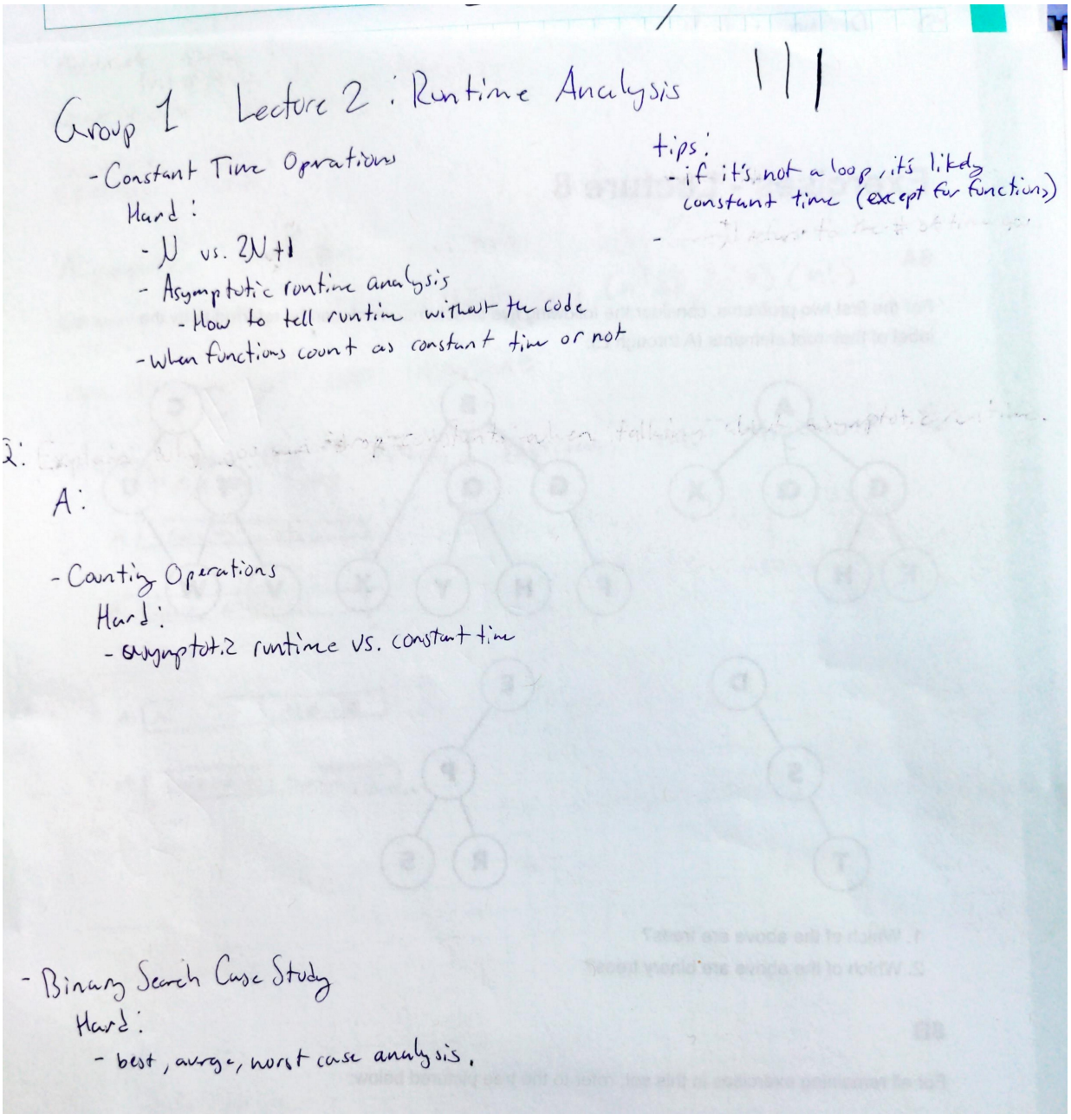
Review notes

Friday, February 10, 2023 4:29 PM



Group 1 Lecture 2: Runtime Analysis 111

- Constant Time Operations

Hard:

- N vs. 2N+1
- Asymptotic runtime analysis
  - How to tell runtime without the code
- When functions count as constant time or not

tips:

- if it's not a loop, it's likely constant time (except for functions)
-

2:

A:

- Counting Operations

Hard:

- Asymptotic runtime vs. constant time

- Binary Search Case Study

Hard:

- best, average, worst case analysis.

Abstract Data types
Interface vs Implementation
what it does vs how it's done

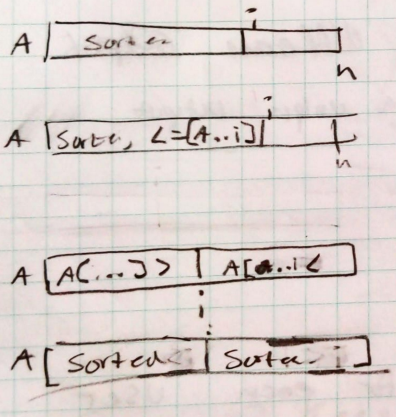
Insertion Sort
Selection Sort

Asymptotic runtime analysis Big O

Common classes O(log n), (1), (n log n), (n^2), (2^n), (n!)

Box diagram of loop invariant

Which of the following represents the invariant for Insertion Sort



5) 1) No, invalid

2) not really, bc an output has no tangible connection to the input

3) No, its already bad

2) integers

3)

1) valid, an input will produce a different output

2) valid, unique input will have a unique output

c) return whom) accuracy unique for each user

Group 3 → Lecture 4

Recursion

It's Easy!

Basics of Recursion

→ 1) precise specification
2) Base case
3) recursive call makes progress on subproblem(s)
4) replace recursive call with spec and verify overall behavior

Tips → Keep it simple
→ Don't try and write out each call, just use the 4 steps!

Moses Miller 5 Anuk Centellas Wyatt Pratch Carver Rasmussen

- Topics:
  - Merge sort
  - Loop invariants II
  - Divide & Conquer

- Hard Things:
  - Recursively creating subarrays I
  - Run time analysis I
  - Divide & Conquer

- Tips:
  - The precondition for merge is that the arrays are already sorted
  - Review A1 (if done correctly...)

- Potential Questions:
  - Execute merge on this array:

[1, 4, 5, 7] [3, 6, 8, 9]

Answer: [1, 3, 4, 5, 6, 7, 8, 9]

- Do the subarrays need to be the same size?

Answer: No!

Lecture 6

Topics - Quicksort

Hard things - Pivot not defined
- Can impact runtime

- Determining stability is tricky

- Conceptualizing mental partitions can be difficult to keep track of

Tips - Pivot should not be first or last

- Picture cards when running through QuickSort

Q's - What are the best, average, and worst-case runtimes for Quicksort?
- $n \log n$, $n \log n$, n^2

- When does the worst-case runtime occur?
- when pivot is on a minimum/maximum

Radix Sort #7

- Bucket Sort ; Counting Sort

Hard Things
- Counting sort ; Bucket sort both traverse backwards
- pseudo code

Tips
- Stably sorted
- Sort from lowest to highest digit

Potential questions

Q: What is the best possible worst case runtime for a comparison-based sorting algorithm?

A: O(Nlogn)

Q; What is the main difference between counting and bucket sort?

A: Bucket sort organizes numbers via digits Counting sort organizes numbers by summing the count.

Lecture 8
- Trees, Traversal, Recursion on Trees

Hard things
- Tree Traversal ✓✓✓✓

Tips
- Pre order Parent is "Pre" PLR
- Post order Parent is "Post" LRP
- In order Parent is "in" between LPR

Write a base case for Tree traversal:

```
If (t==null){
 return false };
```

What is the height of a tree?

Insert elements into a tree in the order they are visited in Pre Order Traversal

LECTURE 10

Topics: Generics, comparable interfaces, balance factors, & tree rotations

Hard things: Rotations - updating pointers when coding

Tips: When calculating balance factor, make sure to subtract the height of the node's children, not right & left subtrees.

Generics: a specialized class that is constructed to ignore types ~~to be~~ to be able to be used across the board.

Questions:

✓

Lecture 11 - AVL Trees

11/11

tips

AVL invariant: -1 <= balance(n) <= 1

worst case for AVL insertion, search + removal is O(log n)

know 2 cases for rebalancing

Hard Things

- Rebalancing
  - knowing which rotations are needed
- Time Complexity

Lecture 12

Topics

- Priority Queue: Queue sorted by priority
- Heaps:
  - Implements Priority Queue
  - Each child node is $\leq$ or $\geq$ than it's parent depending on if it's a min- or max- heap
  - No holes
  - Stored as an array

Hard things

- Inserting nodes/values

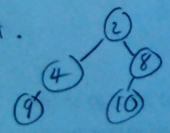
Tips

- Make sure to remember the formulae to find the parent node and child node

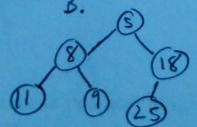
Potential Questions

Which of the trees are valid min heaps?

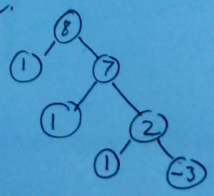
A.



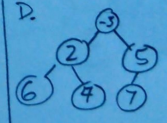
B.



C.



D.



Answer: B, D

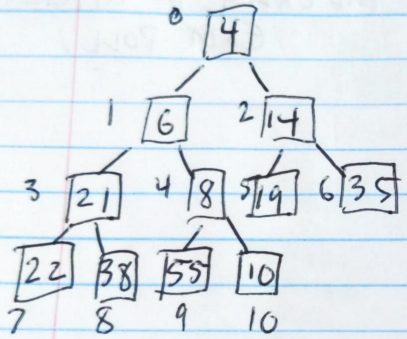
TYAHZN ORDAHL
ROUSHA GOTHAM
COLLIN SARGENT
FELIX HOMAR

GROUP 12
LECTURE 13

HEAP SORT: (AKA: PRIORITY QUEUES)

- poll()
- level order w/ no holes
- ~~O(n) construction~~ don't need to know

LEVEL ORDER TIPS:



TO FIND NODE K'S PARENT:

$(K-1)/2$

TO FIND NODE K'S CHILDREN:

$2K+1$ & $2K+2$

$2K+1$ → LEFT CHILD

$2K+2$ → RIGHT CHILD

poll() TIPS:

Hard Things

O(n) construction

13

HEAP SORT ?:

YOU HAVE A HEAP SORT THAT FAVORS THE MINIMUM VALUE. WRITE THE OUTPUT FOR THE FOLLOWING CALLS. WRITE THE FINAL TREE @ THE END OF THE CALLS: & AFTER EACH POLL.

```
add(12);
add(8);
add(13);
peek();
poll();
add(4);
peek();
add(16);
poll();
add(1);
```