CS180 Discussion

Week 1

Office Hours

Monday: 8:00 - 10:00

Outside 391 ENG VI

Lecture Recap

- Celebrity problem
- Stable Matching
- Big O Small Ω
- Amortized analysis
- Von Neumann architecture
- ~~Interval scheduling~~
- ~~Majority calculation algorithm~~

Celebrity Problem

A party of N people, only one person is known to everyone. Such a person may be present in the party, if yes, (s)he doesn't know anyone in the party. We can only ask questions like "does A know B? ". Find the stranger (celebrity) in minimum number of questions.

Stable Matching

set of two men, {m, m′}, and a set of two women, {w, w′}. The preference lists are as follows:

m prefers w to w′.

m′ prefers w′ to w.

w prefers m′ to m.

w′ prefers m to m′.

(i) M prefers his wife to another woman W; or

(ii) W prefers her current husband over another man M.

Gale-Shapley algorithm

Initially all $m \in M$ and $w \in W$ are free
While there is a man m who is free and hasn't proposed to every woman
a. Choose such a man m
b. Let w be the highest-ranked woman in m's preference list
c. to whom m has not yet proposed;
d. If w is free then
    i. (m, w) become engaged
e. Else
    w is currently engaged to m'
    i. If w prefers m' to m then
        1. m remains free
    ii. Else
        1. w prefers m to m' (m, w) become engaged m' becomes free
f. Endif
    i. Endif
Endwhile
Return the set S of engaged pairs

Gale-Shapley algorithm

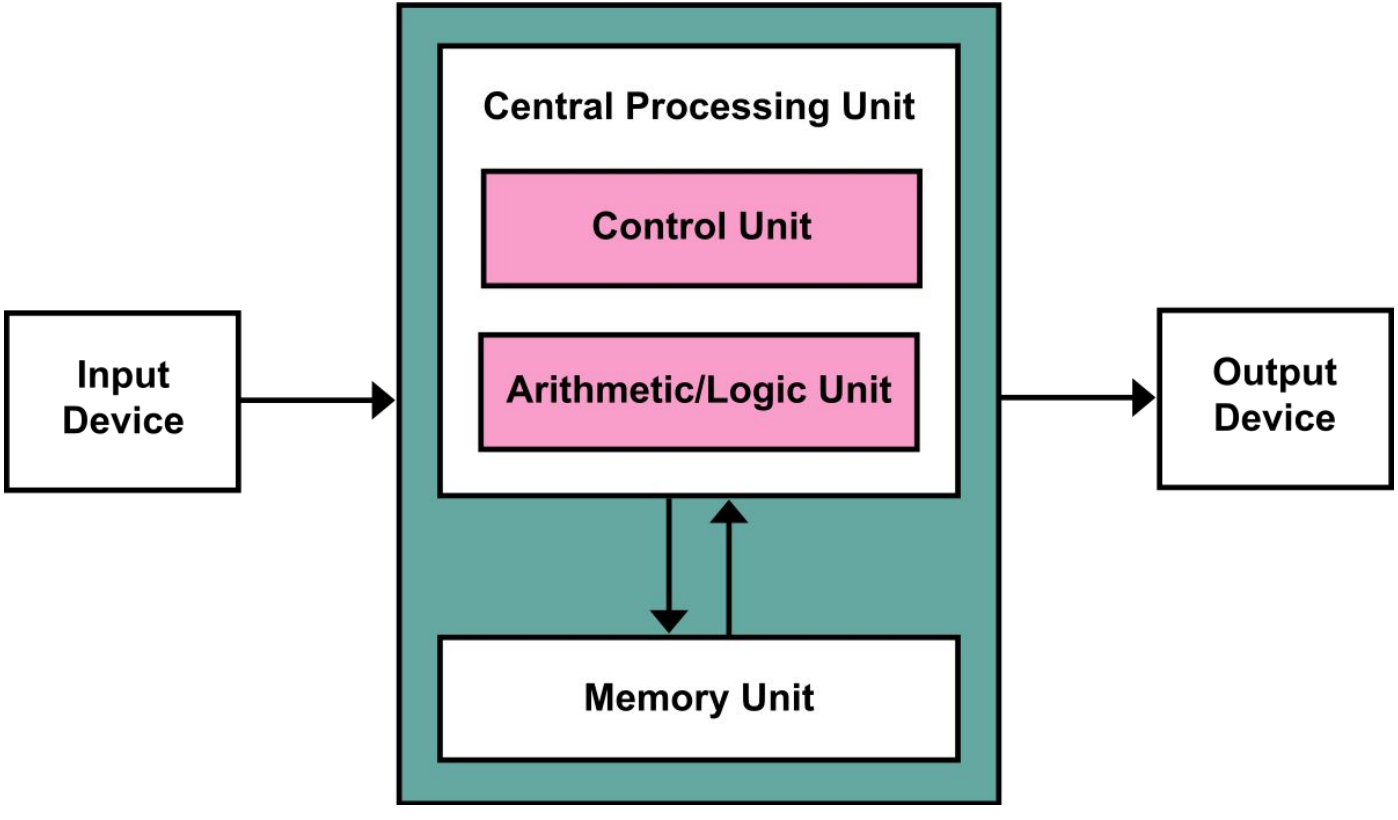
Initially all $m \in M$ and $w \in W$ are free
While there is a man m who is free and hasn't proposed to every woman
  a. Choose such a man m
  b. Let w be the highest-ranked woman in m's preference list
  c. to whom m has not yet proposed;
  d. If w is free then
    i. (m, w) become engaged
  e. Else
    i. w is currently engaged to m'
    ii. If w prefers m' to m then
      1. m remains free
    iii. Else
      1. w prefers m to m'(m, w) become engaged m' becomes free
  f. Endif
    i. Endif
Endwhile
Return the set S of engaged pairs

$O(n^2)$

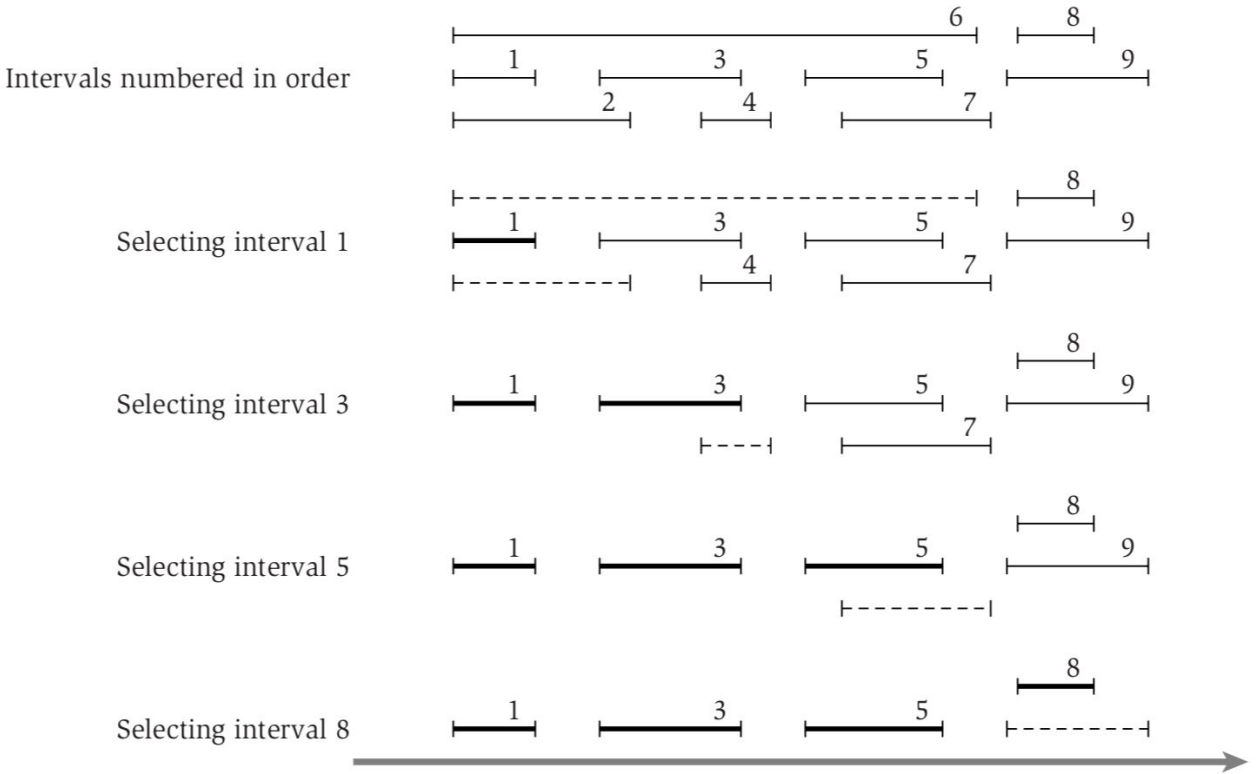
PAIRS:

(m,w), (m', w')

Von Neumann architecture

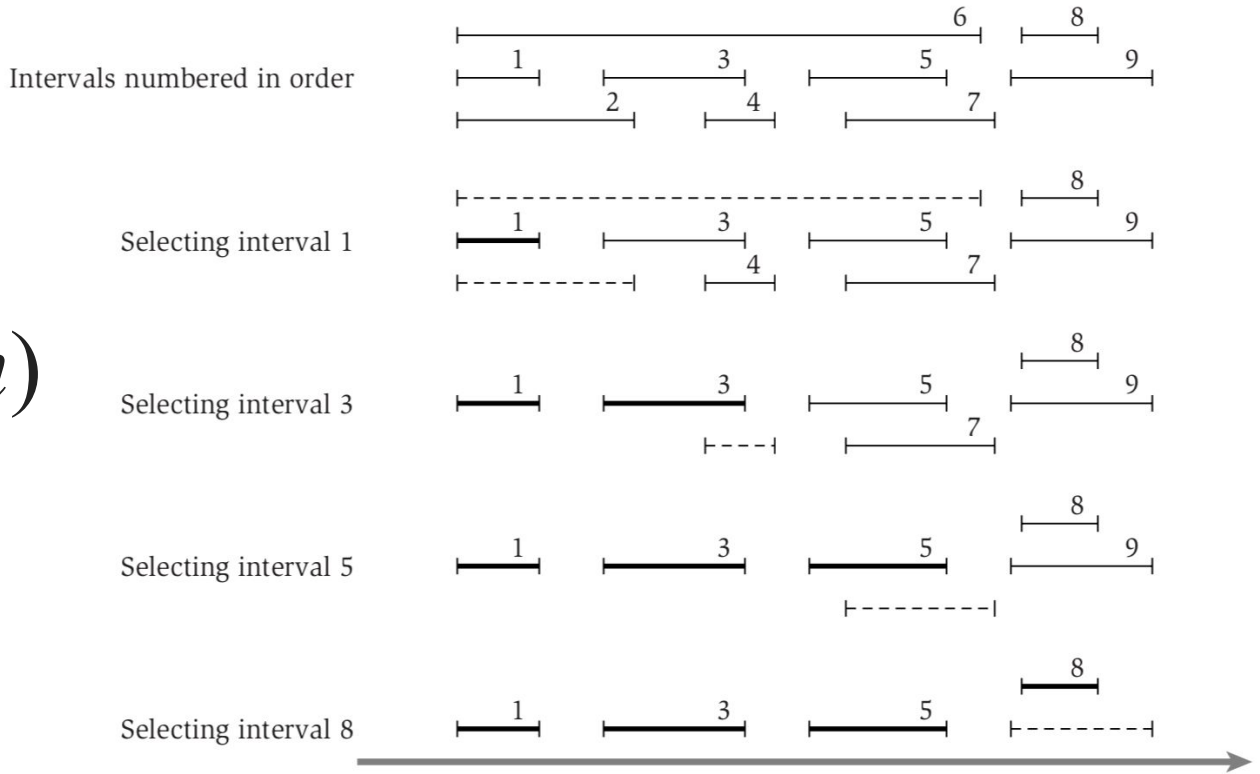


Interval scheduling

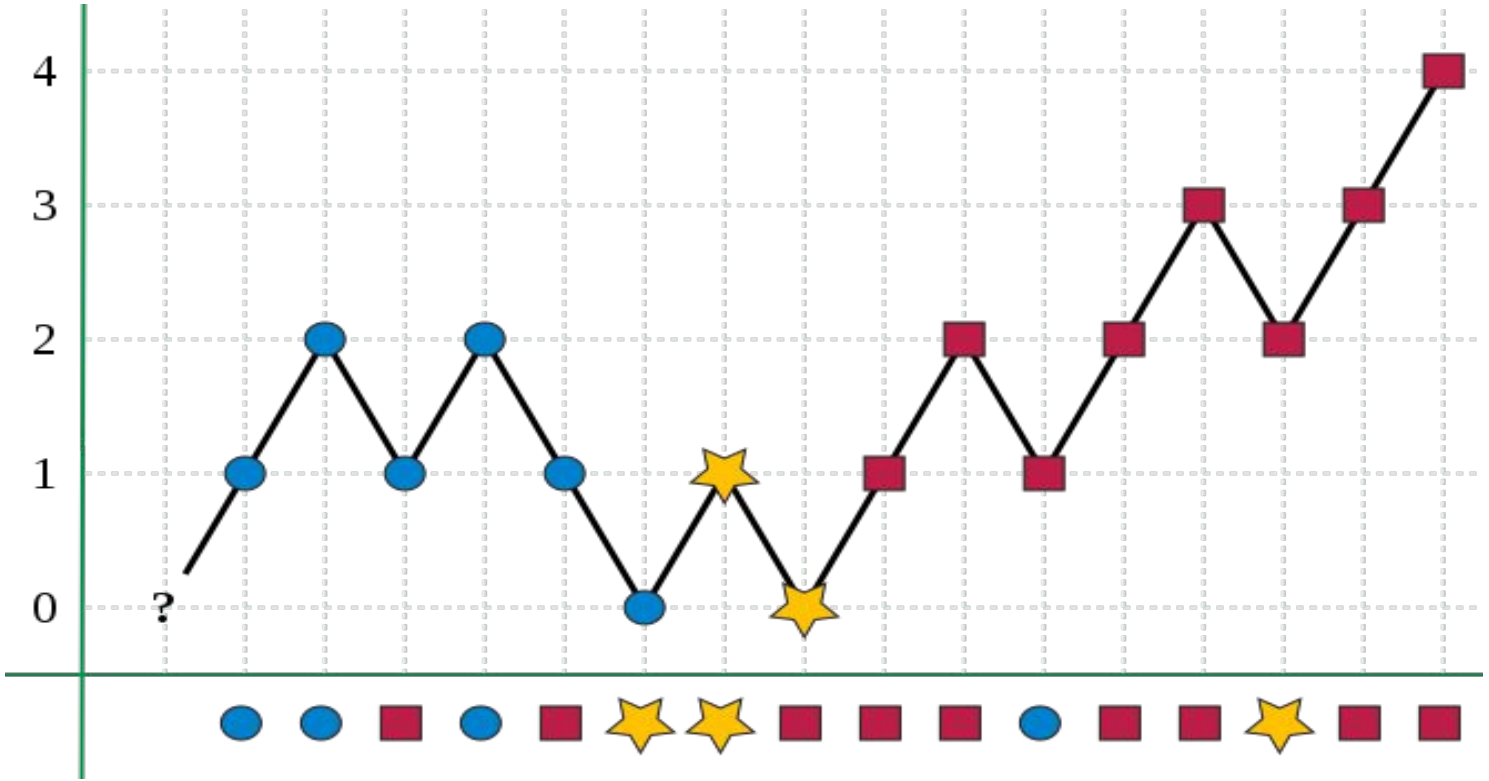


Interval scheduling

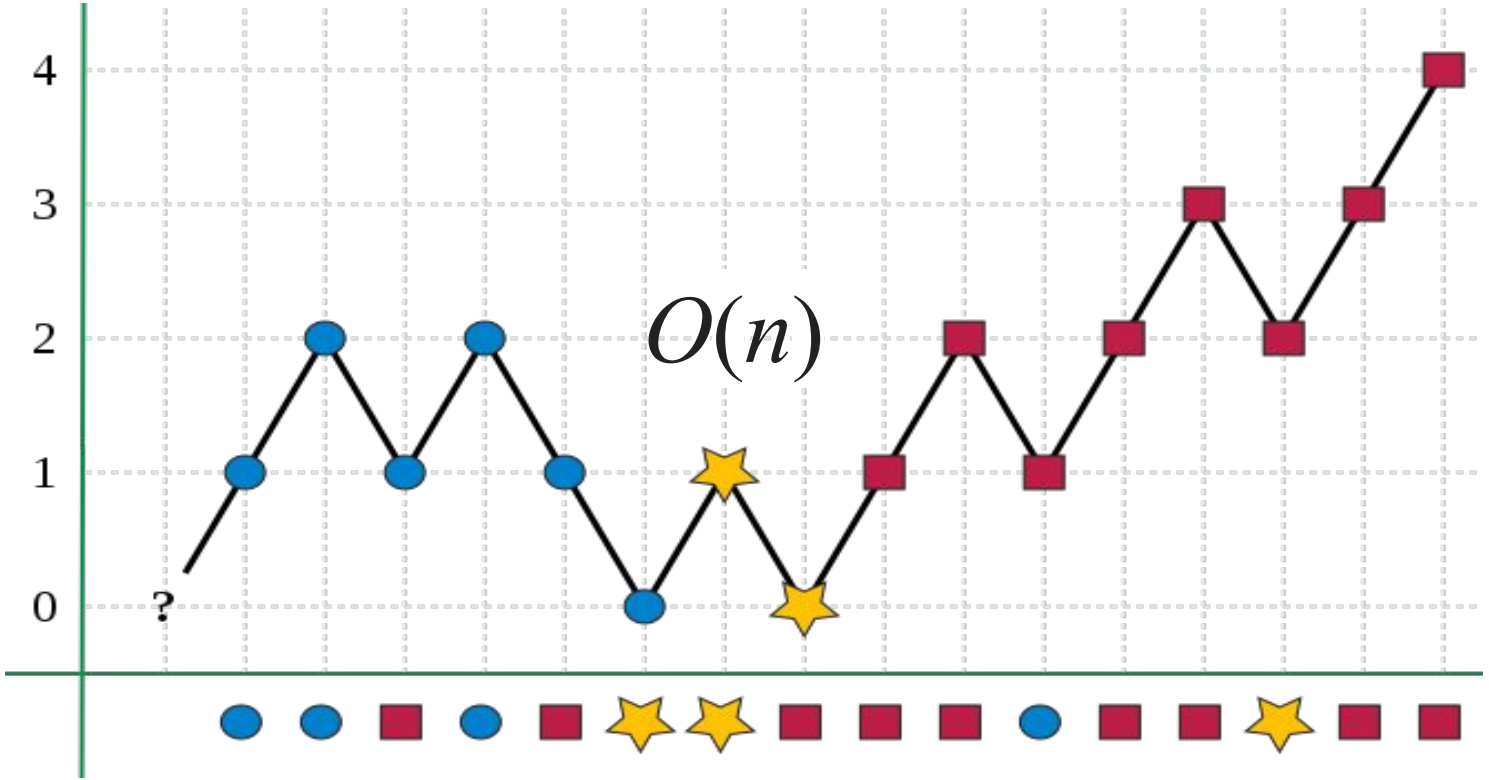
$O(n \log n)$



Calculating majority



Calculating majority



Induction

- *Start with P*(*n*)
- **Base case**: Show that the statement holds for *n* = 0 or n = 1.
- **Assumption** Assume that *P*(*k*) holds
- **Inductive step**: Show that *if P*(*k*) holds, then also *P*(*k* + 1) holds

terview Questions!!!

Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

Palindrome Permutation

Given a string, write a function to check if it is a permutation of a palindrome. A palindrome is a word or phrase that is the same forwards and backwards. A permutation is a rearrangement of letters. The palindrome does not need to be limited to just dictionary words.

EXAMPLE

Input: Tact Coa

Output: True (permutations: "taco cat", "atco cta"; etc.)

Palindrome Permutation

- Count each character (in a hash map)
- Check that each character has an even count
- Exactly one character can be odd

Smallest Difference

Given two arrays of integers, compute the pair of values (one value in each array) with the smallest (non-negative) difference. Return the difference.

EXAMPLE

Input: {1, 3, 15, 11, 2}, {23, 127, 235, 19, 8}

Output: 3. That is, the pair (11, 8).

Smallest Difference

Input: {1, 3, 15, 11, 2}, {23, 127, 235, 19, 8}

Sort the input:

{1, 2, 3, 11, 15}, {8, 19, 23, 127, 235}

Build Order

Build Order: You are given a list of projects and a list of dependencies (which is a list of pairs of projects, where the second project is dependent on the first project). All of a project's dependencies must be built before the project is. Find a build order that will allow the projects to be built. If there is no valid build order, return an error.

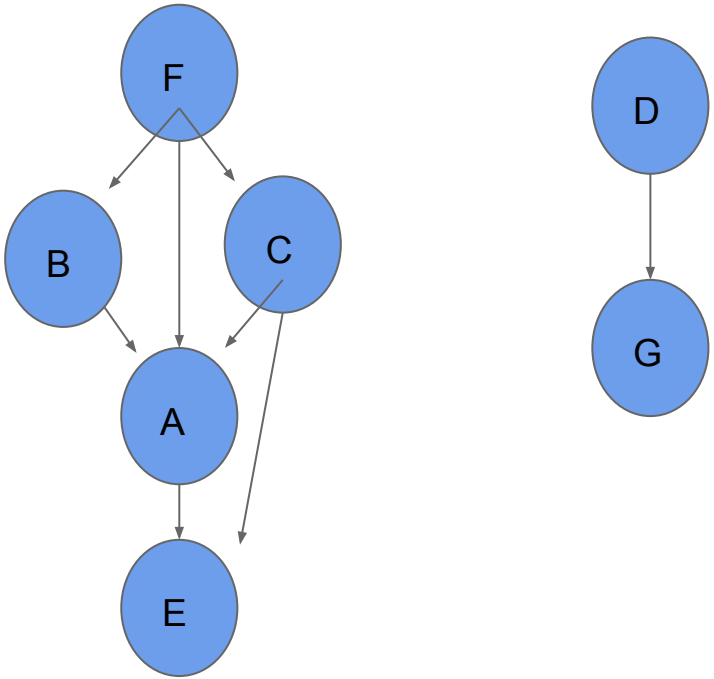
EXAMPLE

Input:

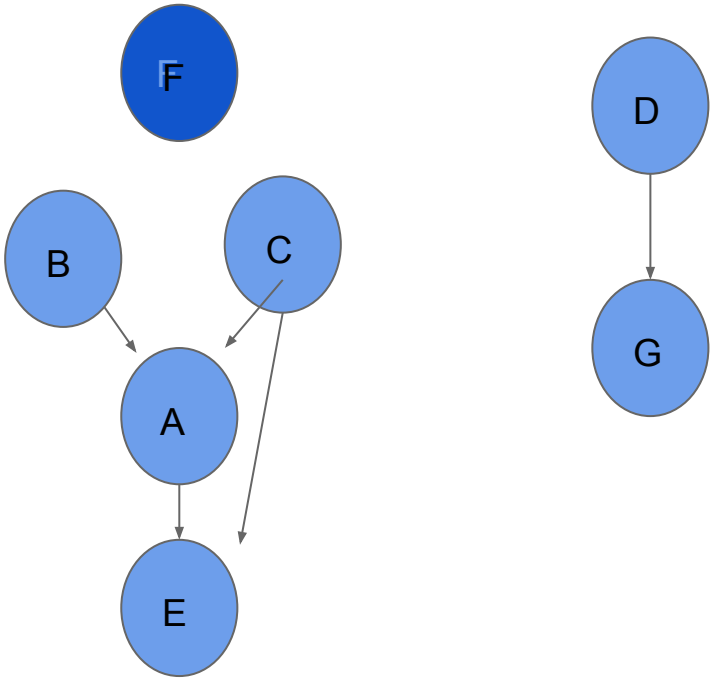
projects: a, b, c, d, e, f

dependencies: (a, e), (f, b), (b, a), (f, c), (d, g) (c, a) (f,a) Output: f, d, c, b, a, g, e

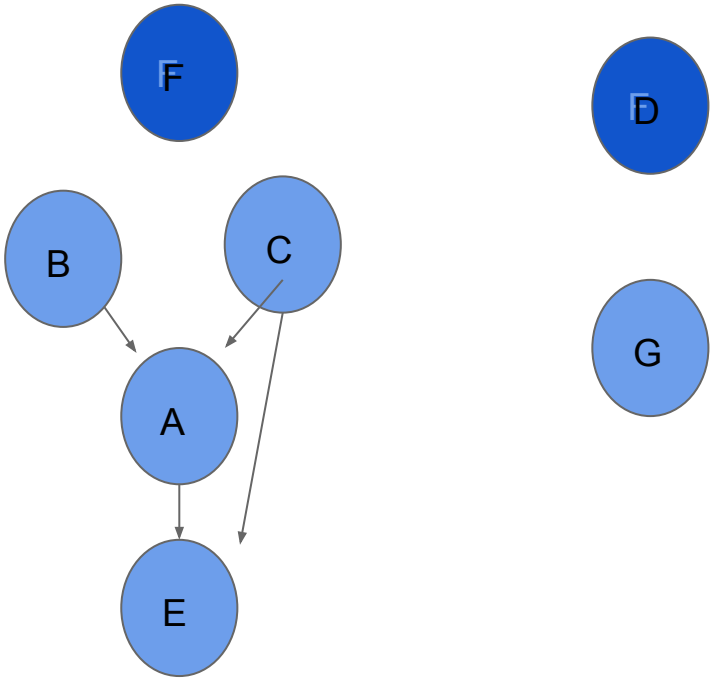
Build dependencies



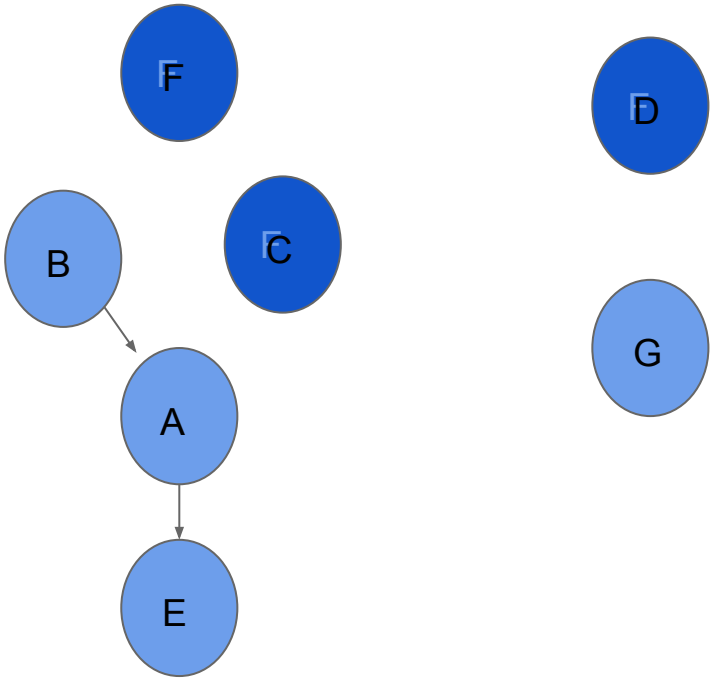
Build dependencies



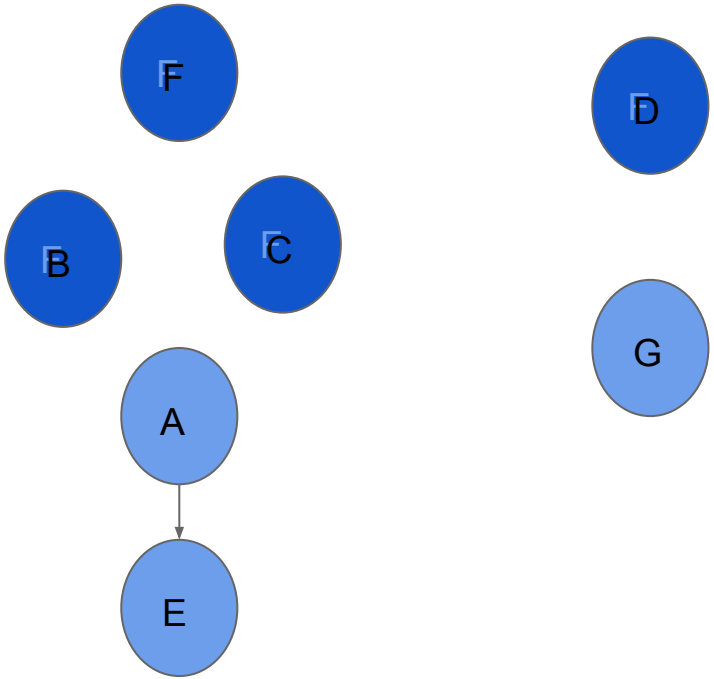
Build dependencies



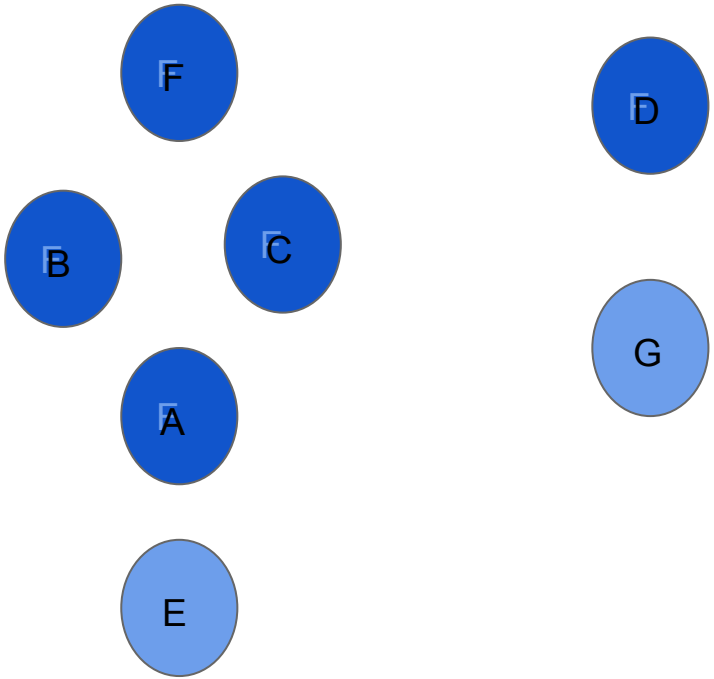
Build dependencies



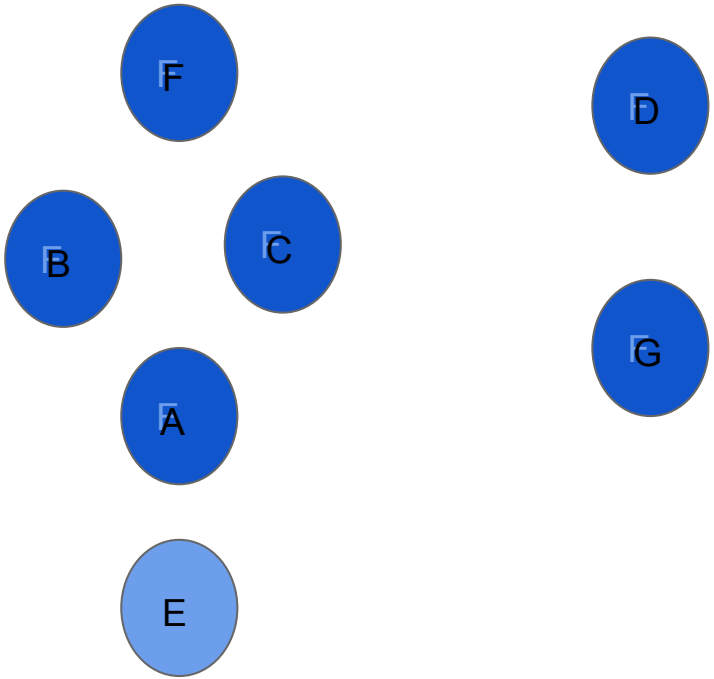
Build dependencies



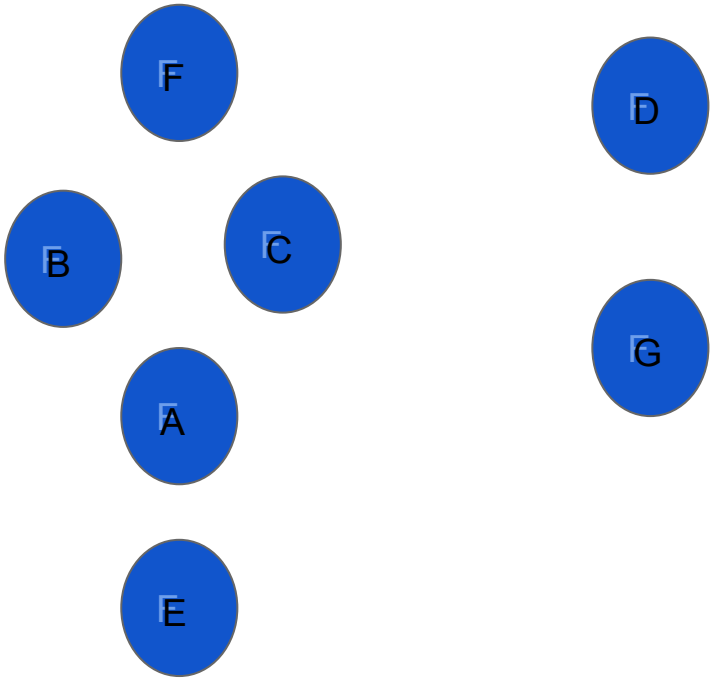
Build dependencies



Build dependencies

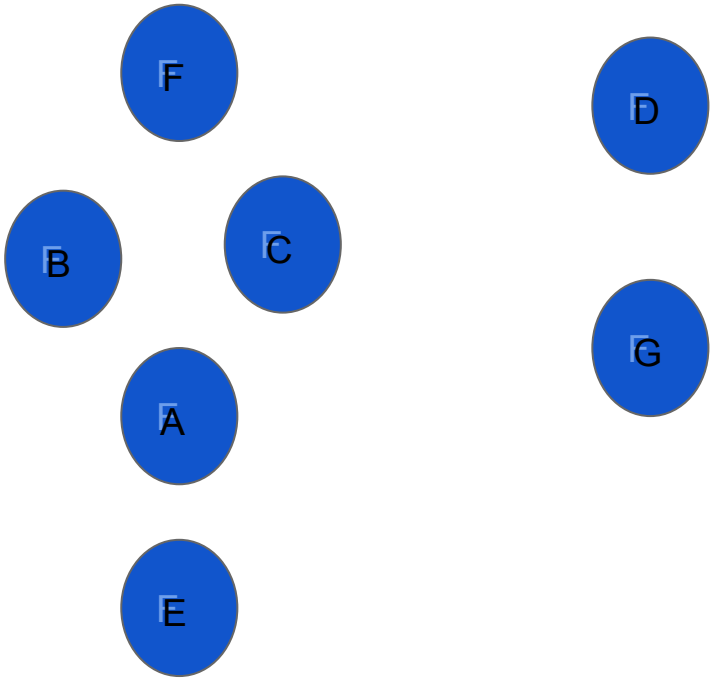


Build dependencies



Build dependencies

O(P + D)



CS180 Discussion

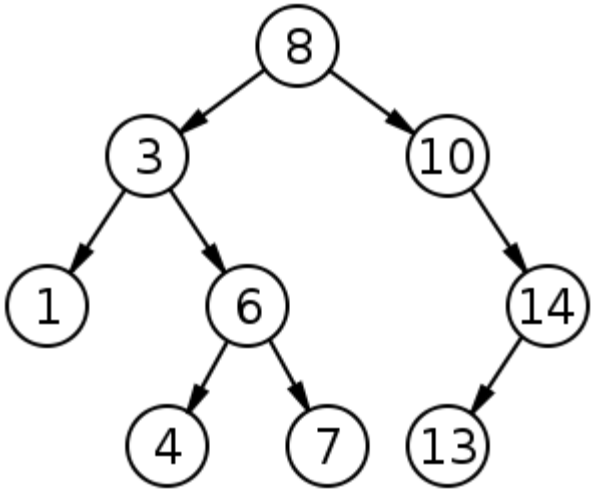
Week 2

Lecture Recap

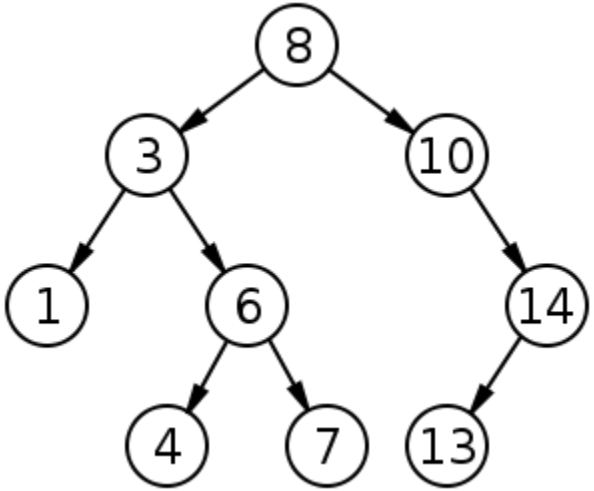
- Greedy algorithm
- Interval scheduling
- Intro to graphs
- BFS
- DFS
- ~~Graph coloring~~
- ~~Topological ordering~~

Binary Search Tree

A binary search tree is a rooted binary tree, whose internal nodes each store a key and each have two distinguished sub-trees, commonly denoted left and right. The tree additionally satisfies the binary search property, which states that the key in each node must be greater than or equal to any key stored in the left subtree, and less than or equal to any key stored in the right subtree

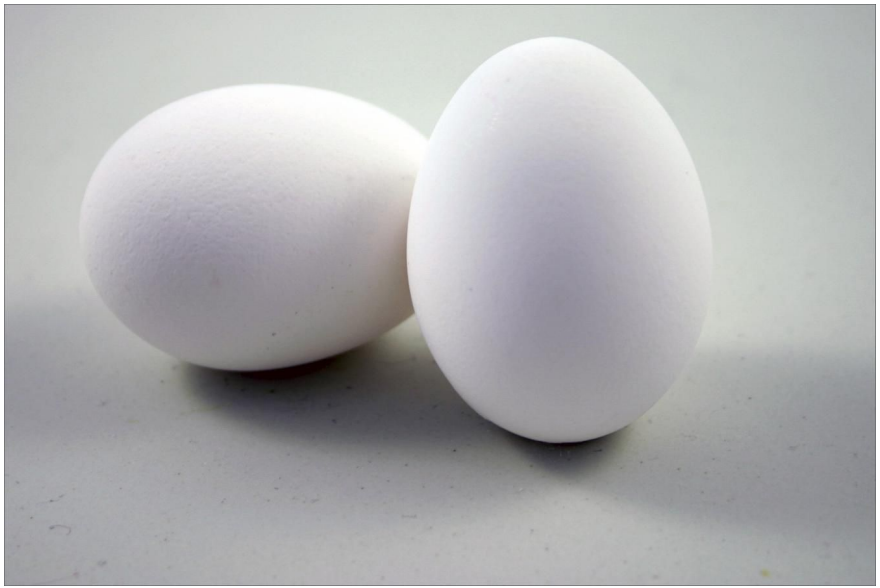


Question



HW 1 Questions

Minimization of Maximum Regret



Hospitals

There were m hospitals, each with a certain number of available positions for hiring residents. There were n medical students graduating in a given year, each interested in joining one of the hospitals. Each hospital had a ranking of the students in order of preference, and each student had a ranking of the hospitals in order of preference. We will assume that there were more students graduating than there were slots available in the m hospitals.

Ships

Given the schedule for each ship, find a truncation of each so that the condition continues to hold: no two ships are ever in the same port on the same day. Show that such a set of truncations can always be found, and give an algorithm to find them.

Example. Suppose we have two ships and two ports, and the “month” has four days. Suppose the first ship’s schedule is:
port P1; at sea; port P2; at sea
and the second ship’s schedule is:
at sea; port P1; at sea; port P2

TV Schedule

Suppose we have two television networks, whom we'll call A and B. There are n prime-time programming slots, and each network has n TV shows. Each network wants to devise a schedule—an assignment of each show to a distinct slot—so as to attract as much market share as possible.

Resolve this question by doing one of the following two things:

(a) give an algorithm that, for any set of TV shows and associated ratings, produces a stable pair of schedules; or

(b) give an example of a set of TV shows and associated ratings for which there is no stable pair of schedules.

Interview Questions!!!

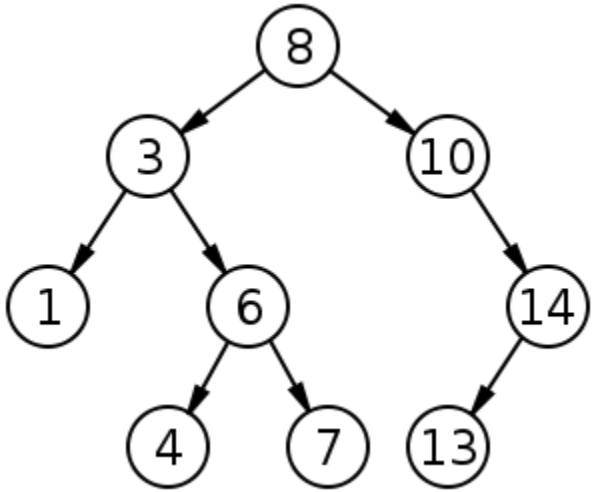
Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

Validate BST

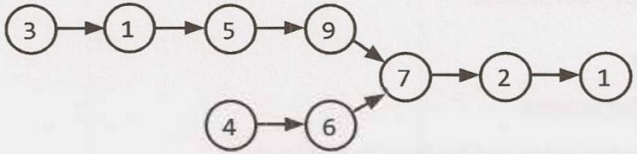
Implement a function to check if a binary tree is a binary search tree.



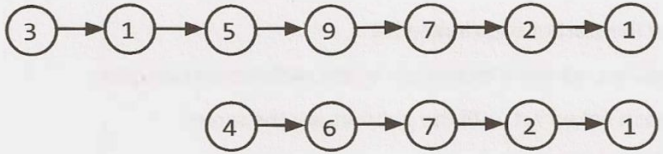
Intersection

intersection: Given two (singly) linked lists, determine if the two lists intersect. Return the intersecting node. Note that the intersection is defined based on reference, not value. That is, if the kth node of the first linked list is the exact same node (by reference) as the jth node of the second linked list, then they are intersecting.

Here is a picture of intersecting linked lists:



And here is a picture of non-intersecting linked lists:



Detecting Loops

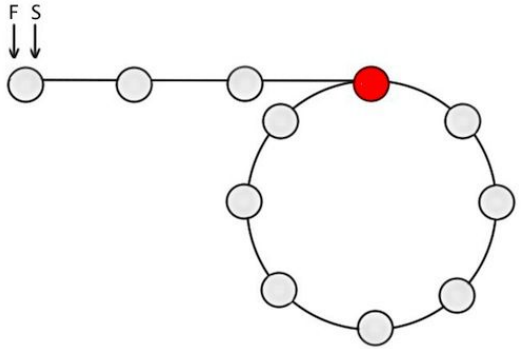
Loop Detection: Given a circular linked list, implement an algorithm that returns the node at the beginning of the loop.

DEFINITION:

Circular linked list: A (corrupt) linked list in which a node's next pointer points to an earlier node, so as to make a loop in the linked list.

EXAMPLE

Input: A -> B -> C -> D -> E -> C [the same C as earlier] Output: C



we move FastPointer twice as fast as SlowPointer. When SlowPointer enters the loop, after k nodes, FastPointer is k nodes into the loop. This means that FastPointer and SlowPointer are LOOP_SIZE - k nodes away from each other.
Next, if FastPointer moves two nodes for each node that SlowPointer moves, they move one node closer to each other on each turn. Therefore, they will meet after LOOP_SIZE - k turns. Both will be k nodes from the front of the loop.
The head of the linked list is also k nodes from the front of the loop. So, if we keep one pointer where it is, and move the other pointer to the head of the linked list, then they will meet at the front of the loop.

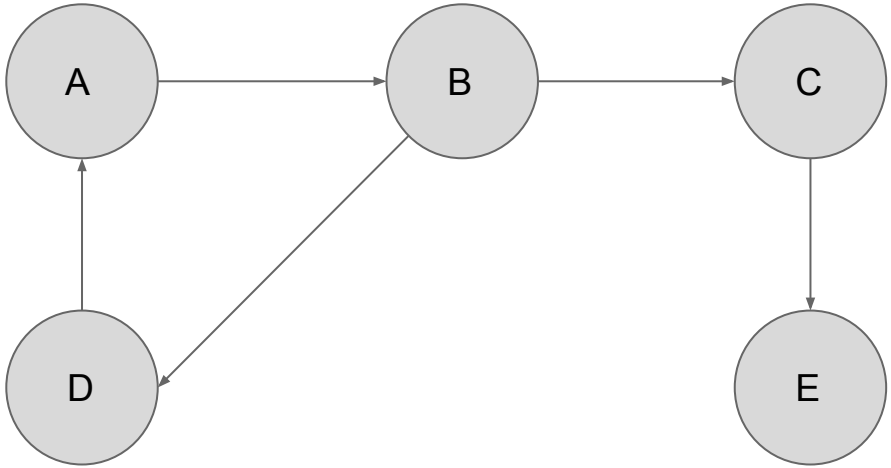
CS180 Discussion

Week 3

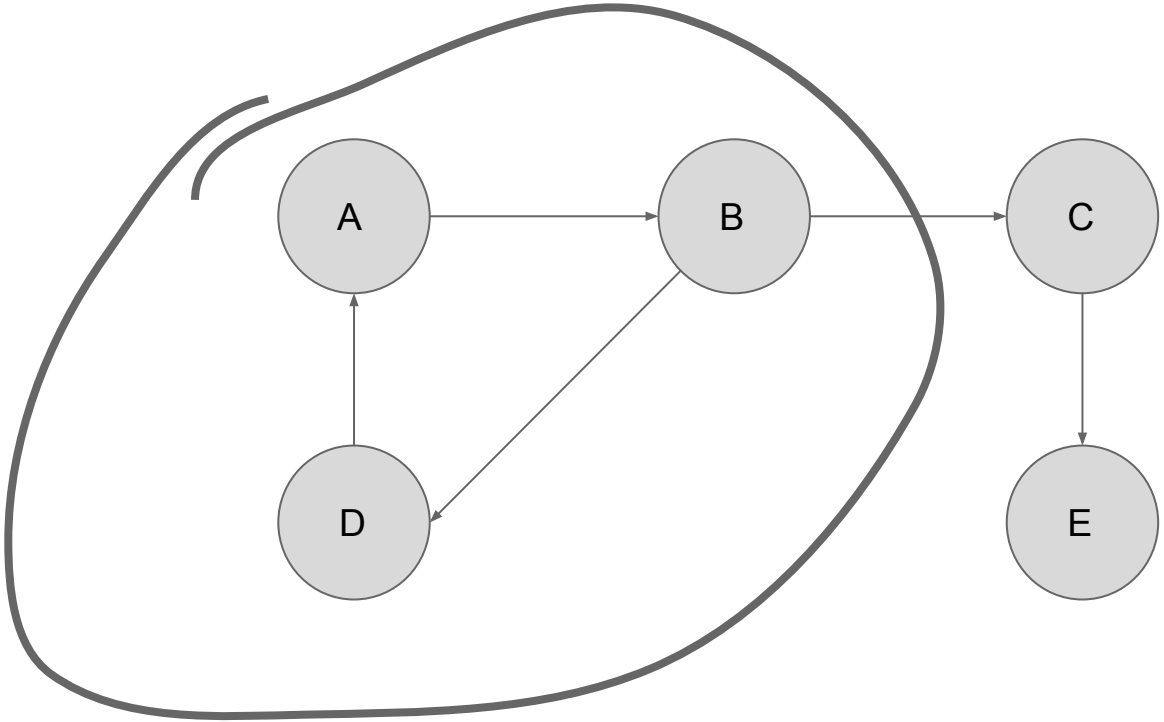
Lecture Recap

- Strongly connected component
- Graph Coloring Problem
- Topological sort
- Introduction to greedy algorithms
- Dijkstra: shortest path algorithm

Strongly connected (?)



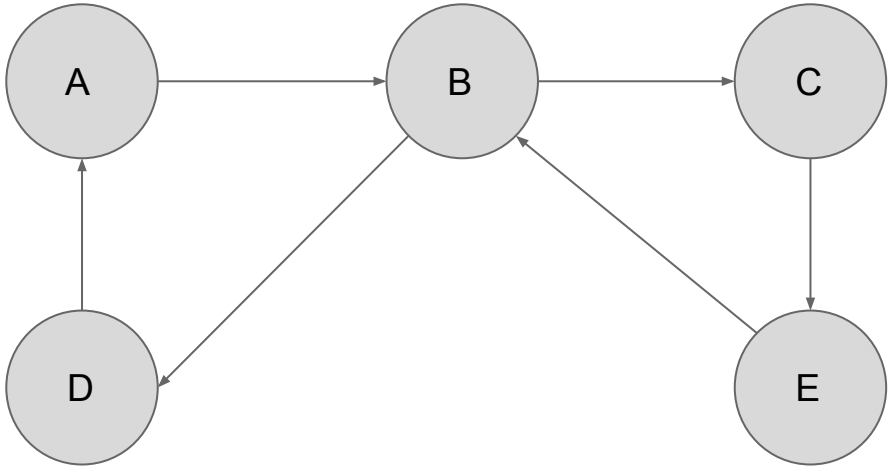
Strongly connected



Kosaraju's algorithm

L = empty list
For each vertex u of the graph do Visit(u), where Visit(u) is the recursive subroutine:
    If u is unvisited then:
        Mark u as visited.
        For each out-neighbour v of u, do Visit(v).
        Prepend u to L.
    Otherwise do nothing.
For each element u of L in order, do Assign(u,u) where Assign(u,root) is the recursive subroutine:
    If u has not been assigned to a component then:
        Assign u as belonging to the component whose root is root.
        For each in-neighbour v of u, do Assign(v,root).
    Otherwise do nothing.

Eulerian path



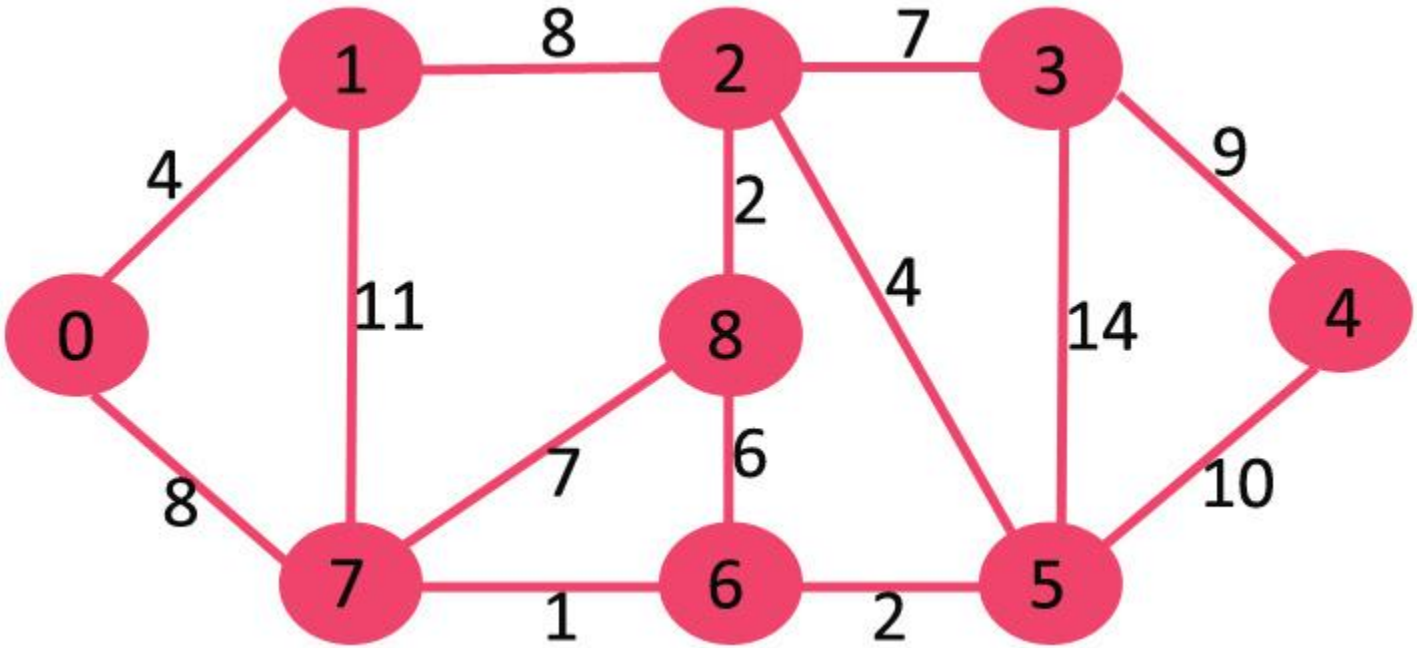
Eulerian path

A directed graph has an eulerian cycle if following conditions are true

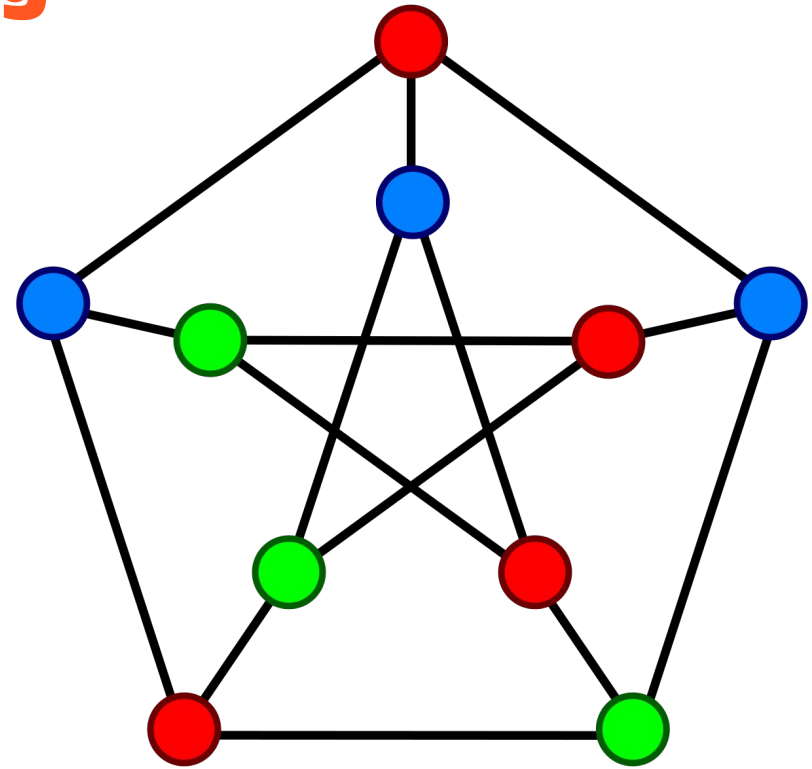
1) All vertices with nonzero degree belong to a single strongly connected component.

2) In degree and out degree of every vertex is same.

Greedy Algorithms



Graph Coloring



HW 2 Questions

Butterflies

we'll declare the m judgments to be consistent if it is possible to label each specimen either A or B in such a way that for each pair (i, j) labeled "same," it is the case that i and j have the same label; and for each pair (i, j) labeled "different," it is the case that i and j have different labels.

Give an algorithm with running time O(m + n) that determines whether the m judgments are consistent.

Infection sequence

Design an algorithm that answers questions of this type: given a collection of trace data, the algorithm should decide whether a virus introduced at computer Ca at time x could have infected computer Cb by time y. The algorithm should run in time O(m + n).

(C1,C2,4), (C2,C4,8), (C3,C4,8), (C1,C4,12),

Dead People

A set of n people (all of them now deceased), whom we'll denote P1, P2, . . . , Pn. Suggested facts about when these people lived relative to one another. Each fact has one of the following two forms:

. For some i and j, person Pi died before person Pj was born; or

. For some i and j, the life spans of Pi and Pj overlapped at least partially.

Give an efficient algorithm to do this: either it should produce proposed dates of birth and death for each of the n people so that all the facts hold true, or it should report (correctly) that no such dates can exist—that is, the facts collected by the ethnographers are not internally consistent

Path destroyer

Suppose that an n-node undirected graph G = (V , E) contains two nodes s and t such that the distance between s and t is strictly greater than n/2. Show that there must exist some node v, not equal to either s or t, such that deleting v from G destroys all s-t paths.

Numbers question

An array of n elements contains all but one of the integers from 1 to n+1.

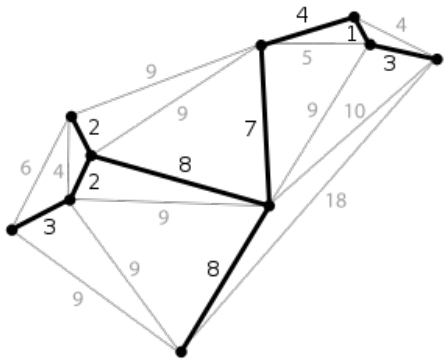
(a) Give the best algorithm you can for determining which number is missing if the array is sorted, and analyze its asymptotic worst-case running time.

(b) Give the best algorithm you can for determining which number is missing if the array is not sorted, and analyze its asymptotic worst- case running time.

True or False

a. Suppose we are given an instance of the Minimum Spanning Tree Problem on a graph G, with edge costs that are all positive and distinct. Let T be a minimum spanning tree for this instance. Now suppose we replace each edge cost ce by its square, ce2, thereby creating a new instance of the problem with the same graph but different costs.

   True or false? T must still be a minimum spanning tree for this new instance.



True or False

b. Suppose we are given an instance of the Shortest s-t Path Problem on a directed graph G. We assume that all edge costs are positive and distinct. Let P be a minimum-cost s-t path for this instance. Now suppose we replace each edge cost ce by its square, ce2, thereby creating a new instance of the problem with the same graph but different costs.

True or false? P must still be a minimum-cost s-t path for this new instance.

Subsequence detection

Give an algorithm that takes two sequences of events—S′ of length m and S of length n, each possibly containing an event more than once—and decides in time O(m + n) whether S′ is a subsequence of S.

buy Yahoo, buy eBay, buy Yahoo, buy Oracle

buy Amazon, buy Yahoo, buy eBay, buy Yahoo, buy Yahoo, buy Oracle

El Goog

distinct jobs, labeled J1, J2, . . . , Jn

Each job consists of two stages: first it needs to be preprocessed on the supercomputer, and then it needs to be finished on one of the PCs. Let's say that job Ji needs pi seconds of time on the supercomputer, followed by fi seconds of time on a PC. 1 supercomputer, n PCs

Give a polynomial-time algorithm that finds a schedule with as small a completion time as possible.

Interview Questions!!!

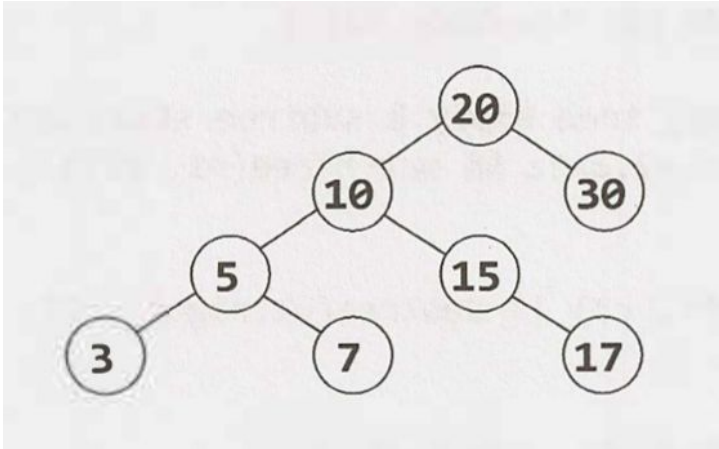
Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

getRandomNode

Random Node: You are implementing a binary search tree class from scratch, which, in addition to insert, find, and delete, has a method getRandomNode() which returns a random node from the tree. All nodes should be equally likely to be chosen. Design and implement an algorithm for getRandomNode(), and explain how you would implement the rest of the methods.



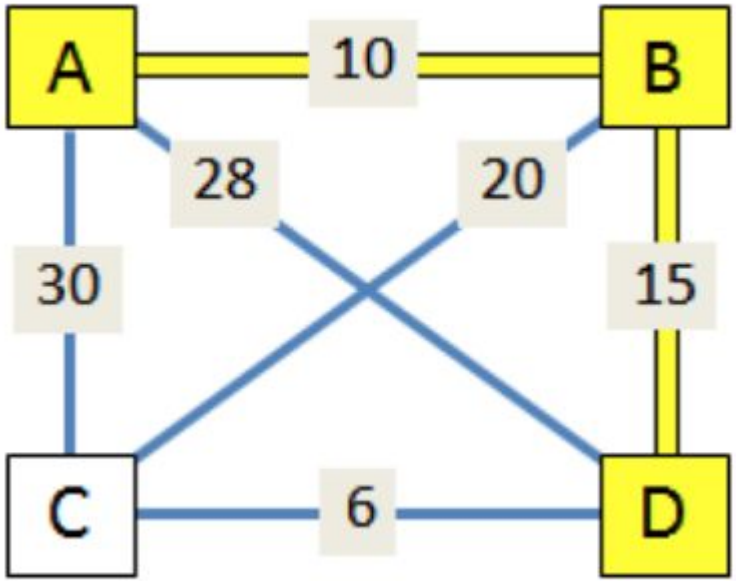
CS180 Discussion

Week 4

Lecture Recap

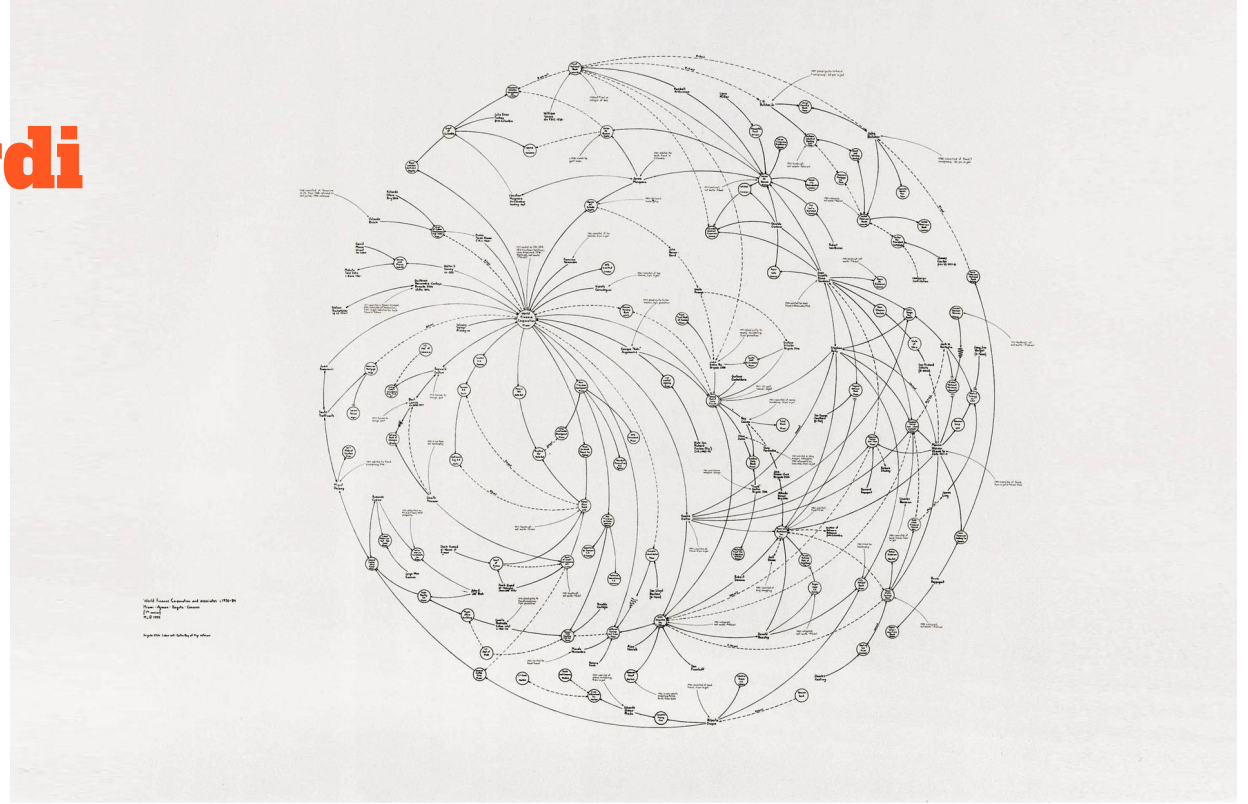
- Dijkstra continued
- Prim's algorithm
- Kruskal
- Minimum spanning trees
- Merge sort
- Some time complexity analysis

Dijkstra VS Prim VS Kruskal



Solving HW3

Mark Lombardi



we are given an undirected graph G = (V , E), and we identify two nodes v and w in G. Give an algorithm that computes the number of shortest v-w paths in G. (The algorithm should not list all the paths; just the number suffices.) The running time of your algorithm should be O(m + n) for a graph with n nodes and m edges.

G = T

We have a connected graph G=(V,E), and a specific vertex $u \in V$. Suppose we compute a depth-first search tree rooted at u, and obtain a tree T that includes all nodes of G. Suppose we then compute a breadth-first search tree rooted at u, and obtain the same tree T. Prove that G = T. (In other words, if T is both a depth-first search tree and a breadth-first search tree rooted at u, then G cannot contain any edges that do not belong to T.)

Is G connected

Claim: Let G be a graph on n nodes, where n is an even number. If every node of G has degree at least n/2, then G is connected.

Decide whether you think the claim is true or false, and give a proof of either the claim or its negation.

Greedy trucks

Prove that, for a given set of boxes with specified weights, the greedy algorithm currently in use actually minimizes the number of trucks that are needed. Your proof should follow the type of analysis we used for the Interval Scheduling Problem: it should establish the optimality of this greedy packing algorithm by identifying a measure under which it “stays ahead” of all other solutions.

Athletic kids

each contestant must swim 20 laps of a pool, then bike 10 miles, then run 3 miles. The plan is to send the contestants out in a staggered fashion, via the following rule: the contestants must use the pool one at a time. In other words, first one contestant swims the 20 laps, gets out, and starts biking. As soon as this first person is out of the pool, a second contestant begins swimming the 20 laps; as soon as he or she is out and starts biking, a third contestant begins swimming . . . and so on.) Each contestant has a projected swimming time, a projected biking time, and a projected running time. Your friend wants to decide on a schedule for the triathlon.

Longest path

a. Can you design an algorithm that finds the longest path in a directed graph (DG)? (in a path you can use an edge at most once)? If yes, describe the algorithm and analyze its time complexity.

b. Can you design an algorithm that finds the longest path in a directed **acyclic** graph (DAG)? (in a path you can use an edge at most once)? If yes, describe the algorithm and analyze its time complexity.

Interview Questions!!!

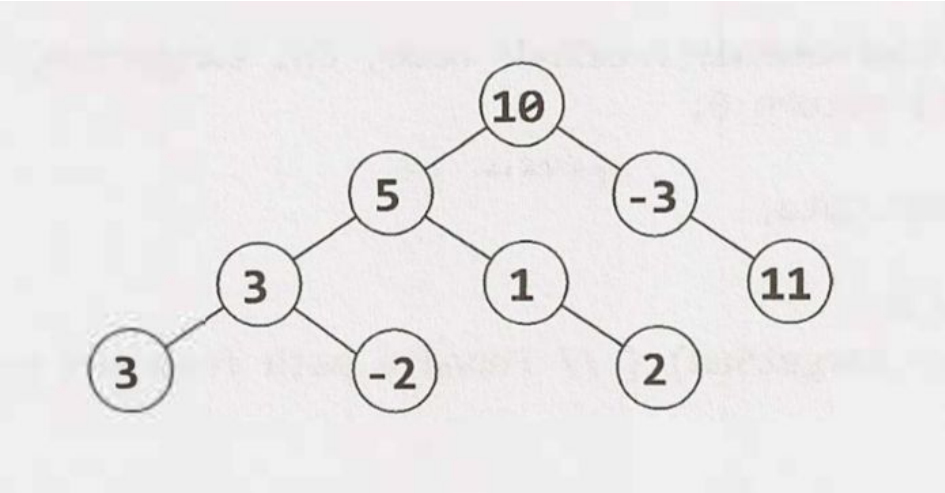
Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

Paths with Sum

You are given a binary tree in which each node contains an integer value (which might be positive or negative). Design an algorithm to count the number of paths that sum to a given value. The path does not need to start or end at the root or a leaf, but it must go downwards (traveling only from parent nodes to child nodes).



CS180 Discussion

Week 5

Lecture Recap

- Introduction to the midterm
- Clustering problem -> maximizing the minimum distance between clusters
- Counting Inversions (divide and conquer)

Solving HW4

A photocopying service

you are given a set of n jobs with a processing time t_i and a weight w_i for each job. You want to order the jobs so as to minimize the weighted sum of the completion times, $\sum_{i=1}^{n} w_i C_i$.

Example. Suppose there are two jobs: the first takes time t1 = 1 and has weight w1 = 10, while the second job takes time t2 = 3 and has weight w2 = 2.

Then doing job 1 first would yield a weighted completion time of 10 · 1 + 2 · 4 = 18, while doing the second job first would yield the larger weighted completion time of 10 · 4 + 2 · 3 = 46.

Variation on the Interval Scheduling Problem

Given a list of n such jobs, your goal is to accept as many jobs as possible (regardless of their length), subject to the constraint that the processor can run at most one job at any given point in time. Provide an algorithm to do this with a running time that is polynomial in n. You may assume for simplicity that no two jobs have the same start or end times.

Example. Consider the following four jobs, specified by (start-time, end- time) pairs.

(6 P.M., 6 A.M.), (9 P.M., 4 A.M.), (3 A.M., 2 P.M.), (1 P.M., 7 P.M.).

The optimal solution would be to pick the two jobs (9 P.M., 4 A.M.) and (1 P.M., 7 P.M.), which can be scheduled without overlapping.

Credit card collection

take two bank cards and, after performing some computations, determines whether they are equivalent.

Their question is the following: among the collection of n cards, is there a set of more than n/2 of them that are all equivalent to one another? Assume that the only feasible operations you can do with the cards are to pick two of them and plug them in to the equivalence tester. Show how to decide the answer to their question with only O(n log n) invocations of the equivalence tester.

Shifted array

Suppose you are given an array of sorted integers that has been circularly shifted k positions to the right. For example taking (1, 3, 4, 5, 7) and circularly shifting it 2 position to the right you get (5, 7, 1, 3, 4). Design an efficient algorithm for finding k.

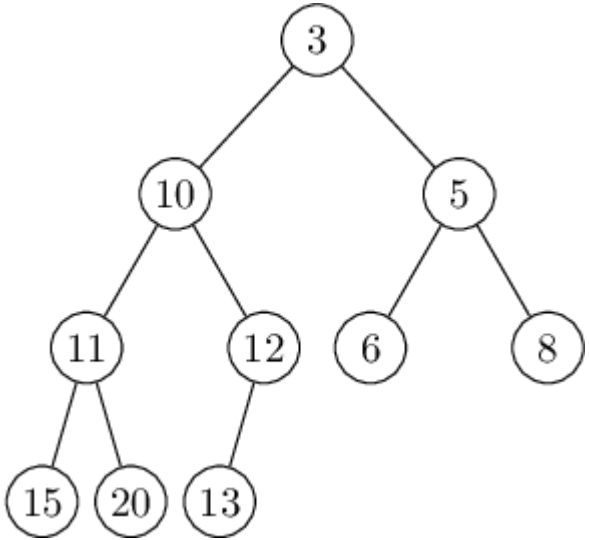
Chess board

Suppose now that you're given an n×n grid graph G. (An n×n grid graph is just the adjacency graph of an n × n chessboard. To be completely precise, it is a graph whose node set is the set of all ordered pairs of natural numbers (i, j), where 1≤ i ≤ n and 1≤ j ≤ n; the nodes (i, j) and (k, l) are joined by an edge if and only if |i − k| + |j − l| = 1.)

We use some of the terminology of the previous question. Again, each node v is labeled by a real number x_v; you may assume that all these labels are distinct. Show how to find a local minimum of G using only O(n) probes to the nodes of G. (Note that G has n2 nodes.)

Heap

Consider a (balanced) heap on n nodes. Show details of how you extract the minimum, insert a new number, and change a number (along with the corresponding post heapify process). Analyze the time complexity of your three algorithms.



Question 5

Consider an n-node complete binary tree T, where $n = 2d - 1$ for some d. Each node v of T is labeled with a real number x_v. You may assume that the real numbers labeling the nodes are all distinct. A node v of T is a local minimum if the label x_v is less than the label x_w for all nodes w that are joined to v by an edge.

You are given such a complete binary tree T, but the labeling is only specified in the following implicit way: for each node v, you can determine the value x_v by probing the node v. Show how to find a local minimum

Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

Sub Sort

Given an array of integers, write a method to find indices m and n such that if you sorted elements m through n , the entire array would be sorted. Minimize n - m (that is, find the smallest such sequence).

EXAMPLE

Input: 1, 2, 4, 7, 10, 11, 7, 12, 6, 7, 16, 18, 19

Output: (3, 9)

Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

CS180 Discussion

Week 7

Lecture Recap

- Dynamic programming
    - Weighted interval
    - Knapsack
- Network flow

Knapsack

(Val / Wt)	0	1	2	3	4	5	6
(1/1)	0	1	1	1	1	1	1
(4/3)	0	1	1	4	5	5	5
(5/4)	0	1	1	4	5	6	6
(7/5)	0						
(8/6)	0						

Longest Common Subsequence

		0	1	2	3	4	5	6	7
		Ø	M	Z	J	A	W	X	U
0	Ø	**0**	0	0	0	0	0	0	0
1	X	0	0	0	0	0	0	1	1
2	M	0	**1**	1	1	1	1	1	1
3	J	0	1	1	**2**	2	2	2	2
4	Y	0	1	1	2	2	2	2	2
5	A	0	1	1	2	**3**	3	3	3
6	U	0	1	1	2	3	3	3	**4**
7	Z	0	1	2	2	3	3	3	4

Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

Interview Questions!!!

The Masseuse

A popular masseuse receives a sequence of back-to-back appointment requests and is debating which ones to accept. He/She needs a 15-minute break between appointments and therefore he/she cannot accept any adjacent requests. Given a sequence of back-to-back appointment requests (all multiples of 15 minutes, none overlap, and none can be moved), find the optimal (highest total booked minutes) set the masseuse can honor. Return the number of minutes.

EXAMPLE

Input: {30, 15, 60, 75, 45, 15, 15, 45}

Output 180 minutes ({30, 60, 45, 45}).

Triple Step

A child is running up a staircase with n steps and can hop either 1 step, 2 steps, or 3 steps at a time. Implement a method to count how many possible ways the child can run up the stairs.

Triple Step Code

```
1   int countWays(int n) {
2      int[] memo = new int[n + 1];
3      Arrays.fill(memo, -1);
4      return countWays(n, memo);
5   }
6
7   int countWays(int n, int[] memo) {
8      if (n < 0) {
9         return 0;
10     } else if (n == 0) {
11        return 1;
12     } else if (memo[n] > -1) {
13        return memo[n];
14     } else {
15        memo[n] = countWays(n - 1, memo) + countWays(n - 2, memo) +
16                  countWays(n - 3, memo);
17        return memo[n];
18     }
19  }
```

Towers of Hanoi

In the classic problem of the Towers of Hanoi, you have 3 towers and N disks of different sizes which can slide onto any tower. The puzzle starts with disks sorted in ascending order of size from top to bottom (i.e., each disk sits on top of an even larger one). You have the following constraints:

(1) Only one disk can be moved at a time.

(2) A disk is slid off the top of one tower onto another tower.

(3) A disk cannot be placed on top of a smaller disk.

Write a program to move the disks from the first tower to the last using Stacks.

Towers of Hanoi

```
1   moveDisks(int n, Tower origin, Tower destination, Tower buffer) {
2     /* Base case */
3     if (n <= 0) return;
4
5     /* move top n - 1 disks from origin to buffer, using destination as a buffer. */
6     moveDisks(n - 1, origin, buffer, destination);
7
8     /* move top from origin to destination
9     moveTop(origin, destination);
10
11    /* move top n - 1 disks from buffer to destination, using origin as a buffer. */
12    moveDisks(n - 1, buffer, destination, origin);
13  }
```