Final Examination

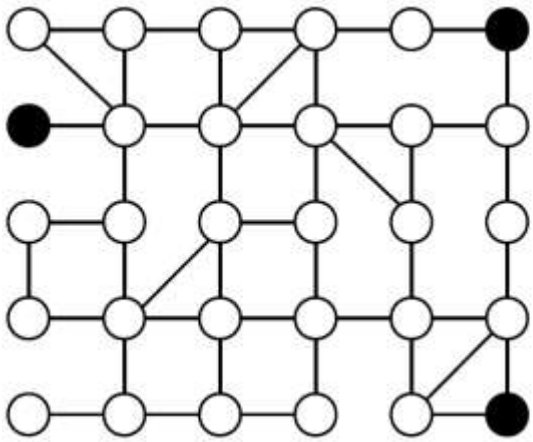
May 25, 2021

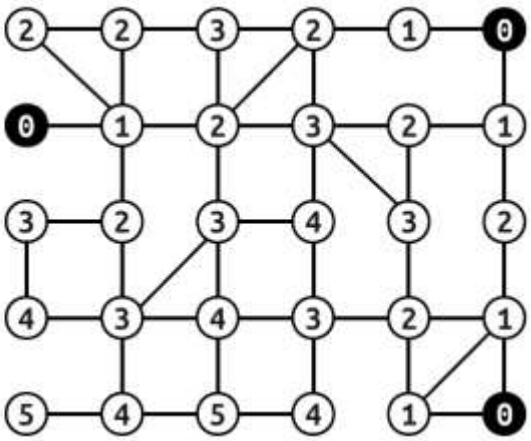
Part A – Graphs

Question 1 [20 marks]

Suppose that we have an undirected graph representing a city's road network. Each edge represents a road (a two-way street), and each node represents a location.

Certain locations in the city have hospitals, and you are interested in finding, for each location, the distance that location is from the nearest hospital, as measured by the number of edges in the path from that location to the nearest hospital. For example, given the following road network, where black nodes represent hospitals, the distances are as follows:





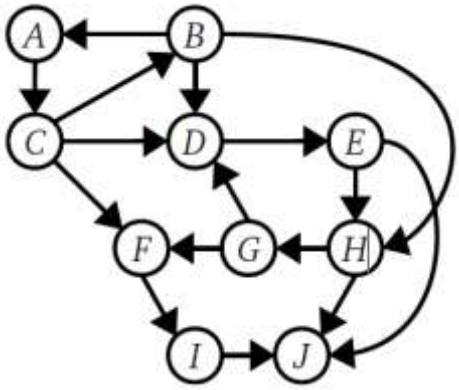
Let n be the number of nodes in the road network and let m be the number of edges. Although in the above example there were exactly three hospitals, any number of nodes in the network can be hospitals.

i. Design an $O(m + n)$-time algorithm for computing the distance from each location to the closest hospital. Note that your algorithm's asymptotic runtime should not depend on the total number of hospitals. (Hint: Try using one of the algorithms we've covered in the lectures as a subroutine.) [14 marks]

ii. Suppose that the city's road network has some one-way streets, meaning that some of the edges in the transportation network are directed. Briefly describe how you would modify your algorithm from part (i) to account for this while still maintaining the $O(m + n)$ runtime. You don't need to write a formal proof here – just give a one-paragraph description of your modified algorithm and a brief justification of why it works. [6 marks]

Question 2 [20 marks]

Suppose that there is a tournament involving n players, where certain pairs of players play a single game against one another. We can then visualize the tournament as a directed graph, where each node is a player and there is an edge from u to v iff u played v and won her game against him.



For example, in the tournament above, player B directly beat A, D, and H, but lost to C; player G directly beat D and F, but lost to H; player C beat players B, D, and F, but lost to A; and player J won no games and lost to E, H, and I.

We will say that a player p_1 transitively beat a player p_2 iff there is a path from p_1 to p_2 in the graph of the tournament. Intuitively, if p_1 and p_2 are different players, either p_1 directly beat p_2 in a game, or p_1 won against a player who in turn transitively beat p_2. Note that it is possible that two players transitively beat one another. In the above graph, player G transitively beat player H (since G beat D, who beat E, who beat H), but H also transitively beat G (since H directly beat G.)

Finally, define a tournament champion to be a player c who transitively beat every player in the tournament. A tournament can have multiple champions; for example, in the above tournament, players A, B, and C are champions, and no other players are. Some tournaments might not have any champions at all.

Let n be the number of players and m the total number of games played. Design an $O(m + n)$-time algorithm for finding all of the tournament champions in a tournament. If there aren't any winners, your algorithm should not output anyone.

Hint: Does finding SCC helps?

Part B – Dynamic Programming

Question 1 [20 marks]

In this question, we solve the problem of covering a line of towns, by placing cell towers in some of them to cover all with minimum cost. Given that the cell towers transmit with enough power so to provide coverage to the town in which they are built and to adjacent towns.

The challenge is to determine the cheapest possible way to build cell towers that cover everyone. Suppose that you have a cost c_i associated with building a cell tower in town i. Your job is to determine the minimum cost of cell towers necessary to cover all of the towns. For example, suppose the towns are arranged in this linear order with their costs indicated:

$314	$159	$265	$358	$979	$323	$846	$264	$338
Town 1	Town 2	Town 3	Town 4	Town 5	Town 6	Town 7	Town 8	Town 9

An expensive way to provide coverage to all towns would be to buy a cell tower in each town. A better option would be to build cell towers in the first, third, fifth, etc. towns, as shown here:

$314	$159	$265	$358	$979	$323	$846	$264	$338
✓		✓		✓		✓		✓

An even better option would be to purchase these towers:

$314	$159	$265	$358	$979	$323	$846	$264	$338
	✓		✓		✓		✓	

An even better option would be to purchase these towers:

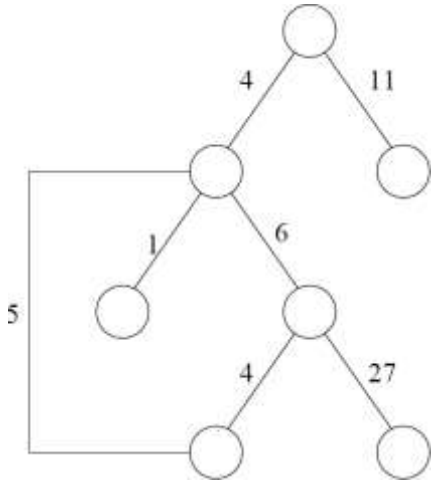
$314	$159	$265	$358	$979	$323	$846	$264	$338
	✓	✓			✓		✓	

Design an algorithm that finds the minimum cost necessary to provide cell coverage to all of the towns. Note that you just need to find the cost of the necessary cell towers and do not need to report which towers you will purchase. Your algorithm should run in time polynomial in n, where n is the number of towns.

(Hint: Consider having different types of subproblems for the case where the towns on the boundary are already covered and where the towns on the boundary still need to be covered.)

Question 2 [20 marks]

Consider an undirected weighted graph which is formed by taking a binary tree and adding an edge from exactly one of the leaves to another node in the tree. We call such a graph a loop-tree. An example of a loop-tree could be the following:



Let n be the number of vertices in a loop-tree and assume that the graph is given in the normal adjacency-list representation without any extra information. In particular, the representation does not contain information about which vertex is the root.

i. How long time would it take Prim's or Kruskal's algorithms in worst case to find the minimal spanning tree of a loop-tree? Make your bound as tight as possible. [6 marks]

ii. Describe and analyze a more efficient algorithm for finding the minimal spanning tree of a loop-tree. Remember to prove that the algorithm is correct. [14 marks]

Part C – Analysis, Divide & Conquer Algorithms

Question 1 [10 marks]

The Counting-Sort is an algorithm to sort an array of integers, in which each integer is less than U, for some number U. For instance, consider the input array $A = \{4, 3, 2, 4, 0, 1, 4, 2, 2, 0\}$, where $n = 10$ and $U = 5$. Here is pseudocode for Counting-Sort:

```
countingSort(array A, int U)

1. counts = new array of size U
2. for i = 0 to U - 1:
3.     counts[i] = 0
4. for i = 0 to length(A) - 1:
5.     counts[A[i]] = counts[A[i]] + 1
6. index = 0
7. for i = 0 to U - 1:
8.     for j = 0 to counts[i] - 1:
9.         A[index] = i
10.        index = index + 1
```

One correct but loose analysis of counting sort's runtime is the following:

"The loop on line 2 runs $O(U)$ times and does $O(1)$ work on each iteration, so it does a total of $O(U)$ work. The loop on line 4 runs $O(n)$ times (where n is the length of the input array) and does $O(1)$ work on each iteration, so it does a total of $O(n)$ work. Thus the initial setup takes $O(n + U)$ time. Loop 7 has a nested loop within it. The nested loop (loop 8) can execute at most $O(n)$ times (because `counts[i]` $\leq$ `n`) and does $O(1)$ work on each iteration, so it does at most $O(n)$ work. Therefore, since Loop 7 runs $O(U)$ times, the work done by the loop is $O(nU)$.
Therefore, the overall work done is $O(nU)$."

While this analysis of the runtime is correct, it overestimates the amount of work done by this counting sort. Prove that counting sort actually runs in time $\Theta(n + U)$.

(When doing this analysis, you do not need to use the formal definition of Θ notation. You can use an intuitive analysis along the lines of the above.)

Question 2 [10 marks]

Explain which of the following sorting tasks would not be appropriate for Counting-Sort and why?

a) sorting list of IBA CS students by their 13-digit CNIC numbers
b) sorting millions of employees by their ages
c) sorting each day of the calendar year by the high temperature (in Celsius rounded to the nearest integer degree) recorded in IBA Karachi on that given day
d) sorting each vehicle registered in Sindh by the year it was manufactured

Question 3 [10 marks]

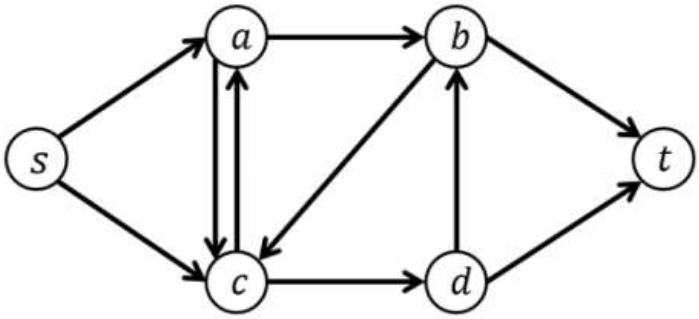
Given an n-bit number and m-bit number where $n \geq m$, recall that multiplying them using the standard school method takes $O(mn)$ time while Karatsuba's algorithm takes $O(n^{\log_2 3})$ time. So, for $m = o(n^{\log_2 3 - 1})$, the school method runs asymptotically faster than Karatsuba's.

i. Argue that when $n \geq 2m$, Karatsuba's algorithm can get away with using only two (2) recursive multiplications of smaller numbers instead of three (3). [6 marks]

ii. Modify Karatsuba's recursive divide-and-conquer algorithm to take advantage of the observation from part (i) at each level of recursion (if applicable). Write down the recurrence relation(s) describing the running time of the modified algorithm (No need to solve it). [4 marks]

Part D – Network Flows

Question 1 [20 marks]



Given the directed graph above, let define edge capacities (using your 5-digits erp `uwxyz`) as follows:

```
c(s,a) = u + 5
c(s,c) = w + 1
c(a,c) = x + 2
c(c,a) = y + 1
c(a,b) = z + 2
c(c,d) = u + 3
c(b,c) = w + 1
c(d,b) = x + 3
c(b,t) = y + 2
c(d,t) = z + 7
```

Run the Ford-Fulkerson algorithm on the resulting network showing the residual graph after each iteration.

What is value of maximum flow? Show the minimum cut.

Test – Apr 22

Knapsack

Exercise 1 [10 marks]

Solve the following instance of the discrete Knapsack Problem (without repetitions) with four items where the maximum allowed weight is $W_{max} = 5$.

Item (i)	**Weight (w_i)**	**Value (v_i)**
1	1	2
2	3	5
3	5	7
4	4	6

Quiz – Apr 15

Edit Distance

Exercise 1 [10 marks]

Compute edit-distance between the strings X = <your-firstname> and Y = <your-lastname> using Dynamic Programming based algorithm discussed in the lectures. (If your first name or last name have more than 5 letters then consider the first 5 letters only)

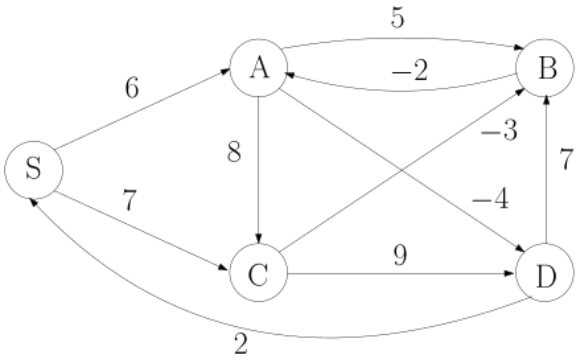
What is the optimal alignment computed?

Test – April 1

Bellman-Ford and Kruskal Algorithms

Exercise 1 [10 marks]

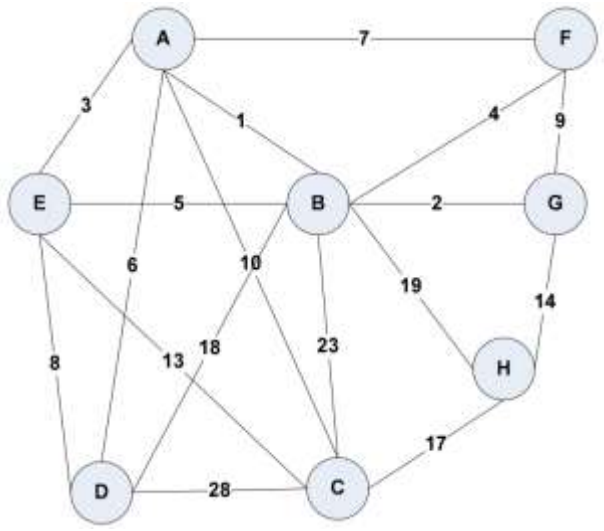
Suppose Bellman-Ford's algorithm is run on the following graph, starting at node S.



Draw a table showing the intermediate distance values of all the nodes at each iteration of the algorithm. Also, draw the final shortest-path tree.

Exercise 2 [10 marks]

Consider the graph below.



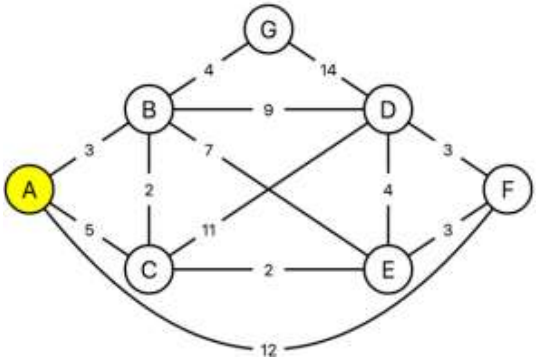
When executing Kruskal's algorithm, what is the fifth edge that is chosen? Give a cut that justifies its addition. What is the total weight of minimum spanning tree in this graph? Does this graph have a unique minimum spanning tree? Explain briefly.

Quiz – Mar 16

Dijkstra

Exercise 1 [2 marks]

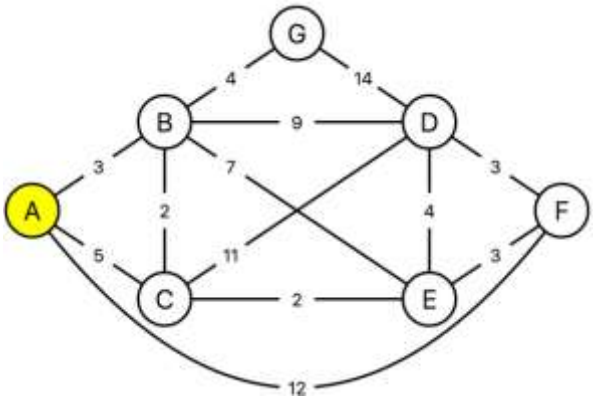
In the following directed graph with edge-weights, what is the shortest path from vertex A to vertex F?



i. A -> B -> D -> F
ii. A -> C -> B -> E -> F
iii. A -> F
iv. A -> C -> E -> F

Exercise 2 [2 marks]

In the following directed graph, if we apply Dijkstra's algorithm to find the shortest distance between vertex A and all the others, in what order do the vertices removed from the priority queue (i.e., their distances have been finalized)?



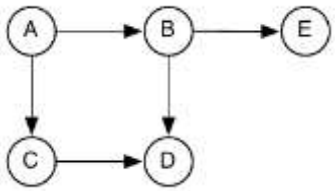
i. B C F G E D
ii. B C G E F D
iii. C B E F G D
iv. C B E G F D

Quiz – Mar 9

DFS Post/Pre numbering

Exercise 1 [5 marks]

Recall the implementation of DFS shown in class. Mark the correct answer for each of the following statements, assuming DFS starts at A



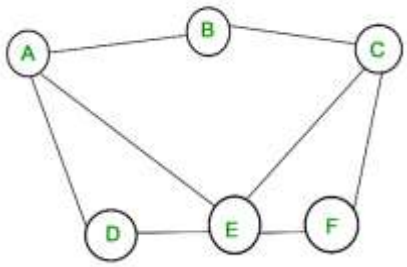
i. D could be assigned pre number before E, i.e., pre(D) < pre(E)
ii. E could be assigned pre number before D, i.e., pre(E) < pre(D)
iii. D could be assigned pre number before C, i.e., pre(D) < pre(C)
iv. A could be assigned post number before B, i.e., post(A) < post(B)
v. D could be assigned post number before B is explored, i.e., post(D) < pre(B)

Quiz – Mar 2

DFS

Exercise 1 [2 marks]

Consider the following graph,



Among the following sequences:

I. A B C D E F
II. A B C F E D
III. A D E F C B
IV. A D E C B F

Which ones are depth first traversals of the above graph?

a) I, II and IV only
b) II, III and IV only
c) I, II and III only
d) I, III and IV only

Exercise 2 [2 marks]

Given two vertices in a graph s and t, which of the two traversals (BFS and DFS) can be used to find if there is path from s to t?

a) Only BFS
b) Only DFS
c) Both BFS and DFS
d) Neither BFS nor DFS

Quiz – Feb 16

QuickSelect

Exercise 1 [2 marks]

In QuickSelect algorithm discussed in the last lecture, suppose you choose a pivot randomly for an array S of size n. In the lecture we have seen that the probability that both SL and SR have size at most 3n/4 is 1/2.

The probability that the resulting two lists S_L and S_R will both have size at most $\frac{2n}{3}$ is ____.

Exercise 2 [2 marks]

Assuming the initial array has n elements, the probability that a worst-case pivot is chosen in QuickSelect is ____.

Quiz – Feb 18

Asymptotic Notations

Exercise 1 [2 marks]

Attempt two out of three (randomly selected)

a) 10^{1000} is $O(n)$
b) 10^{1000} is $\Omega(n)$
c) 10^{1000} is $\Theta(n)$

Exercise 2 [2 marks]

Attempt two out of three (randomly selected)

a) $n(n+1)/2$ is $O(n^2)$
b) $n(n+1)/2$ is $\Omega(n^2)$
c) $n(n+1)/2$ is $\Theta(n^2)$

Exercise 3 [2 marks]

Attempt two out of three (randomly selected)

a) $n\log(n)$ is $O(n)$
b) $n\log(n)$ is $\Omega(n)$
c) $n\log(n)$ is $\Theta(n)$

Quiz – Feb 2

Divide & Conquer

Exercise 1 [2 marks]

The running time of 'School method' to add two n-digit integers is:

a) $O(1)$
b) $O(\log n)$
c) $O(n)$
d) $O(n^2)$

Exercise 2 [2 marks]

The running time of 'School method' to multiply two n-digit integers is:

a) $O(1)$
b) $O(\log n)$
c) $O(n)$
d) $O(n^2)$

Exercise 1 [2 marks]

The running time of 'Recursive method' to multiply two n-digit integers is:

a) $O(\log n)$
b) $O(n)$
c) $O(n \log n)$
d) $O(n^2)$