COSC-311 Sample Midterm Questions

Note: There will be four problems on the actual midterm. The problems on this handout are meant to give you a sense of the types of questions I might ask. This study guide is not comprehensive: there are topics we have covered in class that are not represented in the sample problems. Everything we have done in class or on homework is fair game for the midterm.

1 Minimum Spanning Trees

Ryan's cool new Minimum Spanning Tree algorithm works as follows on a graph with n vertices:

- Initialize set $T = \emptyset$ and $S = u$, where u is a randomly chosen vertex.
- Iterate $n-1$ times: Let v be the vertex most recently added to S. Consider all edges from v to some other vertex x that has not yet been added to S. Among those edges, let (v, w) be the edge of lowest cost. Add edge (v, w) to T and add w to S.
- If v doesn't have any neighbors not already in S, go back to the second-most-recently added vertex, and keep backtracking until you find a vertex that has at least one neighbor not already in S. Use that vertex instead of v in this iteration.

Sadly this algorithm does not work.

(a) State the theorem that tells us which edges are safe to add to a partial MST T.

(b) Explain why Ryan's algorithm doesn't fit the criteria for this theorem.

(c) Give an example of a graph with at least 3 vertices for which Ryan's algorithm returns the wrong answer. What does the algorithm do? What is the correct MST?

2 Scheduling

In the Interval Scheduling problem, we were given a set of jobs, where each job i required our resource starting at time $a(i)$ and ending at time $f(i)$, and only one job could be scheduled to use the resource at a time. If two jobs conflict, we must discard one of them. Our goal was to come up with a subset of jobs $\mathcal{S}$ such that there are no conflicts among the jobs in $\mathcal{S}$, and $|\mathcal{S}|$ is maximized.

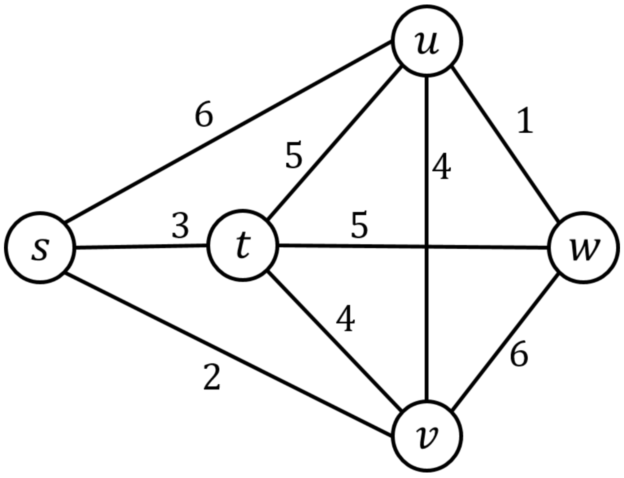
(a) Dana suggests the following greedy algorithm: let $x(i) = f(i) - a(i)$ be the *size* of job i, and let $n(i)$ be the number of jobs conflicting with job i. Add to $\mathcal{S}$ the job i

with the smallest $x(i) * n(i)$, then throw out any jobs conflicting with i and recur on the remaining jobs. Give a counterexample that shows that Dana's algorithm does not always produce the optimal solution.

(b) What is the correct greedy algorithm to solve this problem optimally?

3 More Minimum Spanning Trees

Here is a graph:



(a) Draw the minimum spanning tree that would result from running Prim's algorithm on this graph, starting at vertex s. List the order in which edges are added to the tree.

(b) Draw the minimum spanning tree that would result from running Kruskal's algorithm on this graph. List the order in which edges are added to the tree.

4 Recurrences

Allie and Brandon each have come up with a divide-and-conquer algorithm to solve a problem.

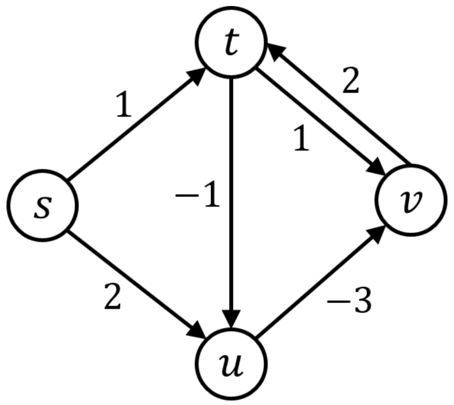
- Allie's algorithm (algorithm A) splits a problem of size n into four pieces, each of size $n/2$, solves the pieces recursively, and takes time $30n$ to combine the results.
- Brandon's algorithm (algorithm B) takes time n^3 to turn a problem of size n into a smaller problem of size $n/2$, which it then solves recursively.

(a) Write a recurrence for the runtime of Allie's algorithm, $T_A(n)$, and a recurrence for the runtime of Brandon's algorithm, $T_B(n)$.

(b) Which algorithm has an asymptotically better runtime? Justify your answer by solving each recurrence using the Master Theorem.

5 Shortest Paths

Here is a graph:



(a) Run the Bellman-Ford algorithm on the graph to find the shortest path from s to every other node. Show the table that you fill in while running the algorithm.

(b) Now suppose you ran one extra iteration of Bellman-Ford. How could you use that last iteration to detect whether the graph has a negative-weight cycle?

6 Making Change

Suppose you are traveling to a foreign country where the local currency has four coins with denominations $d_1 = 1$, $d_2 = 5$, $d_3 = 8$, and $d_4 = 14$ cents. As usual, when making change the goal is to end up with as few coins as possible.

(a) Give a counterexample to show that the greedy algorithm for making change no longer yields the optimal solution.

(b) Write a recurrence for a dynamic programming algorithm that computes the minimum number of coins needed to make change for n cents, $C(n)$.