CSE 550: Final Exam Study Guide

Prof. Joshua J. Daymude

This study guide is intended to help you prepare for the Final Exam. The exam is comprehensive, meaning it covers the full semester's material (as opposed to only the material covered after the Midterm Exam). The exam is closed book, closed notes, and will be completed in-person within the hour and fifty minute final exam period. As stated on the syllabus, this exam counts for 25% of your final grade.

1 Exam Questions and Topics

The exam will have six questions covering a comprehensive set of topics from the full semester of material. The rest of this section describes the format and topics of the six questions in hopes that it will help you study effectively and feel more prepared. By coincidence, the exam will be graded out of 100 points and you will have roughly 100 minutes to complete the exam. I have designed the exam such that you should be able to complete a problem worth X points in roughly X minutes, including some buffer time for thinking.

1.1 Problem 1: Definitions, Problems, and Theorems [20 points]

This problem tests your understanding of important definitions, problem statements, and theorems covered in this class. These may be taken from the lecture slides, the proof supplements, or the assignments. Specifically, you will be given ten (10) true-or-false statements, where correct answers are worth 2 points each. They will either be the correct statement of a definition, problem, or theorem covered in class (true) or they will have some modification that makes their statement incorrect (false). False statements can be incorrect because they are logically incompatible with the correct statement(s), or because they are insufficient compared to the correct statement(s). For example, consider the following correct definition:

Definition (Correct)**.** *Given an undirected graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_{\geq 0}$ and nodes $s, t \in V$, a shortest (s,t)-path in G is an ordering of some subset of nodes $(s = v_1, v_2, \ldots, v_k = t)$ such that $(v_i, v_{i+1}) \in E$ for all $1 \leq i < n$ and, for every other such (s,t)-path $(s = u_1, u_2, \ldots, u_\ell = t)$, we have*

$$\sum_{i=1}^{k-1} w((v_i, v_{i+1})) \leq \sum_{i=1}^{\ell-1} w((u_i, u_{i+1}))$$

This is a correct definition of a shortest (s,t)-path. It defines all its parameters, defines an (s,t)-path, and defines what it means to be shortest. But it is not the *only* correct definition of a shortest (s,t)-path; there may be others that are logically equivalent. For example, the presented definition defines a path as an ordering of nodes. You could also define a path as a set of edges. This changes the statement but not the correctness of the definition. On the other hand, consider the following incorrect definition:

Definition (Incompatible)**.** *Given an undirected graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_{\geq 0}$ and nodes $s, t \in V$, a shortest (s,t)-path in G is an ordering of some subset of nodes $(s = v_1, v_2, \ldots, v_k = t)$ such that $(v_i, v_{i+1}) \in E$ for all $1 \leq i < n$ and, for every other such (s,t)-path $(s = u_1, u_2, \ldots, u_\ell = t)$, we have*

$$\sum_{i=1}^{k-1} w((v_i, v_{i+1})) > \sum_{i=1}^{\ell-1} w((u_i, u_{i+1}))$$

All that has changed is the direction of the inequality in comparing path lengths: from $\leq$ to $>$. This change is logically incompatible with *all* correct definitions. We are no longer defining a shortest (s,t)-path, but a longest one. Consider another way a definition could be incorrect:

Definition (Insufficient)**.** *Given an undirected graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_{\geq 0}$ and nodes $s, t \in V$, a shortest (s,t)-path in G is an ordering of some subset of nodes $(s = v_1, v_2, \ldots, v_k = t)$ such that, for every other such (s,t)-path $(s = u_1, u_2, \ldots, u_\ell = t)$, we have*

$$\sum_{i=1}^{k-1} w((v_i, v_{i+1})) \leq \sum_{i=1}^{\ell-1} w((u_i, u_{i+1}))$$

Everything in this statement is logically compatible with any correct definition. However, it lacks a very important detail: it does not require that the ordering of nodes $(v_1, \ldots, v_k)$ forms a connected path! This statement is thus incorrect because it is *insufficient*; in this case, all shortest (s,t)-paths would satisfy this definition, but so would some non-(s,t)-paths.

1.2 Problem 2: IP/LP Formulations [15 points]

This problem tests your ability to formulate a combinatorial problem statement as an integer or linear program and understand various properties about the resulting IP/LP. This problem will open with a statement of one of the following combinatorial problems: Minimum-Vertex-Cover, Minimum-Edge-Cover, Maximum-Independent-Set, Minimum-Dominating-Set, Maximum-Matching, Minimum-Perfect-Matching, Knapsack, Minimum-Spanning-Tree, or Traveling-Salesman. Note that you have seen all of these problems (and most of their IP/LP formulations) in the lecture slides, proof supplements, or assignments. You will then be asked to do three things:

(a) Write an IP for the combinatorial problem. Unlike on the Midterm Exam where you were asked one part at a time (variables, objective function, and constraints), you will write the full IP and some explanation of the important details (how are the variables defined? what do the constraints mean?).

(b) Count the number of constraints in your IP (asymptotically), *excluding* integrality constraints.

(c) Consider some small modification of the combinatorial problem and explain how it would change your IP formulation from part (a).

Based on the assignments and the Midterm Exam, here are some common mistakes to avoid.

- Make sure your IP has all its parts: an objective function, constraints, and integrality constraints. Forgetting to write any of these leaves points on the table.
- Don't include your integrality constraints when counting the number of constraints in your IP. The purpose of this question is to figure out what m is for your $m \times n$ constraint matrix A.
- In a constraint, the range of a sum is not the same thing as a quantifier for the constraint. Specifically, the following constraints are not the same:

$$\sum_{x \in X} f(x) = b \qquad \text{vs.} \qquad f(x) = b \quad \forall x \in X$$

  On the left, we have a single constraint that says "when we add up $f(x)$ for all $x \in X$, the sum should equal b". On the right, we have $|X|$ separate constraints that say "$f(x)$ should equal b" when considering each $x \in X$ individually. These are not interchangeable.

1.3 Problem 3: Simplex and Duality [20 points]

This problem tests your ability to translate between the different forms of an LP (general, standard, and canonical), initialize a simplex tableau, choose simplex pivots according to Bland's anticycling algorithm, write the dual of a primal LP, and derive optimal feasible solutions using complementary slackness. The problem starts with an LP in general form. You will then be asked to do five things:

(a) Write the LP in standard form.

(b) Write the initial tableau for the LP's standard form using your answer from part (a).

(c) Given a tableau from the simplex execution on this LP, identify the pivot that would be chosen by Bland's anticycling algorithm. Succinctly explain why that pivot is chosen.

(d) Given an incorrect dual of the original LP (*not* of the LP in standard form!), circle and fix the mistakes or write a corrected dual.

(e) Given the optimal solution to the original LP and the corrected dual from part (d), use complementary slackness to find the optimal solution to the dual LP.

1.4 Problem 4: Network Flows and Cuts [15 points]

This problem tests your understanding of network flows and the polynomial-time algorithms used to compute them. This is the only problem containing "new" material in the sense that you will need to extend what you know about network flows to a closely related topic that we have not explicitly discussed in class. Specifically, the problem will begin by defining some *graph quantity* of a directed graph $G = (V, E)$. For example, the shortest path length of a graph is a graph quantity. You will then be asked to do three things:

(a) Describe a polynomial-time algorithm, in words or pseudocode, that takes as input the directed graph $G = (V, E)$ and computes the desired graph quantity for G.

(b) State and briefly justify the running time of your algorithm.

(c) Make a small modification to your algorithm such that it not only computes the desired graph quantity but also the corresponding graph structure. For example, if the graph quantity were the shortest path length, this modification would need to also return the shortest path.

1.5 Problem 5: Matchings [15 points]

This problem tests your understanding of matchings and your mastery over the theoretical results we proved about them. Specifically, you will be asked to prove one of the following three results about matchings: Hall's Theorem (Lecture 3 proof supplement) or Problem 1 or 2 from Assignment 5 (not the extra credit). Any supporting lemmas or theorems for those proofs will be stated in the problem to be used without reproof.

1.6 Problem 6: NP-Completeness [15 points]

This problem tests your ability to prove a problem is NP-complete using a polynomial-time reduction from a known NP-complete problem. Specifically, you will be asked to prove one of the following decision problems is NP-complete: Longest-Path, Partition, Knapsack, Clique, Vertex-Cover, or Traveling-Salesman. Note that you have seen all of these problems' proofs of NP-completeness in either the Lecture 4 proof supplement or Assignment 6. You may reduce from any known NP-complete problem covered in the Lecture 4 proof supplement. Recall from Lecture 4 that your proof that Π is NP-complete should follow these steps:

1. Show Π is in NP by guess-and-check verifying any YES-instance in polynomial time. Remember that many decision problems are phrased as "Is there an x that satisfies condition y?". Usually, showing Π is in NP is as simple as guessing the right x and checking that it satisfies y. Don't forget to argue why the check can be done in polynomial time!

2. State the reduction $\Pi' \leq_p \Pi$ that you intend to prove, for some known NP-complete problem Π'. In the proof supplement, I usually say "To show Π is NP-complete, we prove that $\Pi' \leq_p \Pi$, where Π' is a known NP-complete problem."

3. Explain how you transform any instance I of Π' into the constructed instance $f(I)$ of Π. Remember that this *always* starts with "Consider any instance of Π'" (not of Π!), with details about what that instance looks like. You can explain this transformation in words or in pseudocode steps.

4. Succinctly argue that your reduction f can be computed in polynomial time. This usually is as simple as counting the number of operations or entities you construct in f and observing that that number is a polynomial in the size of the original instance.

5. Argue that I is a YES-instance of Π' if and only if $f(I)$ is a YES-instance of Π. Depending on the reduction, this can either be done in two parts (first considering I is a YES-instance of Π' and proving $f(I)$ is a YES-instance of Π; then doing the same for the reverse direction) or by showing some if-and-only-if relationship all the way through.

2 Frequently Asked Questions

- **What format is the exam?** The exam will be pen-and-paper, closed book and closed notes. You will have roughly 100 minutes to complete the exam. Details about the question topics are given above.
- **Is there a practice exam we can use to study?** No, this document is the complete study guide. If you want, you can try formulating and answering questions following the topics given above.
- **Can I bring a cheat sheet to the exam?** No.
- **Will there be a retake opportunity like we had for the Midterm Exam?** No.