CS 170 HW 11

Due on 2019-15-04, at 9:59 pm

1 Study Group

List the names and SIDs of the members in your study group.

2 Bipartite Vertex Cover

A *vertex cover* of an undirected graph $G = (V, E)$ is a subset of vertices which touches every edge - that is, a subset $S \subset V$ such that for each edge $(u, v) \in E$, one or both of u or v are in S. Design an efficient algorithm to find the minimum sized vertex cover in a *bipartite* graph. (*Hint:* Use the max-flow-min-cut theorem on the network obtained for the maximum bipartite matching problem.)

3 Direct Bipartite Matching

Let $G = (L \cup R, E)$ be a bipartite graph such that for every edge, one of the endpoints is in L and the other is in R. Let M be a matching of G. A vertex, v, is said to be covered by M if one of its edges is in the matching, M. An *alternating path* is a path of odd length that starts and ends with a non-covered vertex, and whose edges alternate between M and $E \setminus M$.

(a) Prove that a matching, M, is a maximum matching if and only if there does not exist an alternating path with respect to it.

(b) Design an algorithm to find such an alternating path if it exists in time $O(|V| + |E|)$ time using a variant of breath first search.

(c) Give a direct $O(|V| \cdot |E|)$ time algorithm for finding a maximum matching in a bipartite graph.

4 Zero-Sum Battle

Two Pokemon trainers are about to engage in battle! Each trainer has 3 Pokemon, each of a single, unique type. They each must choose which Pokemon to send out first. Of course each trainer's advantage in battle depends not only on their own Pokemon, but on which Pokemon their opponent sends out.

The table below indicates the competitive advantage (payoff) Trainer A would gain (and Trainer B would lose). For example, if Trainer B chooses the fire Pokemon and Trainer A chooses the rock Pokemon, Trainer A would have payoff 2.

		Trainer B:		
		ice	water	fire
Trainer A:	dragon	-10	3	3
	steel	4	-1	-3
	rock	6	-9	2

Feel free to use an online LP solver to solve your LPs in this problem.
Here is an example of a Python LP Solver and its Tutorial.

1. Write an LP to find the optimal strategy for Trainer A. What is the optimal strategy and expected payoff?

2. Now do the same for Trainer B. What is the optimal strategy and expected payoff?

5 Domination

In this problem, we explore a concept called *dominated strategies*. Consider a zero-sum game with the following payoff matrix for the row player:

Column:

		A	B	C
Row:	D	1	2	-3
	E	3	2	-2
	F	-1	-2	2

(a) If the row player plays optimally, can you find the probability that they pick D without directly solving for the optimal strategy? (Hint: Notice that the payoff for E is always greater than the payoff for D. When this happens, we say that E *dominates* D, i.e. D is a *dominated strategy*).

(b) Given the answer to part a, if the both players play optimally, what is the probability that the column player picks A?

(c) Given the answers to part a and b, what are both players' optimal strategies? (You might be able to figure this out without writing or solving any LP).