CS 170 DIS 08

Released on 2018-10-22

1 Taking a Dual

Consider the following linear program:

$$\begin{aligned} \max\ 4x_1 + 7x_2 \\ x_1 + 2x_2 &\leq 10 \\ 3x_1 + x_2 &\leq 14 \\ 2x_1 + 3x_2 &\leq 11 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Construct the dual of the above linear program.

2 Weighted Rock-Paper-Scissors

You and your friend used to play rock-paper-scissors, and have the loser pay the winner 1 dollar. However, you then learned in CS170 that the best strategy is to pick each move uniformly at random, which took all the fun out of the game.

Your friend, trying to make the game interesting again, suggests playing the following variant: If you win by beating rock with paper, you get 5 dollars from your opponent. If you win by beating scissors with rock, you get 3 dollars. If you win by beating paper with scissors, you get 1 dollar.

(a) Draw the payoff matrix for this game.

(b) Write a linear program to find the optimal strategy.

3 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).

4 MT2 Practice: Greedy

Assume there are n activities each with its own start time a_i and end time b_i such that $a_i < b_i$. All these activities share a common resource (think computers trying to use the same printer). A feasible schedule of the activities is one such that no two activities are using the common resource simultaneously. Mathematically, the time intervals are disjoint: $(a_i, b_i) \cap (a_j, b_j) = \emptyset$. The goal is to find a feasible schedule that maximizes the number of activities k.

Here are two potential greedy algorithms for the problem.

Algorithm A: Select the shortest-duration activity that doesn't conflict with those already selected until no more can be selected.

Algorithm B: Select the earliest-ending activity that doesn't conflict with those already selected until no more can be selected.

(a) Show that Algorithm A can fail to produce an optimal output.

(b) Show that Algorithm B will always produce an optimal output.

(c) **Challenge Problem:** Show that Algorithm A will always produce an output at least half as large as the optimal output.

5 MT2 Practice: MSTs

Let $G = (V, E)$ be an undirected, connected graph.

(a) Prove that there is a unique MST if all edge weights are distinct.

(b) True or False? If G has more than $|V| - 1$ edges, and there is a unique heaviest edge, then this edge cannot be part of a MST.

(c) True or False? If the lightest edge in G is unique, then it must be a part of every MST.

6 MT2 Practice: Dynamic Programming

Professor Rao loves to play golf, but as a combinatorial mathematician, he prefers to play a version called discrete golf. The goal of discrete golf is to hit a golf ball from checkpoint 1 to checkpoint n on a golf course in as few strokes as possible. Based on the terrain of the course, he knows that from checkpoint i, he can hit the ball up to $d(i)$ checkpoints away in one stroke. That is, if the ball is at checkpoint i, in one stroke he can hit the ball to any of checkpoints $i + 1, i + 2 \dots i + d(i)$.

(a) Suppose he hits the ball as far as possible every stroke, i.e. if he is at checkpoint i, he hits the ball to checkpoint $i + d(i)$. Give a counterexample where this greedy algorithm does not achieve the minimum number of strokes needed to reach checkpoint n from checkpoint 1.

(b) Suppose that all $d(i)$ are at most D. Give a $O(nD)$ time algorithm for computing the minimum number of strokes Professor Rao needs to reach point n. How much memory does this algorithm need?