Graduate Artificial Intelligence 15-780

Homework 4: *Mathematical Programming*

Out on March 1
Due on March 22

Problem 1: Linear and Integer Programming (40 points)

Consider the standard-form LP problem

$$\begin{aligned} \min\ (\text{or } \max) c^T x \\ \text{Subject to:} \\ Ax &= b \\ x &\geq 0 \end{aligned}$$

where $x \in \Re^n$ and A is an $m \times n$ matrix with $n \geq m$.

Recall that the simplex algorithm is an iterative algorithm to solve standard-form LPs. At each iteration, the simplex algorithm has a candidate solution x has the property that at least $n-m$ of its components are zero.

Variables that are allowed to be non-zero are called *basic* variables, and variables that are forced to be zero are called *non-basic* variables. Without loss of generality, we can assume that x is ordered so that all the basic variables come before all the non-basic variables. Letting x_B be the basic variables and x_N be the non-basic variables, we can rewrite x as

$$x = [x_B; x_N]$$

Now, the matrix A is partitioned into two components: an invertible $m \times m$ matrix that we will call S^{-1}; and C, the remaining $m \times (n-m)$ submatrix. We can rewrite A as

$$A = [S^{-1} C]$$

Now consider the original equality constraint of the LP

$$Ax = b$$

multiply both sides by S to get

$$SAx = Sb$$

and since $A = [S^{-1}C]$, this is just

$$[I_m SC]x = Sb$$

where I_m is the $m \times m$ identity matrix. Now, rewriting x we have

$$[I_m SC][x_B; x_N] = Sb$$

This system of equations is called the *simplex tableau.*

- (5 points) Consider the following LP:

$$\begin{aligned} \max x + y \\ \text{Subject To} \\ 2x + y &\leq 10 \\ x + 3y &\leq 12 \\ x, y &\geq 0 \end{aligned}$$

  What is the optimal solution of this LP? Sketch the feasible region.

- (5 points) What is the dual of this LP?

- (15 points) Now suppose we change the program to be an integer program by requiring that x and y both be integers. Adding the slack variable s_1 and s_2 produces the standard form ILP

$$\begin{aligned} \max x + y & \\ \text{Subject To} & \\ 2x + y + s_1 &= 10 \\ x + 3y + s_2 &= 12 \\ x, y, s_1, s_2 &\geq 0 \text{ and integer} \end{aligned}$$

Consider the LP relaxation of this ILP produced by dropping the constraint that the variables be integers. Create the simplex tableau for the basis $\{x, y\}$.

- (15 points) Recall the Gomory cut algorithm described in class. Given a row from the simplex tableau, this algorithm produces a *cut*, a new constraint that both eliminates the current optimal solution in the LP without eliminating any integer-feasible solutions.

  Run the Gomory cut algorithm by hand using the row from your simplex tableau corresponding to the basic variable x. What is the resulting cut? Sketch the feasible region of the LP relaxation and the cut you produced. Does the new cut result in an integer-optimal solution?

Problem 2: Stochastic Programming (60 points)

In this problem, you will take the role of Scott Smith, owner of the East End Brewing Company in Homewood.

Each week, you need to distribute kegs of beer to bars in various Pittsburgh neighborhoods. There are ten neighborhood bars that might demand kegs, and your goal is to minimize the total driving time on your route, where the truck starts and ends at your brewery in Homewood.

The file `times.txt` is an 11-by-11 comma-separated value file showing the travel times between the various neighborhoods in Pittsburgh, calculated using Google maps. The file shows the number of minutes it takes to get between the location listed on the row to the location listed on the column. Because of the vagaries of the Pittsburgh road network, this is not a symmetric matrix (e.g., it takes 14 minutes to get from Troy Hill to Oakland but only 12 minutes to get from Oakland to Troy Hill).

In this problem, whether or not a bar has demand for a keg is a random variable. In particular, we will explore the case where each bar independently has demand drawn from a Bernoulli distribution with probability p, so that the demand d_i of bar i is given by

$$\mathbb{P}(d_i = 1) = p \qquad \mathbb{P}(d_i = 0) = 1 - p$$

Let a *route* be a set of nodes $\vec{r} \equiv \{r_0, \ldots, r_k\}$ such that $r_0 = r_k =$ HomewoodBrewery and all the neighborhoods where $d_i = 1$ appear exactly once in $\vec{r}$ (neighborhoods where $d_i = 0$ may or may not appear in a route). Let R denote the set of all routes, and let $t(a, b)$ denote the amount of time it takes to get from a to b.

As stated, the problem is simply a traveling salesman problem, which you already know how to solve optimally by using Integer Programming techniques. Now, we introduce one additional complication into the problem: at neighborhood r_5, the demands for all the bars becomes known, and then the route can be re-optimized. This is now a *two-stage stochastic program with recourse*, where the two stages are before and after the information is revealed, and the recourse refers to the ability to change plans once the information is revealed. In particular, once the truck reaches r_5 you can re-optimize the route from that point to only visit the remaining bars that have demand in an optimal way.

To state your problem formally, you are looking to solve

$$\min_{\vec{r} \in R} \left[\sum_{i=1}^{5} t(r_{i-1}, r_i) + \mathbf{E}_{\vec{d}} \left(f(\vec{d}, \vec{r}) \right) \right]$$

where $f(\vec{d}, \vec{r})$ is the amount of time to traverse the remaining bars and return the truck to `HomewoodBrewery`, given the demand vector $\vec{d}$ and starting from r_5.

When $0 < p < 1$, the optimal policy is given by the first five neighborhoods to visit, and then by a function that maps realizations of the demand vector to paths through space. However, since you would never want to traverse a route that is not a minimal path through the remaining bars that have demand, for the purposes of the answers you give in this problem the optimal policy for $0 < p < 1$ can be characterized just by a path to the first 5 bars.

You will solve this problem by using MIPs. Solvers such as `CPLEX` are available on `linux.gp.cs.cmu.edu`, and there are also open-source solvers such as `lpsolve` you can run on your own machine. Probably the easiest way to input a MIP into a solver is to use the CPLEX LP file format (and, in particular, writing a meta-program to write the file for you). CPLEX LP files look like:

```
Maximize
 obj: x1 + 2 x2 + 3 x3 + x4
Subject To
  c1: - x1 + x2 + x3 + 10 x4 <= 20
  c2: x1 - 3 x2 + x3 <= 30
  c3: x2 - 3.5 x4 = 0
Bounds
  0 <= x1 <= 40
  2 <= x4 <= 3
General
  x4
End
```

These files can be directly imported into your MIP solver. A guide to this file format can be found at `goo.gl/RJWuw`. Another, less-powerful approach is to use the functionality of the `MATLAB` optimization toolbox. As a last resort, if you are unable to use `CPLEX` and have problems running `lpsolve` locally, please contact the TAs and we will work out a way for you to run your LP files.

- (10 points) Solve the Traveling Salesman Problem (i.e., solve the problem for the case where $p = 1$). What is the optimal route? What is the time to traverse that route?

  Recall from recitation that a polynomial-sized MIP representation of the TSP is the Miller-Tucker-Zemlin formulation (MTZ). If you did not attend recitation, Google is your friend.

- (10 points) In general stochastic programs with recourse, the space the random variables live in is too large to be approached precisely. Consequently, the second-stage expectation must be approximated, customarily through sampling. However, in this problem the randomness is limited to the demand vector, and there are only $2^{10} = 1024$ possible demand vectors. In the next parts of the problem, you will solve the problem both for a sampled approximation as well as exactly.

  Given a realization of N random variables drawn from the true demand distribution, write a MIP to solve for the optimal policy. Feel free to use abbreviated notation as long as your solution is clear. Hint: for any realization of the true demands, it is straightforward to find both the optimal route through space as well as the cost that arises from any choice of first five bars.

- (15 points) When $p = 0.5$, solve for the optimal policy when the expectation is calculated from 50 drawn realizations of the demand vector. What is the optimal policy? What is the expected time to traverse that route?

- (15 points) Solve for the true optimum of the problem when $p = 0.5$ by exhaustively enumerating all 1024 realizations of the demand vector. What is the optimally policy and expected time? How is your answer different from the case where $N = 50$? Why do you think that is?

- (10 points) Solve the problem optimally when $p = 0.01$, so that there is a only a very small chance of any bar demanding a keg. Did your optimal policy change? Why do you think this is?