ISyE6669 Deterministic Optimization
Team Project 1
Due On-Campus Nov 2, 2016; DL Nov 9, 2016

Instructions

1. Form a group of no more than four people. Email the names of your group members to the TA Ian (iherszterg@gatech.edu), and cc'd me (andy.sun@isye.gatech.edu) as soon as possible.

2. Collaboration between groups is not allowed.

3. A list of provided files:

   (a) Problem 1: **diet.colgen.partial.mos**, **diet.xls**.

   (b) Problem 2: **cs.Kantorovich.partial.mos**, **cs.colgen.partial.mos**, **kant1.dat**, **kant2.dat**, **cs1.dat**, **cs2.dat**.

   (c) Problem 3: **DW.partial.mos**

4. Each group submits one zipped file of all the codes (including mos files, dat files, .m files, and the final report) to T-Square. In the final report, you should clearly state your answers to all the questions, and print out the numerical results required by each problem. The report should also clearly specify which problems and in what way each member has contributed to.

5. Each on-campus group should also submit a physical copy of the final report and all the codes in class on Nov 2, 2016.

6. Each DL group should submit electronic copies of the final report and all the codes to T-Square on or before 11:55PM EST, Nov 9, 2016.

7. Some hints:

   - Sometimes, it's useful to first make the overall framework work, then divide into each subtask.
   - Divide and conquer: When one problem consists of different subproblems, work on each subproblem as a stand alone problem. Write separate code for the subproblems, and make sure they work before putting things together.
   - Problem 1 provides a nice exercise for constructing the overall framework of column generation. It may be helpful to work out Problem 1 before attacking Problems 2 and 3.
   - It is a good practice to have clear comments in your code.
   - Through this project, you will get a pretty decent working knowledge of column generation and Dantzig-Wolfe. The project requires a considerable amount of effort in understanding the algorithms as well as in coding. To be successful, all four members in the group need to work dilligently and collaboratively. Also start as early as you can.
   - Have fun!

Problem 1: Column Generation for the Diet Problem

In this problem, we will solve a large diet problem. There are 7035 kinds of food listed, and 30 nutrients. We want to find a diet that has the minimum level of cholesterol intake and at the same time satisfies all the nutritional requirement.

For each nutrient i, let m_i be the minimum daily intake of i and let M_i be the maximum daily intake of i. For each food j, let a_{ij} be the amount of nutrient i in food j. Let $chol_j$ be the amount of cholesterol in food j, and define variables x_j to be the amount of food j in the daily diet. Then, the formulation of the diet problem is:

$$\min \quad \sum_{j=1}^{7035} chol_j \cdot x_j \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{7035} a_{ij} x_j \leq M_i, \quad \forall i = 1, \ldots, 30 \tag{2}$$

$$\sum_{j=1}^{7035} a_{ij} x_j \geq m_i, \quad \forall i = 1, \ldots, 30 \tag{3}$$

$$x_j \geq 0, \forall j = 1, \ldots, 7035 \tag{4}$$

Suppose you wanted to solve it using the student (free) version of Xpress, which cannot handle 7035 variables. Apparently, you need to solve the problem in a more clever way. Column generation is an ideal choice. The attached file (**diet.colgen.partial.mos**) gives the general framework of column generation, but it is missing some crucial parts that you need to provide.

Questions:

1. Complete the construction of the column generation code and submit the mos file online and also submit a physical copy with the final report.
2. Print out the final solution from the code, and submit it in the final report.
3. Now, you want to find an optimal diet with the total calorie intake between 1800 cal and 2200 cal. Modify the data file diet.xls (the minimum and maximum levels in the calorie column), and resolve the above problem. Report the composition of optimal diet and the total calorie in the final report.

HINT:

1. getdual(name) is the Xpress function to retrieve the optimal dual variable (also called shadow price) of a constraint called "name". To store that shadow price, you need to save that value in the array given to you.
2. A general way to add a new column to a constraint or the objective function is: name += XXX, where name is the name of the constraint or the objective function, and XXX is the part of the constraint that comes with the new variable.

Problem 2: Different Formulations and Solution Methods for the Cutting Stock Problem

The cutting stock problem formulation that we learned in class is due to P. C. Gilmore and Ralph E. Gomory. They published this formulation in their 1961 paper "A linear programming approach to the cutting-stock problem", *Operations Research*, 8 (1961), 849-859. We call this formulation the Gilmore-Gomory formulation. A little history: Ralph E. Gomory is a renowned mathematician and a key figure in the development of theoretical understanding and computational methods for integer programming in the early days. He has also been very influential in bringing OR to the real world. He first worked at IBM as a research mathematician and later became IBM's Senior Vice President for Science and Technology. He helped IBM attain the best minds for research for over 20 years. He still maintains a blog in the Huffington Post on interesting topics such as technology development and industry research[1].

An alternative optimization formulation for the cutting stock problem was proposed by the Soviet mathematician and economist Leonid V. Kantorovich in 1939. His paper was published in English in 1960 ("Mathematical Methods of Planning and Organising Production" *Management Science*, 6 (1960), 366-422. You can get the paper online through Tech's library.) Later, Kantorovich won the Nobel Prize in Economics in 1975, shared with another pioneer of operations research Tjalling Koopmans, "for their contributions to the theory of optimal allocation of resources."

Kantorovich's formulation for the cutting stock problem is very different from the Gilmore-Gomory formulation. In this problem, we lead you through steps to formulate the Kantorovich formulation and implement in Xpress. Then, we ask you to implement Column Generation on the Gilmore-Gomory formulation. The purpose is to compare these two different formulations and solution strategies for solving the same cutting stock problem. Through this exercise, you will learn that the a more compact formulation of an optimization problem might not always be computationally easier to solve than a larger formulation. You will also see the power of column generation as a solution strategy to solve large-scale linear programs.

Problem 2.1: The Kantorovich Formulation: A Modelling Exercise

The Kantorovich formulation uses the following data and decision variables:

1. Data:
   - w_i, $i = 1, \ldots, m$: the width of small roll item i.
   - b_i, $i = 1, \ldots, m$: the demand for item i.
   - W: the width of the large rolls.
   - K: the total number of large rolls. (Note that this data is not required in the Gilmore-Gomory formulation.)
2. Decision variables:
   - y^k: if large roll k is cut then $y^k = 1$, otherwise $y^k = 0$, for $k = 1, \ldots, K$.
   - x_i^k: number of times that item i of width w_i is cut on the large roll k.

Now we lead you through the steps to formulate the Kantorovich model. Write down the following three constraints.

- The objective is given as to minimize the number of large rolls cut to satisfy demand:

$$\min \quad \sum_{k=1}^{K} y^k$$

[1] `http://www.huffingtonpost.com/ralph-gomory/`

- Constraint 1: We must satisfy demand with equality for each item $i = 1, \ldots, m$:

- Constraint 2: We cannot exceed the width of each large roll $k = 1, \ldots, K$:

- Constraint 3: The decision variables must also satisfy their bounds and integrality constraints:

Questions:

1. How many variables and constraints are there in this formulation? Comparing to the Gilmore-Gomory formulation we discussed in class, does the Kantorovich formulation have more variables?

2. Use the files **cs.Kantorovich.partial.mos** as a starting point. Finish the Xpress code and solve the problem using the data file **kant1.dat**. Print out the optimal solution in the final report.

3. Solve the Kantorovich formulation for the following instances **kant2.dat**, **cs1.dat**, **cs2.dat**. Notice that kant2.dat has only one more type of small roll than kant1.dat. You can manually terminate the solver after running 5 minutes on each instance (click on the red button in Xpress, then select the first option in the pop-up window "Accept the current MIP solution and continue execution if applicable."). Answer the following questions for each instance:

   (a) Did the solver terminate within 5 minutes?

   (b) How many branch-and-bound (BB) nodes are searched by the BB algorithm when the solver terminates?

   (c) What is the objective value of the best lower bound (denoted as Z_L) and the objective value of the best integer solution (denoted as Z_U) found?

   (d) What is the optimality gap, i.e. $(Z_U - Z_L)/Z_U$?

   (e) How many integer solutions are found by the BB algorithm?

   To answer these questions, click on the "Stats" tab on the right-hand side panel in Xpress. Look at the results after "Current node", "Best Bound", "Best solution", "Gap", "Status", and "Time". Lastly, relax the integrality constraints. Write your answers in the report.

Problem 2.2: The Gilmore-Gomory Formulation: Use Column Generation

You should encounter some difficulty with the Kantorovich formulation for the larger data set. In this part, we want to solve the Gilmore-Gomory formulation (exactly the same formulation we discussed in class) using column generation. The master problem is given below.

$$\begin{aligned} \min \quad & \sum_{j=1}^{N} x_j \\ \text{s.t.} \quad & \sum_{j=1}^{N} a_{ij} x_j = b_i, \qquad \forall i = 1, \ldots, m \quad \text{(Demand Constraints)} \\ & x_j \geq 0, \qquad \forall j = 1, \ldots, N. \end{aligned}$$

Attached file (**cs.colgen.partial.mos**) gives the general framework of the column generation algorithm. It also has detailed instructions on how to complete the code. The programming experience you gained in solving Problem 1, the diet problem, will be useful here. You need to do the following.

Questions:

1. Complete the column generation code. Submit the completed mos file online and also submit a physical copy with the final report.
2. Print out solution summary for each test instance (**kant1.dat**, **kant2.dat**, **cs1.dat**, **cs2.dat**), following the instructions in the **cs.colgen.partial.mos** file.
3. In your report, discuss the differences between the LP solutions and the integer solutions obtained in the three test instances, also the differences between the integer solutions of the two approaches (rounding and resolving). Which approach produces better solution?
4. For each instance of **kant1.dat**, **kant2.dat**, **cs1.dat**, **cs2.dat**, compare the best integer solutions obtained through the column generation algorithm and the best integer solutions obtained by solving the Kantorovich formulation. Which method gives the better solution for each instance?

HINT:

1. The Xpress code uses dynamic array to implement an array whose size can change dynamically. To create a new variable, use create(variable name).
2. Use ceil to round to the nearest larger integer.
3. You can use a similar way to add new columns to constraints and the objective function as hinted in problem 1.

Problem 3: Dantzig-Wolfe Decomposition

In this problem, you are asked to implement the Dantzig-Wolfe decomposition for the following linear optimization problem:

$$
\begin{aligned}
(MP_{\boldsymbol{x}}) \quad \min \quad & \boldsymbol{c}^\top \boldsymbol{x} && (5)\\
\text{s.t.} \quad & \boldsymbol{D}\boldsymbol{x} = \boldsymbol{b}_0 && (6)\\
& \boldsymbol{F}\boldsymbol{x} = \boldsymbol{b} && (7)\\
& \boldsymbol{x} \geq \boldsymbol{0}. && (8)
\end{aligned}
$$

Here (6) is the coupling constraint. (7)-(8) define a polyhedral P. Denote the extreme points of P as $\{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^N\}$. Each point $\boldsymbol{x} \in P$ can be written as a convex combination of these extreme points as

$$
\begin{aligned}
& \boldsymbol{x} = \sum_{i=1}^{N} \lambda_i \boldsymbol{x}^i \\
& \sum_{i=1}^{N} \lambda_i = 1 \\
& \lambda_i \geq 0 \qquad \forall i = 1, \ldots, N.
\end{aligned}
$$

Substitute this extreme point representation to $(MP_{\boldsymbol{x}})$, we have an equivalent problem $(MP_{\boldsymbol{\lambda}})$ in the $\boldsymbol{\lambda}$ variable as

$$
\begin{aligned}
(MP_{\boldsymbol{\lambda}}) \quad \min \quad & \sum_{i=1}^{N} \lambda_i \left(\boldsymbol{c}^\top \boldsymbol{x}^i\right) \\
\text{s.t.} \quad & \sum_{i=1}^{N} \lambda_i \left(\boldsymbol{D}\boldsymbol{x}^i\right) = \boldsymbol{b}_0 \\
& \sum_{i=1}^{N} \lambda_i = 1 \\
& \lambda_i \geq 0 \qquad \forall i = 1, \ldots, N.
\end{aligned}
$$

Now we can apply column generation to $(MP_{\boldsymbol{\lambda}})$ as discussed in class.

1. Start from a subset I of extreme points of the polyhedron P. Solve the following **restricted master problem**:

$$
\begin{aligned}
(RMP) \quad \min \quad & \sum_{i \in I} \lambda_i \left(\boldsymbol{c}^\top \boldsymbol{x}^i\right) \\
\text{s.t.} \quad & \sum_{i \in I} \lambda_i \left(\boldsymbol{D}\boldsymbol{x}^i\right) = \boldsymbol{b}_0 && (9)\\
& \sum_{i \in I} \lambda_i = 1 && (10)\\
& \lambda_i \geq 0 \qquad \forall i \in I.
\end{aligned}
$$

   Let $\hat{\boldsymbol{y}}$ be the optimal dual variable associated with the coupling constraint (9), and let $\hat{r}$ be the optimal dual variable associated with the convexity constraint (10).

2. Solve the **pricing problem**:

$$\begin{aligned} \hat{Z} = \min \quad & \left(\boldsymbol{c}^\top - \hat{\boldsymbol{y}}^\top \boldsymbol{D}\right) \boldsymbol{x} \\ \text{s.t.} \quad & \boldsymbol{F}\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0}. \end{aligned}$$

Denote the optimal solution of the above pricing problem as $\hat{\boldsymbol{x}}$. There are two possibilities:

(a) If $\hat{Z} - \hat{r} \geq 0$, then all the reduced costs are nonnegative. Terminate the column generation algorithm, and report the optimal solution.

(b) Otherwise, we add $\hat{\boldsymbol{x}}$ as a newly generated extreme point to I, and go to 1.

Question:

1. Implement the Dantzig-Wolfe decomposition code using the data in the following Question 2(a). Submitted the complete mos file both online and in the physical copy.

2. Do the following two experiments with your code:

(a) $\boldsymbol{c} = [-4, -1, -6, -3, -5], \boldsymbol{D} = [3, 2, 4, 3, 5], \boldsymbol{b}_0 = 25$, the polyhedron P is the five dimensional cube $1 \leq x_i \leq 2$ for $i = 1, ..., 5$. Start with two initial extreme points $(1, 2, 1, 1, 1)$ and $(2, 2, 2, 1, 1)$.

(b) $\boldsymbol{c} = [-4, -1, -6, -3, -5], \boldsymbol{D} = [3, 2, 4, 3, 5], \boldsymbol{b}_0 = 25$, the polyhedron P is the five dimensional cube $1 \leq x_i \leq 3$ for $i = 1, ..., 5$. Start with two initial extreme points $(1, 1, 1, 1, 1)$ and $(3, 3, 3, 3, 3)$.

For each experiment, print out the optimal solution of the (RMP) and the associated dual solution, also print out the minimum reduced cost and the new extreme points generated in each intermediate iteration of the algorithm. Of course, you also need to print out the final optimal solutions in $\boldsymbol{x}$ variable and $\boldsymbol{\lambda}$ variable, respectively.