Felipe F. Lorenci

Challenge - Data Scientist (Optimization and Prescriptive Analytics)

August 9th, 2022 | Centro de analytics

1 Introduction

This document contains the resolution of the challenge for the position of Data Scientist (Optimization and Prescriptive Analytics) at the company Raízen. The proposed problem statement is as follows:

"*You are a participant in a TV show, and you are running for the big prize. In this last challenge, you have in front of you a total of 40 containers. Each container contains up to 3 boxes, and those boxes contain some cylinders. The volume and weight of each cylinder is listed below. To win this challenge, you must choose 35 containers and use cylinders from the boxes in the chosen containers to reach a total volume of 5163.69 milliliters and a total weight of 18.844 kg. For any selected container, you may use as many boxes as you want (at least one for each chosen container). However, you can choose only one cylinder from each box. Which containers, boxes and cylinders will you choose? Is there more than one winner option? The table below presents all necessary data to support your decision. Besides the problem solution, please also send the mathematical formulation and your code (commented). Feel free to select the desired optimization approach. PuLP is a free and open-source solver that can handle this problem, but other options are also valid (Reference: Optimization Modeling in Python: PuLP, Gurobi, and CPLEX — by Opex Analytics — The Opex Analytics Blog — Medium).*"

The text above is accompanied by a table composed of the columns "Container", "Box", "Cylinder", "Cylinder weight (g)", "Cylinder volume (mL)" and "Density (g/mL)", which respectively represent the identification of the container in which a cylinder is stored, the identification of the box where the cylinder is inserted, the identification of the cylinder itself, and finally, its weight, volume, and density.

The remainder of this document is organized as follows: in the next section, we define the problem formally. In section three, we present our mathematical formulation. In the last section, we discuss the experimental tests and show the final results.

2 Problem definition

Before modeling the problem mathematically to solve it using a software solver, let's define it formally. This step helps us to have more clarity during the problem formulation, avoiding errors and reducing possible redundancies.

Problem 1 *Raízen's TV Show - the final challenge*
Input: *a tuple $\langle \mathcal{A}, \mathcal{B}, \mathcal{C}, B_a, C_b, \upsilon, \omega \rangle$, where:*

- *$\mathcal{A}$ is the set of containers. Each container is unique. In this particular case, the instance have 40 containers (i.e., $|\mathcal{A}| = 40$). A single container $a \in \mathcal{A}$ is identified as a;*
- *$\mathcal{B}$ is the set of all boxes. Each box is unique. The boxes are inside the containers. Each container contains up to 3 boxes. A single box $b \in \mathcal{B}$ is identified as b;*
- *$\mathcal{C}$ is the set of all cylinders. Each cylinder is unique. The cylinders are stored in the boxes. A single cylinder $c \in \mathcal{C}$ is identified as c;*
- *B_a represents the set of boxes indexed by containers (i.e., B_a gives us the boxes which are inside the container a). We define as $|B_a|$ the number of boxes contained in the container a;*
- *C_b represents the set of cylinders indexed by boxes (i.e., C_b gives us the cylinders which are inside the box b);*
- *$\upsilon : \mathcal{C} \to \mathbb{R}$ is a function that returns the volume of cylinder c;*
- *$\omega : \mathcal{C} \to \mathbb{R}$ is a function that returns the weight of cylinder c.*

Output: *a subset $\mathcal{S} \subset \mathcal{A}$ (selected containers), such that $|\mathcal{S}| = 35$, a subset $\mathcal{S}_2 \subset \mathcal{B}$ (selected boxes), and a subset $\mathcal{S}_3 \subset \mathcal{C}$ (selected cylinders), such that $\sum_{c \in \mathcal{S}_3} \upsilon(c) = 5163.69$ and $\sum_{c \in \mathcal{S}_3} \omega(c) = 18844$. For any selected container we can use as many boxes as we want (at least one for each chosen container). For each selected box, we can choose only one cylinder.*

Once the problem is defined, we present our mathematical model in the next section.

3 Mathematical formulation

We modeled the problem 1 as an integer linear program (ILP). From now on, we consider an instance $I = \langle \mathcal{A}, \mathcal{B}, \mathcal{C}, B_a, C_b, \upsilon, \omega \rangle$ from the problem. We emphasize that each element of the problem (containers, boxes, and cylinders) is treated as a unique entity during the formulation process.

Our model is defined over three sets of binary variables. The first set controls the choice of containers:

$$x_a = \begin{cases} 1, & \text{if the container } a \text{ is chosen} \\ 0, & \text{otherwise.} \end{cases}$$

The second set of variables, determines which boxes are selected:

$$y_b = \begin{cases} 1, & \text{if the box } b \text{ is chosen} \\ 0, & \text{otherwise.} \end{cases}$$

The last of variables, establishes if a cylinder is selected or not:

$$z_c = \begin{cases} 1, & \text{if the cylinder } c \text{ is chosen} \\ 0, & \text{otherwise.} \end{cases}$$

The problem does not have an optimization sense, i.e., we are not searching for a specific maximum/minimum value at the objective function. According to the rules, what is demanded of us, is just a feasible solution from the search space, which is delimited by the constraints. So, we can define an arbitrary objective function as follows:

$$min \quad 0$$

To guarantee that there are exactly 35 selected containers, we define the following set of constraints:

$$\sum_{a \in \mathcal{A}} x_a = 35 \tag{1}$$

The sum of cylinders' volumes and cylinder's weights should be equal to 5163.69 (ml) and 18844 (g), respectively. So, we state the next two groups of constraints:

$$\sum_{c \in \mathcal{C}} \upsilon(c) z_c = 5163.69 \tag{2}$$

$$\sum_{c \in \mathcal{C}} \omega(c) z_c = 18844 \tag{3}$$

To control which boxes are selected, we implemented the following two constraint groups:

$$\sum_{b \in B_a} y_b \leq x_a |B_a| \quad \forall a \in \mathcal{A} \tag{4}$$

$$\sum_{b \in B_a} y_b \geq x_a \quad \forall a \in \mathcal{A} \tag{5}$$

The first one (constraint group 4), guarantee that no boxes from the container a will be chosen if this container is not in the solution ($\sum_{b \in B_a} y_b \leq 0$). Note that if a is selected, the maximum number of boxes selected in this container is equal to $|B_a|$ (i.e., the number of boxes in the container a). The second one (constraint group 5) works guaranteeing that whether a container a is selected, at least one box from this container also should be selected.

The last set of constraints guarantee that just one cylinder will be selected for each selected box (note that it also excludes from solution, cylinders contained in boxes that are not in the solution):

$$\sum_{c \in C_b} z_c = y_b \quad \forall b \in \mathcal{B} \tag{6}$$

The above ideas lead us at the following model:

Model 1 *Integer linear program for instance $\langle \mathcal{A}, \mathcal{B}, \mathcal{C}, B_a, C_b, \upsilon, \omega \rangle$ of Raízen's TV Show - the final challenge problem:*

$$\begin{aligned} \min \quad & 0 \\ s.t. \; constraints \; & (1) - (6) \\ x_a, \; y_b, \; & z_c \in \{0, 1\} \end{aligned}$$

The above model works very well for finding a solution to the problem (i.e., a winning option for the challenge) in low computational time. However, there is still an open question: is there more than one winning option for the TV show challenge?

One way to get such an answer is through modern (paid) solvers, which have functions like ”PoolSearchMode” (Gurobi), where we can explore the whole search space with a lot of flexibility. Another way, using free softwares, is to manually add cuts to the formulation in a dynamic way, prohibiting the solver from returning in future runs the same solution already returned in past runs.

Thus, since we are using free software, our methodology consists of adding a cut to the formulation after each call to the solver, so this inequality prohibits the software from returning a solution equal to the last one found. These steps are repeated until some limit of iterations is exhausted, or all possible solutions are returned. The proposed cut is as follows:

$$\sum_{c \in \mathcal{S}_3} z_c \leq |\mathcal{S}_3| - 1$$

Then, in the next section, we discuss about the experiments and results.

4 Experiments and results

The algorithms were implemented in Python Language (version 3.8.10) and the mathematical formulation was modeled through the library PuLP (version 2.5.0). The mixed-integer program (MIP) solver called by PuLP is the CBC (COIN-OR Branch and Cut). All the experiments were executed in an AMD Ryzen 7 (3700U) with 5.7 GB RAM and Ubuntu 20.04.4 LTS operational system.

The complete implementation is available at the main branch from the Github repository (`https://github.com/lorencifelipe/Raizen_Analytics`).

The first step was to extract data from the received *.xls* document and check for missing data, possible outliers or inconsistencies. After this first analysis, the ready dataset was saved to main branch as the “data.csv” file. You will notice that some columns have different names, that were changed to simplify the readings.

We chose to encapsulate all the methods relative to the problem to a class “Problem” (file “problem.py”). All of these methods are commented at the code. The main algorithm is “main.py”, which calls the methods implemented in the Problem Class and executes the approach presented in the last section. All solutions delivered by the solver were tested for all the problem constraints in execution time.

Thus, answering the main question, “*Which containers, boxes and cylinders will you choose?*”: a possible solution is in table 1, where the first column represents the container, the second column represents the selected boxes from the respective container, and the last column includes which cylinder from the respective box is added to the solution.

About the second question, “*Is there more than one winner option?*”, the answer is yes. We defined the iteration limit of our algorithm to 1000, and in all of theses iterations, the

Table 1: A feasible solution to the problem

Container	Boxes	Cylinders
B	LB_1, LB_2	LB_1: 2 LB_2: 1
C	LB_1, LB_2	LB_1: 1 LB_2: 1
D	LB_2	LB_2: 2
F	LB_1	LB_1: 10
G	LB_1	LB_1: 2
H	LB_1, LB_2	LB_1: 6 LB_2: 2
I	LB_1	LB_1: 1
J	LB_1	LB_1: 4
K	LB_1	LB_1: 3
L	LB_1	LB_1: 3
M	LB_1, LB_2	LB_1: 14 LB_2: 2
N	LB_1	LB_1: 15
O	LB_1	LB_1: 10
Q	LB_1	LB_1: 3
R	LB_1	LB_1: 6
S	LB_1	LB_1: 9
T	LB_1, LB_2	LB_1: 1 LB_2: 5
U	LB_1, LB_2	LB_1: 1 LB_2: 1
X	LB_1	LB_1: 6
Y	LB_1, LB_2	LB_1: 4 LB_2: 1
Z	LB_1, LB_2	LB_1: 6 LB_2: 1
AA	LB_1	LB_1: 1
AB	LB_1	LB_1: 1
AC	LB_1	LB_1: 12
AD	LB_1, LB_2	LB_1: 2 LB_2: 3
AF	LB_1	LB_1: 6
AG	LB_1	LB_1: 13
AI	LB_1	LB_1: 7
AJ	LB_1	LB_1: 4
AK	LB_1	LB_1: 3
AL	LB_2	LB_2: 1
AM	LB_1, LB_2	LB_1: 6 LB_2: 1
AN	LB_2	LB_2: 2
AO	LB_2	LB_2: 1
AQ	LB_1	LB_1: 3

algorithm found a different feasible solution (solutions are stored at the folder “results” at the Github repository).