INFO-F424 – Combinatorial Optimization Project

Renaud Chicoisne
renaud.chicoisne@ulb.be

Cristian Aguayo
cristian.aguayo.quintana@ulb.be

In this project, we focus on the *Assortment Planning Problem* (AP), where a retailer wants to determine which products it has to propose to its customers in order to maximize its expected revenue. More precisely, consider a set of products $\mathcal{I} = \{1, \dots, n\}$ that the retailer can propose to its customers, and selling product i generates a net revenue of $r_i > 0$ for him. We assume that the products are sorted in decreasing order of revenue, i.e. $r_1 > r_2 > \cdots > r_n > 0$, and that each product i will be purchased according to a particular probability that depends on:

1. The *mean utilities* $(\mu_j)_{j\in\mathcal{I}}$ for the customers when they buy product $j \in \mathcal{I}$,
2. The set of alternatives $\mathcal{S}$ made available to the customers.

These probabilities come from a discrete choice model called *multinomial logit*, and can be written as follows:

$$P_i(\mathcal{S}) = \frac{e^{\mu_i}}{e^{\mu_0} + \sum_{j\in\mathcal{S}} e^{\mu_j}} = \frac{e^{\mu_i}}{1 + \sum_{j\in\mathcal{S}} e^{\mu_j}}, \quad \forall i \in \mathcal{I} \cup \{0\}$$

where $\mu_0 = 0$ represents the utility - for the customers - of buying nothing. As we will see later on, it is convenient to assume that selling nothing *does come with a revenue* $r_0 \geqslant 0$ (that is, however, usually zero). With all of the above, the problem can be posed as the following *combinatorial optimization problem*:

$$\text{(AP)} \quad \max_{\mathcal{S}\subseteq\mathcal{I}} \left\{ r_0 \cdot P_0(\mathcal{S}) + \sum_{i\in\mathcal{S}} r_i \cdot P_i(\mathcal{S}) \right\}$$

1 From Combinatorial Optimization to Linear Programming

1. Show that AP can be rewritten as the following integer programming problem:

$$\text{(AP-IP)} \quad \max_{x\in\{0,1\}^n} \quad \frac{r_0 + \sum_{i=1}^n x_i r_i e^{\mu_i}}{1 + \sum_{i=1}^n x_i e^{\mu_i}}.$$

   Explain why this formulation is valid for the Assortment Planning Problem (variables, constraints, objective function), and why this formulation is nonlinear.

2. Considering the following change of variables:

$$y_0 := \frac{1}{1 + \sum_{j=1}^n x_j e^{\mu_j}}$$
$$y_i := \frac{x_i e^{\mu_i}}{1 + \sum_{j=1}^n x_j e^{\mu_j}}, \qquad \forall i \in \mathcal{I},$$

prove that AP-IP can be rewritten as the following problem:

$$\begin{aligned}
\text{(AP-IPL)} \quad \max_{y, y_0 \geqslant 0} \quad & r_0 y_0 + \sum_{i=1}^{n} r_i y_i \\
\text{s.t.} \quad & y_0 + \sum_{i=1}^{n} y_i = 1 \\
& y_i \in \{0, y_0 e^{\mu_i}\}, \qquad \forall i \in \mathcal{I},
\end{aligned}$$

where $y_i \in \{0, y_0 e^{\mu_i}\}$ means that y_i must take either 0 or $y_0 e^{\mu_i}$ as a value.

3. Consider the continuous relaxation of AP-IPL:

$$\begin{aligned}
\text{(AP-L)} \quad \max_{y, y_0 \geqslant 0} \quad & r_0 y_0 + \sum_{i=1}^{n} r_i y_i \\
\text{s.t.} \quad & y_0 + \sum_{i=1}^{n} y_i = 1 \\
& y_i \leqslant y_0 e^{\mu_i}, \qquad \forall i \in \mathcal{I}.
\end{aligned}$$

Prove that its linear programming dual is:

$$\begin{aligned}
\text{(AP-LD)} \quad \min_{\pi \geqslant 0, \pi_0 \in \mathbb{R}} \quad & \pi_0 \\
\text{s.t.} \quad & \pi_0 - \sum_{i=1}^{n} \pi_i e^{\mu_i} \geqslant r_0 \\
& \pi_0 + \pi_i \geqslant r_i, \qquad \forall i \in \mathcal{I},
\end{aligned}$$

and explain the relationships between dual (resp. primal) variables and primal (resp. dual) constraints.

2 Ideal Formulation and a Greedy Algorithm

We now show that, with the help of AP-L, we can solve AP in polynomial time.

1. For any $k \in \mathcal{I}$, consider the following solution:

$$\begin{aligned}
y_0^k &:= \frac{1}{1 + \sum_{j=1}^{k} e^{\mu_j}} \\
y_i^k &:= \begin{cases} \frac{e^{\mu_i}}{1 + \sum_{j=1}^{k} e^{\mu_j}} & \text{if } i \leqslant k \\ 0 & \text{otherwise.} \end{cases}, \qquad \forall i \in \mathcal{I}.
\end{aligned}$$

   (a) Show that (y^k, y_0^k) is feasible for AP-L. What is its objective value?

   (b) Using the change of variables in (1.2), prove that the associated x^k is integer.

   (c) Show that (y^k, y_0^k) is a strictly better solution than (y^{k-1}, y_0^{k-1}) if and only if:

$$r_k > \frac{r_0 + \sum_{j=1}^{k-1} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k-1} e^{\mu_j}}.$$

   (d) Prove that there is no $k \in \mathcal{I}$ such that

$$r_k \leqslant \frac{r_0 + \sum_{j=1}^{k-1} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k-1} e^{\mu_j}} \quad \text{and} \quad r_{k+1} > \frac{r_0 + \sum_{j=1}^{k} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k} e^{\mu_j}}$$

   at the same time. What does the graph of the function $k \to r^\top y^k$ look like?

(e) Deduce that there is exactly one $k^* \in \mathcal{I}$ such that

$$r_1 > \cdots > r_{k^*} > \frac{r_0 + \sum_{j=1}^{k^*-1} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k^*-1} e^{\mu_j}} \quad \text{and} \quad \frac{r_0 + \sum_{j=1}^{k^*} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k^*} e^{\mu_j}} \geqslant r_{k^*+1} > \cdots > r_n$$

(f) Show that we cannot have $r_{k^*} < \frac{r_0 + \sum_{j=1}^{k^*} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k^*} e^{\mu_j}}$, and deduce that we have

$$r_1 > \cdots > r_{k^*} \geqslant \frac{r_0 + \sum_{j=1}^{k^*} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k^*} e^{\mu_j}} \geqslant r_{k^*+1} > \cdots > r_n$$

2. For any $k \in \mathcal{I}$, consider the following solution:

$$\pi_0^k := \frac{r_0 + \sum_{j=1}^{k} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k} e^{\mu_j}}$$

$$\pi_i^k := \begin{cases} r_i - \frac{r_0 + \sum_{j=1}^{k} e^{\mu_j} r_j}{1 + \sum_{j=1}^{k} e^{\mu_j}} & \text{if } i \leqslant k \\ 0 & \text{otherwise.} \end{cases}, \qquad \forall i \in \mathcal{I}.$$

(a) What is its objective value? In which conditions is (π^k, π_0^k) feasible for AP-LD? Is there such a k?

(b) Given $k \in \mathcal{I}$, if (π^k, π_0^k) is feasible for AP-LD, prove that $\mathcal{S} := \{1, \cdots, k\}$ is optimal for AP. Is AP-L an ideal formulation for AP?

(c) Propose a polynomial time algorithm that solves AP. What is its worst-time complexity?

3 A more practical model

We now suppose that the retailer can only offer up to p products to its customers.

1. Starting from the AP-L formulation, explain why the new model admits the following Mixed Integer Linear Program formulation (variables, constraints, objective function):

$$\begin{aligned} \text{(APC-MILP)} \quad \max_{y, y_0 \geqslant 0, z \in \{0,1\}^n} \quad & r_0 y_0 + \sum_{i=1}^{n} r_i y_i \\ \text{s.t.} \quad & y_0 + \sum_{i=1}^{n} y_i = 1 \\ & y_i \leqslant y_0 e^{\mu_i}, \qquad \forall i \in \mathcal{I} \\ & y \leqslant z, \quad \sum_{i=1}^{n} z_i \leqslant p \end{aligned}$$

2. Is the previous algorithm still working for the APC-MILP problem? Implement APC-MILP with one of the allowed solvers, and compare the results for $p \in \{1, n/5, n/2, n\}$.

3. We now propose to use a Lagrangean Relaxation framework to (approximately) solve the practical model. Coming back to AP-IP, it is easy to cast yet another version of the practical model:

$$\begin{aligned} \text{(APC-IP)} \quad \max_{x \in \{0,1\}^n} \quad & \frac{r_0 + \sum_{i=1}^{n} x_i r_i e^{\mu_i}}{1 + \sum_{i=1}^{n} x_i e^{\mu_i}} \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i \leqslant p, \end{aligned}$$

on which we will apply a Lagrangian algorithm.

(a) **Pricing:** after noticing that the constraint $\sum_{i=1}^{n} x_i \leqslant p$ is equivalent to

$$\frac{\sum_{i=1}^{n} x_i}{1+\sum_{i=1}^{n} x_i e^{\mu_i}} \leqslant \frac{p}{1+\sum_{i=1}^{n} x_i e^{\mu_i}},$$

prove that the pricing problem with a penalization $\lambda \geqslant 0$ can be recast as:

$$\begin{aligned} \text{(AP-L)}(\lambda) \quad \omega(\lambda) := \max_{y, y_0 \geqslant 0} \quad & r_0(\lambda) y_0 + \sum_{i=1}^{n} r_i(\lambda) y_i \\ \text{s.t.} \quad & y_0 + \sum_{i=1}^{n} y_i = 1 \\ & y_i \leqslant y_0 e^{\mu_i}, \qquad \forall i \in \mathcal{I}. \end{aligned}$$

Specify the values of $r_0(\lambda)$ and $r_i(\lambda)$ in function of r_0, λ, p, the revenues r_j and the utilities μ_j.

(b) Recall that we want to find the best dual bound, i.e. solve the following univariate problem:

$$\min_{\lambda \geqslant 0} \quad \omega(\lambda),$$

and that the function $\lambda \to \omega(\lambda)$ is convex and piecewise linear. Design a binary search to find the optimal λ^*. **Hint:** there is an optimal λ^* that is no greater than $\max\left\{0, \frac{r_1 - r_0}{p}\right\}$ (why?).

(c) **Heuristic:** whenever some λ provides a feasible solution for our practical problem, we compare it with the best solution so far and keep the best one.

(d) Implement the Lagrangean algorithm and plot a graph of the best bounds obtained so far (primal and dual). Granted that $r_1 > r_0$, prove that solving the lagrangean dual at ϵ precision requires the solution of a number of assortment planning problems in

$$O\left(\log_2\left(\frac{r_1 - r_0}{p\epsilon}\right)\right).$$

4 Implementation and evaluation of models and algorithms

Test the proposed models, as well as the algorithms proposed by you on different instance sizes on the $m = 100$ instances provided. The `size` of the instances is defined by the number of products:

- `small`: $n = 10$ products
- `medium`: $n = 5.000$ products

Each set of instances consists of 2 `.csv` files:

- `size-mu.csv`: A $(n+1) \times m$ matrix with μ_i values
- `size-r.csv`: A $(n+1) \times m$ matrix with r_i values

You are also asked to test your models with a large instance of $n = 1.000.000$ products, which you should generate following the same structure as the data you were given. Assume:

$$\mu_0 = 0 \qquad r_0 = 0 \qquad \mu_i \sim U[0,1] \qquad r_i \sim U[0,1]$$

When generating this instance you should either save the values in a .csv file like the one we give you, or set the random number generator seed so as not to produce different instances each time you run your code.
For each instance size and each model/algorithm, report the minimum, average and maximum optimal values, as well as the minimum, maximum and average solution times.

Guidelines and Evaluation

Guidelines

- The project must be carried out by groups of no more than three people.
- The required models and algorithms must be implemented in `python` or `julia` along with the solver `Gurobi` when necessary. To use `Gurobi`, you have to use the `gurobipy` package for `python` and the `JuMP` package for `julia`. You can use other packages to read and manage files.
- The implementations must be done in **one and only one** of the mentioned programming languages.
- You are asked to prepare a paper-like report which may include:
  - An introduction to the Assortment Planning Problem including appropriate references
  - The answers to the theoretical questions presented in this statement
  - Results and resolution times obtained for each model/algorithm, and discussion/conclusions about the different results.

Evaluation

- You must submit the report in `.pdf` format, as well as the `.py` or `.jl` files used to solve the problem along with their respective instructions for use. Reports made on `jupyter notebook` (`.ipynb` files) are also welcome.
- The report must be written in english and it must contain the names and last names of the members of the group, and their respective ULB ID numbers.
- Please **DON'T** upload the instance files. Just clarifiy in which directory should the instance flies must be placed to run your code.
- Verify that your codes work correctly before submitting them.
- The deadline is on June 2nd, 2024 at 23:59.
- Any disrespect of project and/or submission guidelines will lead to penalization.