Use Case Name Formulation

Author(s): Name of the author(s)
Month, Year.

The statement of the use case is on Mip Wise's website: www.mipwise.com/use-cases/use-case-name.

Note to authors:

Besides conciseness and clarity, we want our documentation and code to be as clean and consistent as possible across all use cases. This way we ensure a pleasant experience for the readers, facilitate communication between collaborators and cut down designing time. Therefore, we strongly encourage you to adopt the *4C-approach*.

> 4C: *Clean, Clear, Concise, and Consistent.*

It's true that being clear and concise at the same time can be a challenge. But that's one aspect that often distinguishes great work from the rest.

To be consistent, we use in the implementation the exact same notation adopted in the formulation. And we try as much as possible to stick to some conventions:

- **Set of indices** are denoted with a single capital letters, preferably I, J, K, L, and T.
- **Indices** are denoted with lower case letters and, whenever possible, the same letter of the set it comes from. That way we can be more consise and simply write $\sum_i x_i$ instead of $\sum_{i \in I} x_i$, for example.
- **Input parameters** are denoted with lower case letters, making suggestive choices whenever possible. For example, pc_i and sc_i for production cost and storage cost of product i, p_t for penalty in period t, sl_j and su_j for storage lower and upper bound at location j, and d_t for demand in period t.
- **Decision variables** are denoted with lower case letters, preferably x, y, z, and w. If more letters are needed, then we may also use u and v. Alternately, we may combine two letters, such as xa and wb, where the first letter denotes an already defined decision variable. This way we can easily distinguish decision variables from input parameters when reading a formulation.

Input Data Model

The input data model can be split into two sections, one that defines the set of indices and one that defines the input parameters of the model. When implementing the optimization model in Python, we typically declared set of indices as lists (to preserve the order of its elements and facilitate slicing) and input parameters as dictionaries.

Set of indices

- I: Set of products.
- J: Set of locations.
- T: Set of time periods.

Parameters

It's recommended to specify the unit of measure of every quantity parameter defined to avoid conversion mistakes.

- c_{jt}: Cost (dollar/period) to operate Location j in Period t.
- d_{it}: Demand (Kg) of Product i in Period t.

Decision Variables

It's also a good practice to specify the unit measure of decision variables whenever it makes sense.

- x_{ijt}: Amount (Kg) of Product i to be produced in Location j in Period t.
- z_{jt}: Equals 1 if Location j is operational in Perido t, 0 otherwise.

Constraints

We typically write a short description of each set of constraints and, when appropriate, add some additional comments right after the constraint to help the reader understand the formulation.

- Demand for Product i must be met:

$$\sum_{j} x_{ijt} \geq d_{it}, \quad \forall i, t.$$

- There is production in Location j in Period t if and only if Location j is operational in Period t:

$$\sum_{i} x_{ijt} \leq M \cdot z_{jt}, \quad \forall j, t.$$

  The constant M is an upper bound on the production of Location j in Period t.

Objective

It's often helpful to provide a short description of the objective function, especially if it has multiple components.

The goal is to minimize the total operation cost:

$$\min \sum_{jt} c_{jt} \cdot z_{jt}.$$

Final Formulation

It may also be helpful for the reader to see the full formulation together like below.

$$
\begin{aligned}
\min \quad & \textstyle\sum_{jt} c_{jt} \cdot z_{jt} \\
\text{s.t.} \quad & \textstyle\sum_{j} x_{ijt} \geq d_{it}, \quad \forall i, t, \\
& \textstyle\sum_{i} x_{ijt} \leq M \cdot z_{jt}, \quad \forall j, t, \\
& x_{ijt} \geq 0, \quad \forall i, j, t, \\
& z_{jt} \in \{0, 1\}, \quad \forall j, t.
\end{aligned}
$$