Very_cool_project.mzn

Valerio Costa, Luca Domeniconi, Claudia Maiolino, Tian Cheng Xia
{valerio.costa, luca.domeniconi5, claudia.maiolino, tiancheng.xia}@studio.unibo.it

1 Introduction

2 CP model

2.1 Decision variables

2.2 Objective function

2.3 Constraints

2.4 Validation

3 SAT model

The resolution of the problem is approached following two models.

- **Unified model**: based on the definition of two decision variables `assignments` and `paths`.
- **Matrix model**: based on the definition of one decision variable X.

Some modifications were applied to those model in order to visualize eventual differences in terms of performances.

The discussion is going to be much more focused on the Unified model just to mantain a logical thread within the explication of models used for other solvers.

3.1 Unified Model

General Definition The Unified model aims to find the correct assignment to both decision variables, in such a way that it satisfy all constraints.
The idea behind this kind of model is based on the separation of the two task just specified in one.

Original Workflow The original workflow follows the next steps:

1. find satisfying values for the `assignments` variable, else end the algorithm
2. find satisfying values for the `paths` variable, else proceed to step 4
3. repeat step 2 to find a new optimized solution,
4. repeat step 1 just to find another assignment

New Workflow The new algorithm simply tries to optimize the assignment to both variable.

Pro The model was designed to improve certain intrinsic problems of the definition of a problem through SAT:

- **Specificity of constraints**: constraining much more the assignments of the two variable guarantes to mantain lower width of exploration of the resolution tree.

Contro The model falls into a few issues such:

- **Dimension**: being based on two decision variables, the dimension of the problem scale exponentially with them. [1]

3.1.1 Decision variables

- `assignments`: $n \times m$ boolean matrix

$$\texttt{assignments}[p,\ c] = 1 \tag{1}$$

$$\texttt{if courier } c \texttt{ delivers pack } p$$

- `paths`: $m \times (n+1) \times (n+1)$ boolean matrix

$$\texttt{paths}[c,\ loc_1,\ loc_2] = 1 \tag{2}$$

$$\texttt{if courier } c \texttt{ moves from location } loc_1 \texttt{ to } loc_2$$

- `u`: $m \times n \times n$ boolean matrix[2]

$$\texttt{u}[c,\ p_1,\ p_2] = 1 \tag{3}$$

$$\texttt{if in the courier } c \texttt{ path the node } p_1 \texttt{ is associated to value } p_2$$

3.1.2 Objective function

The model aims to minimize the maximum distance travelled by any courier. The objective function can estimated as the maximum between the total distances of each courier:

$$\max_{c \in \{1,\ldots,m\}} \sum_{loc_1=1}^{n+1} \sum_{loc_2=1}^{n+1} \texttt{D}[loc_1,\ loc_2] * \texttt{paths}[c,\ loc_1,\ loc_2] \tag{4}$$

[1] The scaling dimension of the problem implies higher needs of time to build the model.
[2] The definition of the `u` variable is useful for the MTZ formulation.

A lower bound is set to costrain the objective during the resolution phase.

$$\max_{p\in\{1,\ldots,n\}} \texttt{D[}n+1\texttt{, }p\texttt{] + D[}p\texttt{, }n+1\texttt{]} \tag{5}$$

Its value is hypotesized to be the maximum between the minimum distance paths that a courier can travel. And this can be computed as the distance needed for a courier to reach a certain pack and return to the depot.

3.1.3 Constraints

Assignment related constraints

- Capacity constraint:
  - Sum of sizes of packs delivered by a singular courier must be under its load limit

$$\forall c \in \{1\ldots m\}: \quad \sum_{p=1}^{n} \texttt{assignments[}p\texttt{, }c\texttt{] * sizes[}p\texttt{]} \leq \texttt{loads[}c\texttt{]} \tag{6}$$

  - Each pack must be delivered only by a courier

$$\forall p \in \{1,\ldots,n\} : \exists! c \in \{1,\ldots,m\} \quad \texttt{s.t.} \quad \texttt{assignments[}p\texttt{, }c\texttt{]} = \texttt{1} \tag{7}$$

Path related constraints

- General path constraints
  - If courier delivers at least one package, there must exist a destination from DEPOT[3] with true value, else it can stay in DEPOT

$$\forall c \in \{1\ldots m\} : \forall p \in \{1\ldots n\} : \left\{ \begin{array}{ll} \texttt{NOT(paths[}c\texttt{, DEPOT, DEPOT])} & \texttt{iff } \sum_{p=1}^{n} \texttt{assignments[}p\texttt{, }c \\ \texttt{paths[}c\texttt{, DEPOT, DEPOT]} & \texttt{iff } \sum_{p=1}^{n} \texttt{assignments[}p\texttt{, }c \end{array} \right. \tag{8}$$

  - If courier delivers pack p, its destination must be different from p

$$\forall c \in \{1\ldots m\} : \forall p \in \{1\ldots n\} : \left\{ \begin{array}{ll} \texttt{NOT(paths[}c\texttt{, }p\texttt{, }p\texttt{])} & \texttt{iff assignments[}p\texttt{, }c\texttt{]} \\ \texttt{paths[}c\texttt{, }p\texttt{, }p\texttt{]} & \texttt{iff NOT(assignments[}p\texttt{, }c\texttt{])} \end{array} \right\} \tag{9}$$

  - Each pack must be delivered by a single courier only once

$$\forall c \in \{1\ldots m\} : \forall loc_1 \in \{1\ldots n{+}1\} : \quad \sum_{loc_2=1}^{n+1} \texttt{paths[}c\texttt{, }loc_1\texttt{, }loc_2\texttt{]} = \texttt{1} \tag{10}$$

---

[3]DEPOT is the original position, hypotesized as `n+1`.

- Subcircuit constraints
  - Only a single courier must deliver pack in location loc_2 from loc_1

$$\forall c \in \{1 \dots m\} : \forall loc_2 \in \{1 \dots n{+}1\} : \quad \sum_{loc_1=1}^{n+1} \texttt{paths[}c\texttt{, } loc_1\texttt{, } loc_2\texttt{] = 1} \tag{11}$$

  - Subtour elimination
    - u relative to first pack p of courier c path must have value $= 1$

$$\forall c \in \{1 \dots m\} : \forall p \in \{1 \dots n\} : \quad \texttt{iff} \ \ \texttt{paths[}c\texttt{, DEPOT, } p\texttt{]} \ \ \texttt{then} \ \ \texttt{u[}c\texttt{, } p\texttt{, 1]} \tag{12}$$

    - $u_j \geq u_i + 1$

$$\forall c \in \{1, \dots, m\} : \ \forall i, j, k \in \{1, \dots, n\} : \quad \texttt{paths}[c, i, j] \wedge \texttt{u}[c, i, k] \Longrightarrow \sum_{l=k+1}^{n} \texttt{u}[c, j, l] = 1 \tag{13}$$

    - Exatcly one true value for each $\texttt{u[}c, p, :\texttt{]}$ vector

$$\forall c \in 1 \dots m : \forall p_1 \in 1 \dots n : \quad \sum_{p_2=1}^{n} \texttt{u[}c\texttt{, } p_1\texttt{, } p_2\texttt{] = 1} \tag{14}$$

    - MTZ formulation constraint:

$$u_i - u_j + 1 \leq (n-1) * (1 - paths[c, i, j])$$
$$\forall c \in \{1, \dots, m\}, \ \forall i, j, k_1, k_2 \in \{1, \dots, n\} : \tag{15}$$

$$\texttt{u}[c, i, k_1] \wedge \texttt{u}[c, j, k_2] \Longrightarrow (k_1 - k_2 + 1) \leq (n-1) \cdot (1 - \texttt{paths}[c, i, j])$$

Symmetry breaking constraint A possible way to reduce tree exploration is to introduce symmetry breaking constraints. One of them in this case can consist in constraining the order of assignments of packs between couriers with the same amount of load capacity.

But the experimentation didn't lead to major improvements.

3.2 Validation

Experimental design

Some modifications were applied to the basic model in order to visualize eventuale differences in performances. The original Unified Model was modified in three different versions:

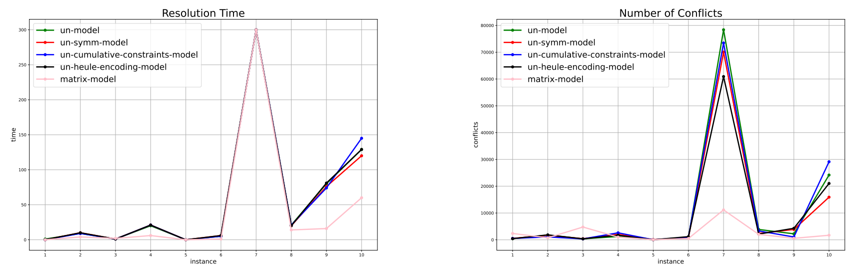
- Unified Model with Symmetry Breaking Constraint
- Unified Model with Cumulative Constraint Application
- Unified Model with Heule Encoding approach for `at_most_one()`

Experimental results

As we can notice from the following table the performances were not very much different from the basic model. Only in the first 10 instances it has been possible to rach at least a suboptimal solution, while for the remaining ones the construction of the model required too much time causing the exceeding of the timeout limit.

Id	un-model	un-symm-model	un-cum-constr-model	un-heule-enc-model	matrix-model
1	**14**	**14**	**14**	**14**	**14**
2	**226**	**226**	**226**	**226**	**226**
3	**12**	**12**	**12**	**12**	**12**
4	**220**	**220**	**220**	**220**	**220**
5	**206**	**206**	**206**	**206**	**206**
6	**322**	**322**	**322**	**322**	**322**
7	232	238	222	296	292
8	**186**	**186**	**186**	**186**	**186**
9	**436**	**436**	**436**	**436**	**436**
10	**244**	**244**	**244**	**244**	**244**
11	–	–	–	–	–
12	–	–	–	–	–
13	–	–	–	–	–
14	–	–	–	–	–
15	–	–	–	–	–
16	–	–	–	–	–
17	–	–	–	–	–
18	–	–	–	–	–
19	–	–	–	–	–
20	–	–	–	–	–
21	–	–	–	–	–

Table 1: Objective value through instances



(a) Resolution time for each instance (b) Number of conflicts for each instance

Figure 1: Statistics about the resolution of the problem for the first 10 instances

4

4 SMT model

4.1 Decision variables

4.2 Objective function

4.3 Constraints

4.4 Validation

5 MIP model

5.1 Decision variables

5.2 Objective function

5.3 Constraints

5.4 Validation