Magic Factory Optimization using Z3

Kaiwen Tan*

September, 2025

1 Modeling the Problem

1.1 Parameters

- $T = 8$: number of trucks.
- $C = 8000$: capacity (kg) per truck.
- $P = 8$: maximum number of pallets per truck.
- Pallet types:
  - Nuzzles: 4 pallets, 700 kg each.
  - Skipples: 8 pallets, 1000 kg each, require cooling.
  - Crottles: 10 pallets, 2500 kg each.
  - Dupples: 20 pallets, 200 kg each.
  - Prittles: unlimited supply, 400 kg each (objective: maximize).
- Cooling: only 3 trucks can carry skipples.

1.2 Decision Variables

Let:

$$x_{i,t} \in \mathbb{N}$$

be the number of pallets of type i assigned to truck t, where $i \in \{\text{nuzzle}, \text{prittle}, \text{skipple}, \text{crottle}, \text{dupple}\}$ and $t \in \{1, \ldots, 8\}$.

Additionally:

$$y_t \in \{0, 1\}$$

indicates whether truck t is equipped with cooling (1) or not (0).

1.3 Constraints

Truck capacity (weight):

$$\forall t: \quad \sum_i (w_i \cdot x_{i,t}) \leq C$$

Truck capacity (pallet count):

$$\forall t: \quad \sum_i x_{i,t} \leq P$$

*Student Number: 2291797 Email: k.tan@student.tue.nl

Supply limits:

$$\sum_t x_{\text{nuzzle},t} = 4, \quad \sum_t x_{\text{skipple},t} = 8, \quad \sum_t x_{\text{crottle},t} = 10, \quad \sum_t x_{\text{dupple},t} = 20$$

Cooling requirement:

$$\sum_t y_t = 3, \quad \forall t : x_{\text{skipple},t} \leq P \cdot y_t$$

Nuzzle distribution:

$$\forall t : \quad x_{\text{nuzzle},t} \leq 1$$

Explosive combination (Part b only):

$$\forall t : \quad x_{\text{prittle},t} \cdot x_{\text{crottle},t} = 0$$

1.4 Objective Function

$$\text{maximize} \quad \sum_t x_{\text{prittle},t}$$

2 Solver Implementation using Z3

- Integer variables n, p, s, c, d represent pallets per truck; boolean variables *cool* indicate cooled trucks.
- Truck-level constraints enforce:
  - Maximum 8 pallets per truck
  - Weight limits
  - Nuzzle distribution (at most 1 per truck)
  - Skipples only on cooled trucks
- Global constraints ensure delivery of all nuzzles, skipples, crottles, dupples, and exactly 3 cooled trucks.
- Part (b) is handled by preventing prittles and crottles in the same truck using `Or(p[i]==0, c[i]==0)`.
- The solver maximizes the total number of prittles.

3 Results Part (a)

3.1 Truck Assignments

Table 1: Truck assignment of pallets for Part (a)

Truck	Nuzzles	Prittles	Skipples	Crottles	Dupples	Total Weight (kg)	Cooling
1	0	1	0	2	5	6400	False
2	1	5	0	0	2	3100	False
3	1	0	0	2	5	6700	False
4	0	0	0	2	6	6200	False
5	1	7	0	0	0	3500	False
6	0	3	1	2	2	7600	True
7	1	1	6	0	0	7100	True
8	0	5	1	2	0	8000	True
Total	4	22	8	10	20	48600	3

3.2 Solver Performance and Optimization Results

Table 2: Optimization results summary (template)

Metric	Value
Total Prittles delivered	22
Solver runtime (s)	0.0241
Solver status	sat

- From the obtained solution, we can see that the solver successfully enforced all constraints. in Part (a), the solver was able to place them flexibly, which allowed the total number of prittles to reach the maximum.
- Observe solver runtime and efficiency. The solver completed the optimization very quickly, taking only 0.0241 s.

4 Results Part (b)

4.1 Truck Assignments

Table 3: Truck assignment of pallets for Part (b)

Truck	Nuzzles	Prittles	Skipples	Crottles	Dupples	Total Weight (kg)	Cooling
1	0	4	4	0	0	5600	True
2	1	0	0	2	5	6700	False
3	1	0	0	2	5	6700	False
4	0	0	2	2	4	7800	True
5	0	8	0	0	0	3200	False
6	1	0	2	2	1	7900	True
7	1	0	0	2	5	6700	False
8	0	8	0	0	0	3200	False
Total	4	20	8	10	20	48400	3

4.2 Solver Performance and Optimization Results

Table 4: Optimization results summary

Metric	Value
Total Prittles delivered	20
Solver runtime (s)	0.5243
Solver status	sat

- In Part (b), the additional constraint preventing prittles and crottles from being assigned to the same truck adds a layer of complexity. The solver correctly respects this restriction along with all previous constraints. As a result, trucks carry either prittles or crottles but not both, and the nuzzles are still properly distributed.
- The runtime increased to 0.5243 s compared to Part (a), reflecting the added combinatorial challenge introduced by the explosive combination constraint. Despite this, the solver still finds the optimal solution reliably.