Optimization of the Waste Collection and Recycling Process

Ulises Díez Santaolalla
Ignacio Felices Vera
Teresa Franco Corzo
Sofía Negueruela Avellaneda

Optimización y Simulación
3ºA Grado en Ingeniería Matemática e Inteligencia Artificial

Content Table

1. Introduction

The goal of this project is to provide an optimal solution for an everyday problem: waste collection. This is a crucial step towards building a sustainable and eco-friendly urban environment. Waste management refers to the activities and actions required to handle waste from its generation to its final disposal. This includes collection and transportation to the landfill site.

However, as global populations continue to grow and waste generation increases, the efficient management of garbage becomes more important than ever. By streamlining collection systems, improving sorting techniques, and enhancing recycling processes, communities can reduce their environmental impact, conserve natural resources, and contribute to a circular economy.

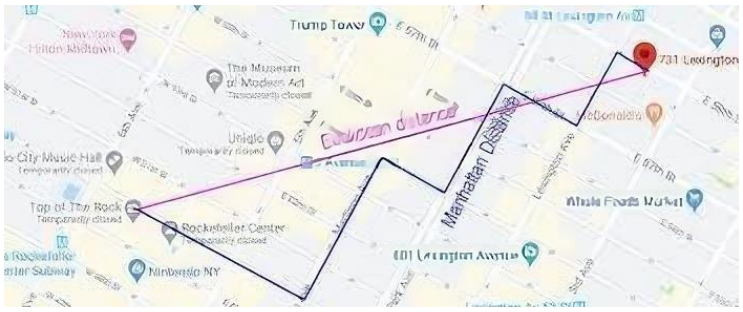
In the following pages we have described how to manage the routes a truck could use to collect as much litter as possible. We want to reduce how long it takes to empty all trash containers and the number of trucks needed for it. This approach not only benefits the planet but also creates economic opportunities and promotes healthier living conditions for residents. If we reduce the time, we are reducing carbon dioxide emissions; and by using less trucks the price goes down, making it more affordable.

With the solution proposed below, we want to transform waste management into a more effective and responsible practice by optimizing the truck routes. Such a small change can make a difference for the planet and our pockets.

2. Problem Statement and Visualization

Waste collection companies must manage routes with varying distances in their daily operations, which presents a significant challenge in terms of efficiency and cost. To illustrate the real-world relevance of this issue, consider the example of managing an average of 19,600 daily routes to serve 20 million residential customers and 2 million commercial clients. The company operates a fleet of 26,000 collection trucks, with an approximate operational cost of $120,000 per vehicle. Given this high cost, it is crucial that each route is managed efficiently and cost-effectively.

In this context, route optimization must focus on reducing costs while ensuring the complete collection of waste within the estimated time. In our specific context, the goal is to optimize the number of recycling trucks within a network and their distribution, based on a system that measures the distance between network nodes using Manhattan distance, to allow coverage of the greatest number of containers. To simulate traffic and the conditions of a street, we have included the time it takes to pass through all containers considering that maybe the time spent on different streets varies.



3. Final Formulation

Here we will present the step-by-step final formulation. The maximum is decided in each level of development, that is, at the basic level, number of rows and number of columns equals to three, and higher afterwards.

3.1 Sets

1. **Rows - i**
   They represent the rows of the matrix that we will go down.
2. **Columns -j**
   The columns of the matrix.
3. **Vehículos -k**
   The number of trucks that will be operating.
4. **Step-s**
   We introduced this set which is 2*rows*columns to make sure each vehicle moved once upon the matrix, taking into account the round-trip path, the maximum being to travel through the entire matrix.

3.2 Parameters

For the development of these parameters, we initialized an empty matrix of the desired width and height specified in the sets.

1. **Supposition on time of service – T(i,j)**
   As we said, at the basic level we considered the time a constant therefore there was no need to reduce the time in the objective function, because reducing the number of steps was reducing the time in the journey. But at the higher level we developed a matrix of random numbers that represented the time that would take a truck to use that cell, and therefore we added it to the objective function.
2. **Garbage in each stop – CP(i,j)**
   We used another matrix to represent the dirt quantity in each stop.
3. **Deposit-D(i,j)**
   This parameter represented the deposit form which all vehicles started their route, and due to the complexity of the problem we have only tried with one deposit through the whole matrix.
4. **Capacity of truck - $\boldsymbol{C_K}$**
   This parameter symbolizes the amount of dirt that a vehicle can collect, and therefore it can vary to add more realism to the project.

3.3 Variables

The variable design has been through a lot of significant changes thought the implementation of this problem, but we came up with the final design that includes these:

1. **Truck route - R_{ijks}**
   It represents the steps between i and j for a vehicle k. It is a binary variable, so it will have a 1 if it uses that cell or 0 if not.
2. **Amount of garbage in the truck in a step - B_{ks}**
   Indicated the amount of dirt for each step that the vehicle makes in the matrix.
3. **Binary variable indicating if a vehicle can collect more garbage – $can\_collect_k$**
   We must model and make sure that each truck does not collect more garbage than its capacity constraint, for that, we need this binary variable.
4. **Binary variable indicating if all the garbage has been collected – $all\_dirt\_collected_s$**
   To send all the vehicles into the deposit we need to have a "flag" to know when this is supposed to happen. This binary variable indicates when the trucks should stop collecting and head back to the deposit.
5. **Number of steps taken to return to deposit – $return\_step_k$**
   This variable controls the number of steps each vehicle takes to return to the deposit which cannot surpass the number of cells of the matrix. This is what we will minimize, since it controls the number of cells that the vehicle goes trough since he leaves the depot until it returns. A truck can pass multiple times through all the cells except for the deposit cell, which can only be passed two times (leaving and arriving).
6. **Mark if garbage has been collected at each step – $is\_collected_{i,j,s}$**
   This variable controls if the truck collects garbage at each step it takes in the route of the matrix. As in our problem statement the truck can move multiple times in a cell, this variable restricts that it will only collect the garbage once it passes the first time. If it uses the same cell later, it will not add up the garbage.

3.4 Objective Function

The final objective function is:

$$\min \sum R_{i,j,k,s} * T_{i,j} + \sum return\_step_k$$

This function represents what we want to minimize, on the one hand, the time that the truck takes to collect the garbage, and on the other hand, the number of steps taken on the route. Notice that the variable R indicates if the truck has used a cell in a step s, therefore, if by any case the truck uses the same cell multiple times through its route, it will be counted.

3.5 Constraints

1. **The truck starts in the depot**
   For this constraint we need to specify that the first step (s = 0) for each truck is initialized in the depot.

$$R_{deposit[0],deposit[1],k,0} = 1 \ \forall k$$

2. **The Manhattan movement constraint**
   To make sure that the truck moves according to the Manhattan distance, which only allows it to move up, down, right and left, and therefore it cannot move diagonally, we implemented this constraint.

$$R_{i,j,k,s} \leq R_{i-1,j,k,s-1} + R_{i+1,j,k,s-1} + R_{i,j-1,k,s-1} + R_{i,j+1,k,s-1} \quad \forall k, s > 0, i, j$$

3. **A truck can only occupy a single cell in each step**
   This helps to visualize that only one movement can be done in a unit of time.

$$\sum_{I}\sum_{j} R_{I,j,k,s} \leq 1 \quad \forall k, s$$

4. **Trucks capacity restriction**
   For each step that the truck makes, the collected garbage cannot exceed its capacity.

$$\sum_{s=1}\sum_{i=1}\sum_{j=1} CP_{i,j} * R_{i,j,k,s} \leq C_k \quad \forall k$$

5. **Actualize the trucks garbage load**
   This constraint allows to update the trucks load when it is collecting garbage. It restricts to accumulate the load only the first time a garbage cell is visited.
   When s = 0 the truck's load capacity is zero, afterwards, the truck's load is equal to the previous cell load, adding the garbage collected in the current step. It is only added if it has not been collected in previous steps.

$$B_{k,s} = B_{k,s-1} + \sum_{i=1}\sum_{j=1} CP_{i,j} * R_{i,j,k,s} * (1 - is\_collected_{i,j,s-1}) \quad \forall s > 0$$

6. **Restrict that a regular stop with garbage will only be recollected once**
   First, one cell can be marked as collected only if it is visited and the truck is able to recollect, so it has space.

$$is\_collected_{i,j,s} - can\_collect_k \geq R_{i,j,k,s} \quad \forall s > 0, i, j, k$$

   Second, we use an if-then logic condition to state that the truck's capacity cannot be exceeded when it is recollecting garbage from the cell in the step s, where the big M is set to a big number to make sure that it is applied if the truck can collect.

$$C_k - (b_{k,s} - d_{i,j}) \geq M * (can\_collect_k - 1) \quad \forall\, s > 0, i, j, k$$

Third, if the cell has been marked as collected in a step, it should remain collected in the following.

$$is\_collected_{i,j,s} \geq is\_collected_{i,j,s-1} \quad \forall\, s > 0, i, j$$

7. **Activating all dirt collected when all garbage has been collected for all the trucks**
This is algo an if-then logic condition, but we want to ensure this, so we use an if and only if, because if all dirt has been collected the variable *all_dirt_collected* has to be set to 1, and if *all_dirt_collected* = 1 then all dirt has been collected, which results in two conditions, the upper bound and the lower bound.

$$M * (1 - all\_dirt\_collected_s) \geq \sum_{i,j} CP_{i,j} - \sum_{k} B_{k,s} \quad \forall s$$

$$m * (1 - all\_dirt\_collected_s) \leq \sum_{i,j} CP_{i,j} - \sum_{k} B_{k,s} \quad \forall s$$

8. **The depot can only be visited twice**
Once it leaves and once it arrives.

$$\sum_{s} R_{deposit[0],deposit[1]} = 2 \quad \forall k$$

9. **A truck can visit the depot only of all garbage has been collected**
This ensures that a second visit to the depot can only be made once all the dirt is collected.

$$R_{deposit[0],deposit[1],k,s} \leq all\_dirt\_collected_s \quad \forall k, s > 0$$

10. **Explicitly indicating the return step of each truck**
This constraint indicates the steps taken for each truck since the departure of the deposit and allows us to minimize afterwards.

$$return\_step_k = \sum_{s=1} s * R_{deposit[0],deposit[1],k,s} \quad \forall k$$

11. **Stop condition**
Once all dirt is collected and all trucks are in the deposit, the trucks should not leave the depot**.**

$$\boldsymbol{\sum_{i=1} \sum_{j=1} R_{i,j,k,s} \leq 1 - R_{deposit[0],deposit[1],k,s-1} \quad \forall k, s > 1}$$

12. **At each cell the garbage is collected only once**
    This constraint ensures that garbage from each cell is collected at most once, as the total garbage collected cannot exceed the initial amount of garbage present in that cell.

$$\sum_{k}\sum_{s=1} R_{i,j,k,s} * CP_{i,j} \;\leq\; CP_{i,j}$$

4. Methodology

In this optimization problem, as explained before, we have considered minimizing both the number of trucks and the distance traveled by each of them. However, we have observed that reducing the distance does not always decrease the total cost, as some routes, although longer, may be completed in less time. Our goal is to optimize the waste management process.

Minimizing the number of trucks also presents limitations. Although fewer trucks could be used, this might result in less efficiency than before. In the long run, it could be more cost-effective to employ a larger number of trucks, which would significantly reduce the total time of the process. In this sense, it is a trade-off between labor costs and the additional use of trucks, which could ultimately be more beneficial.

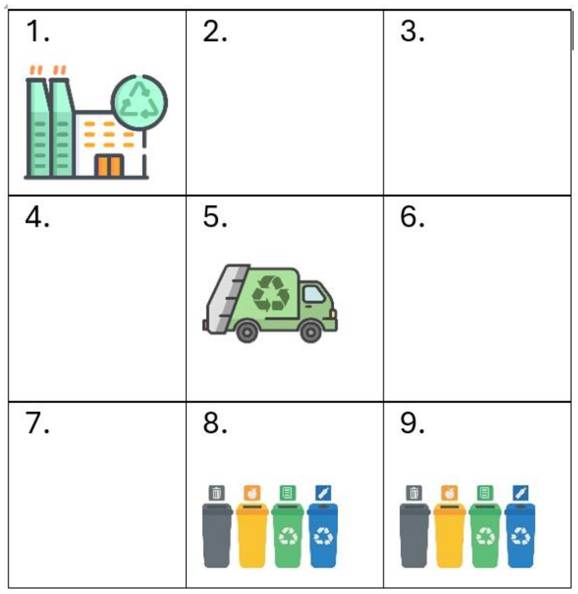
Therefore, we have determined that time is the key variable to minimize, as it directly impacts costs such as wages, fuel, and other expenses related to the actual duration of the process, which are not directly addressed in our optimization problem. For this reason, we have decided that the best strategy to optimize this problem is to minimize the route time for each truck, so that is our goal.

The difficulty of the problem makes us divide the project into two phases: functionality and realism. We want for sure a real solution, but we are giving priority to finding an optimal one first. We will first find the shortest route for one truck, if minimizing distance and time is equivalent, although we have discussed before that it is not always true.

After achieving an optimal solution, we will start increasing the difficulties of the problem step by step, only moving forward once we have reached an optimal solution for each new upgrade. We want the final version of the problem to address the time it takes to travel from one node to another in the grid, simulating the streets, with some taking longer to travel. This time is predetermined. Initially, it will be constant, but in an expanded version of the problem, it will vary, allowing for optimization of the different possible routes.

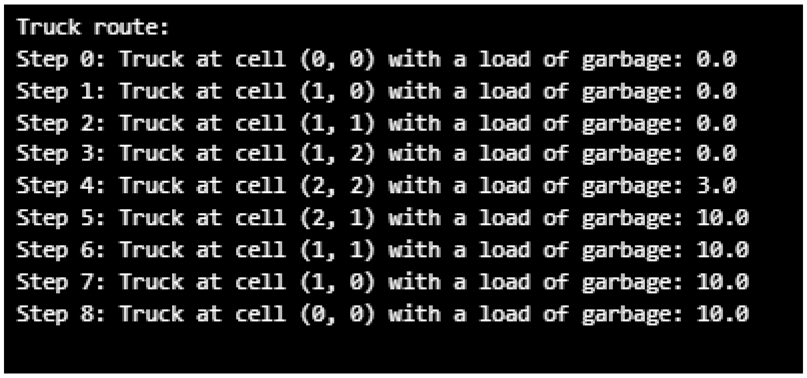
4.2 First Phase – Functional solution

To start with, the problem will consist of a 3x3 grid with only one truck, one deposit, and three litter dumps. We included initially a constant representing the time it takes to move from one cell to another. However, we noticed soon that this was useless, because minimizing the time of the route is the same as minimizing the number of cells the truck passes, so we formulated the solution of the simplified version. The initial grid is presented below.



Modifications: we simplified the objective function to be just minimizing the return steps, as we considered time to be a constant trough cells, and all constraints and variables are reduced to hold only one truck, therefore tone dimension is missing from the formulation in point 3.

Results: as this model is very basic, the gurobi solver only indicates one possible solution.



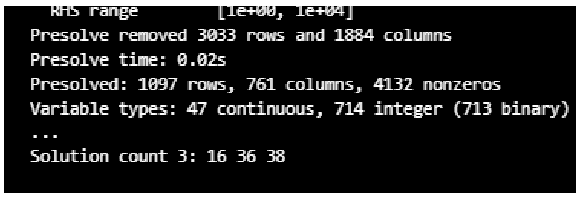
```
Truck route:
Step 0: Truck at cell (0, 0) with a load of garbage: 0.0
Step 1: Truck at cell (1, 0) with a load of garbage: 0.0
Step 2: Truck at cell (1, 1) with a load of garbage: 0.0
Step 3: Truck at cell (1, 2) with a load of garbage: 0.0
Step 4: Truck at cell (2, 2) with a load of garbage: 3.0
Step 5: Truck at cell (2, 1) with a load of garbage: 10.0
Step 6: Truck at cell (1, 1) with a load of garbage: 10.0
Step 7: Truck at cell (1, 0) with a load of garbage: 10.0
Step 8: Truck at cell (0, 0) with a load of garbage: 10.0
```

A visible step by step route in the matrix can be seen represented in the code for each of the verions described.

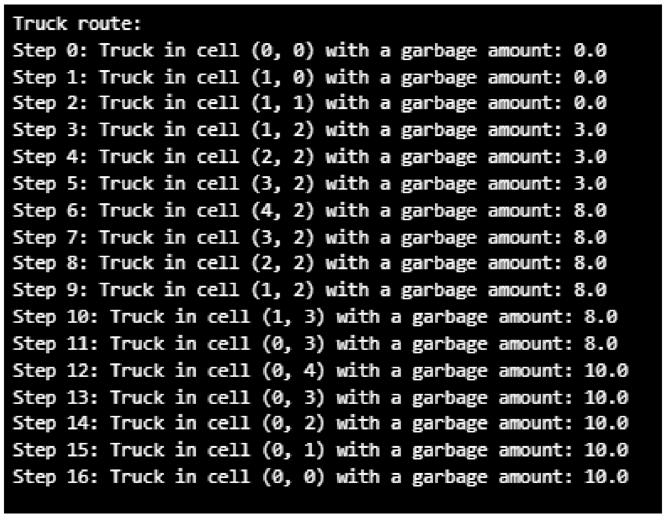
After obtaining an optimal solution, we tried with a 5x5 grid and three litter dumps.

1. 🏭	2.	3.	4.	5. 🗑️
6.	7.	8. 🗑️	9.	10.
11.	12. 🚛	13.	14.	15.
16.	17.	18.	19.	20.
21.	22.	23. 🗑️	24.	25.

Results: we managed to see that the gurobi solver finds multiple possibilities, but one with a lower number of steps, which is what we are searching for.



```
  RHS range        [1e+00, 1e+04]
Presolve removed 3033 rows and 1884 columns
Presolve time: 0.02s
Presolved: 1097 rows, 761 columns, 4132 nonzeros
Variable types: 47 continuous, 714 integer (713 binary)
...
Solution count 3: 16 36 38
```

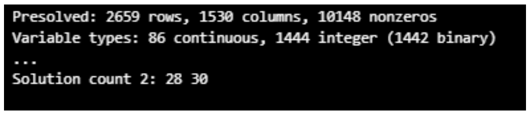


```
Truck route:
Step 0: Truck in cell (0, 0) with a garbage amount: 0.0
Step 1: Truck in cell (1, 0) with a garbage amount: 0.0
Step 2: Truck in cell (1, 1) with a garbage amount: 0.0
Step 3: Truck in cell (1, 2) with a garbage amount: 3.0
Step 4: Truck in cell (2, 2) with a garbage amount: 3.0
Step 5: Truck in cell (3, 2) with a garbage amount: 3.0
Step 6: Truck in cell (4, 2) with a garbage amount: 8.0
Step 7: Truck in cell (3, 2) with a garbage amount: 8.0
Step 8: Truck in cell (2, 2) with a garbage amount: 8.0
Step 9: Truck in cell (1, 2) with a garbage amount: 8.0
Step 10: Truck in cell (1, 3) with a garbage amount: 8.0
Step 11: Truck in cell (0, 3) with a garbage amount: 8.0
Step 12: Truck in cell (0, 4) with a garbage amount: 10.0
Step 13: Truck in cell (0, 3) with a garbage amount: 10.0
Step 14: Truck in cell (0, 2) with a garbage amount: 10.0
Step 15: Truck in cell (0, 1) with a garbage amount: 10.0
Step 16: Truck in cell (0, 0) with a garbage amount: 10.0
```

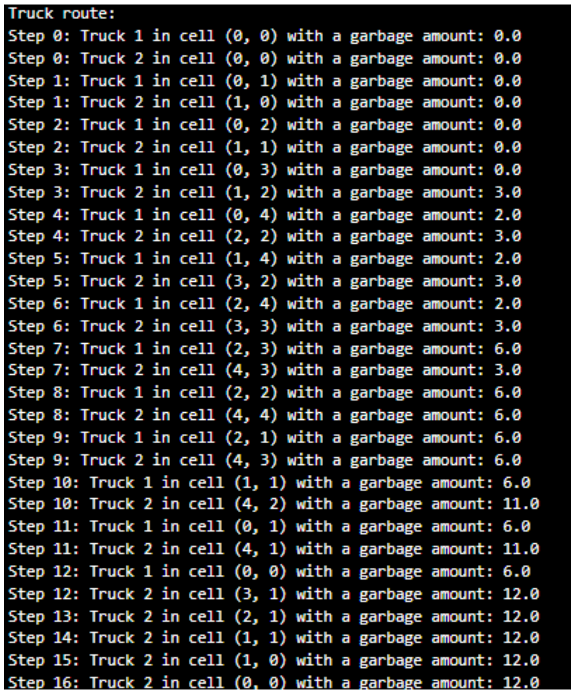
4.2 Second Phase – Realistic solution

Now we modified the code and finally came up with the formulation in point 3 that makes the problem work efficiently, modifying the constraints to add more trucks and the objective function to have the varying time matrix.

Case 1: for this, we set off from the 5x5 example but adding two trucks, and therefore maintaining time constant, and this is what we obtained:



```
Presolved: 2659 rows, 1530 columns, 10148 nonzeros
Variable types: 86 continuous, 1444 integer (1442 binary)
...
Solution count 2: 28 30
```

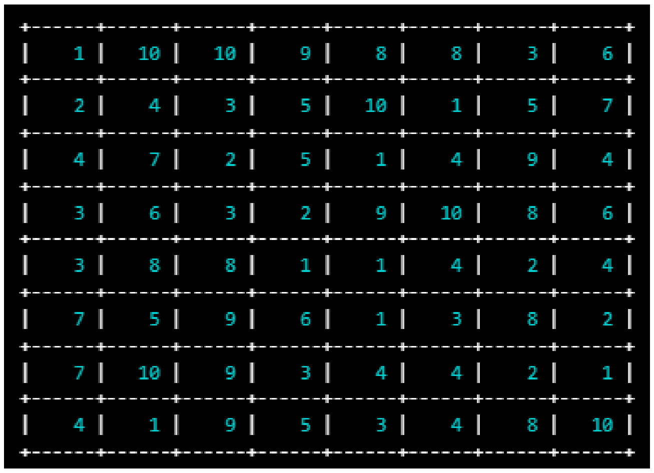


```
Truck route:
Step 0: Truck 1 in cell (0, 0) with a garbage amount: 0.0
Step 0: Truck 2 in cell (0, 0) with a garbage amount: 0.0
Step 1: Truck 1 in cell (0, 1) with a garbage amount: 0.0
Step 1: Truck 2 in cell (1, 0) with a garbage amount: 0.0
Step 2: Truck 1 in cell (0, 2) with a garbage amount: 0.0
Step 2: Truck 2 in cell (1, 1) with a garbage amount: 0.0
Step 3: Truck 1 in cell (0, 3) with a garbage amount: 0.0
Step 3: Truck 2 in cell (1, 2) with a garbage amount: 3.0
Step 4: Truck 1 in cell (0, 4) with a garbage amount: 2.0
Step 4: Truck 2 in cell (2, 2) with a garbage amount: 3.0
Step 5: Truck 1 in cell (1, 4) with a garbage amount: 2.0
Step 5: Truck 2 in cell (3, 2) with a garbage amount: 3.0
Step 6: Truck 1 in cell (2, 4) with a garbage amount: 2.0
Step 6: Truck 2 in cell (3, 3) with a garbage amount: 3.0
Step 7: Truck 1 in cell (2, 3) with a garbage amount: 6.0
Step 7: Truck 2 in cell (4, 3) with a garbage amount: 3.0
Step 8: Truck 1 in cell (2, 2) with a garbage amount: 6.0
Step 8: Truck 2 in cell (4, 4) with a garbage amount: 6.0
Step 9: Truck 1 in cell (2, 1) with a garbage amount: 6.0
Step 9: Truck 2 in cell (4, 3) with a garbage amount: 6.0
Step 10: Truck 1 in cell (1, 1) with a garbage amount: 6.0
Step 10: Truck 2 in cell (4, 2) with a garbage amount: 11.0
Step 11: Truck 1 in cell (0, 1) with a garbage amount: 6.0
Step 11: Truck 2 in cell (4, 1) with a garbage amount: 11.0
Step 12: Truck 1 in cell (0, 0) with a garbage amount: 6.0
Step 12: Truck 2 in cell (3, 1) with a garbage amount: 12.0
Step 13: Truck 2 in cell (2, 1) with a garbage amount: 12.0
Step 14: Truck 2 in cell (1, 1) with a garbage amount: 12.0
Step 15: Truck 2 in cell (1, 0) with a garbage amount: 12.0
Step 16: Truck 2 in cell (0, 0) with a garbage amount: 12.0
```

Case 2: we amplified the matrix to an 8x8, now adding the different times and 4 trucks.

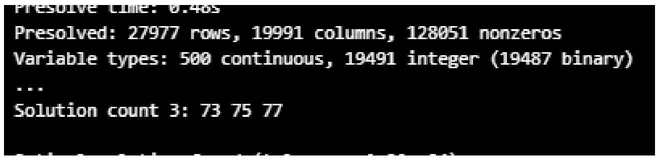
1. 🏭	2.	3.	4.	5. 🗑️	6.	7.	8.
9.	10. 🚛	11. 🗑️	12.	13.	14.	15. 🗑️	16.
17.	18.	19.	20. 🗑️	21.	22. 🗑️	23.	24.
25.	26. 🗑️	27.	28.	29.	30.	31. 🗑️	32.
33.	34.	35. 🗑️	36.	37. 🗑️	38.	39.	40.
41.	42.	43. 🗑️	44.	45.	46. 🗑️	47.	48. 🚛
49. 🗑️	50.	51.	52.	53.	54. 🚛	55.	56.
57.	58. 🗑️	59.	60.	61. 🚛	62.	63.	64.

And a time matrix:

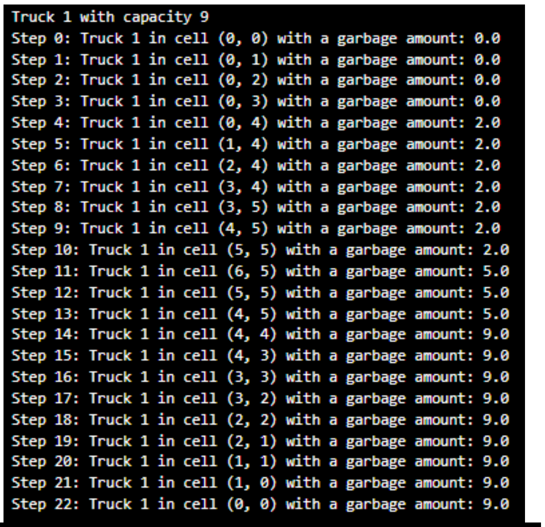


```
+------+------+------+------+------+------+------+------+
|    1 |   10 |   10 |    9 |    8 |    8 |    3 |    6 |
+------+------+------+------+------+------+------+------+
|    2 |    4 |    3 |    5 |   10 |    1 |    5 |    7 |
+------+------+------+------+------+------+------+------+
|    4 |    7 |    2 |    5 |    1 |    4 |    9 |    4 |
+------+------+------+------+------+------+------+------+
|    3 |    6 |    3 |    2 |    9 |   10 |    8 |    6 |
+------+------+------+------+------+------+------+------+
|    3 |    8 |    8 |    1 |    1 |    4 |    2 |    4 |
+------+------+------+------+------+------+------+------+
|    7 |    5 |    9 |    6 |    1 |    3 |    8 |    2 |
+------+------+------+------+------+------+------+------+
|    7 |   10 |    9 |    3 |    4 |    4 |    2 |    1 |
+------+------+------+------+------+------+------+------+
|    4 |    1 |    9 |    5 |    3 |    4 |    8 |   10 |
+------+------+------+------+------+------+------+------+
```

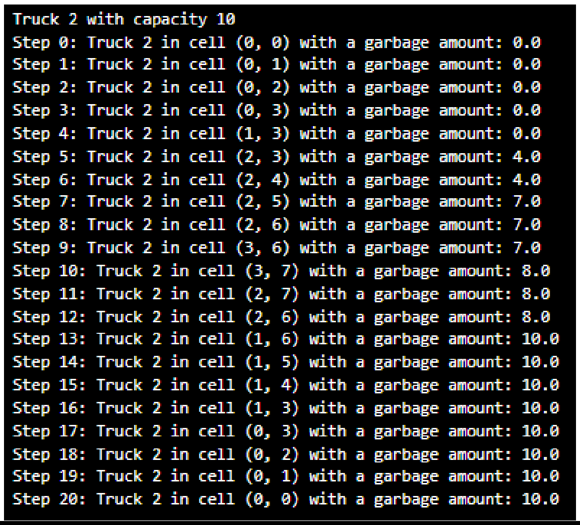
Which resulted in three different possibilities:



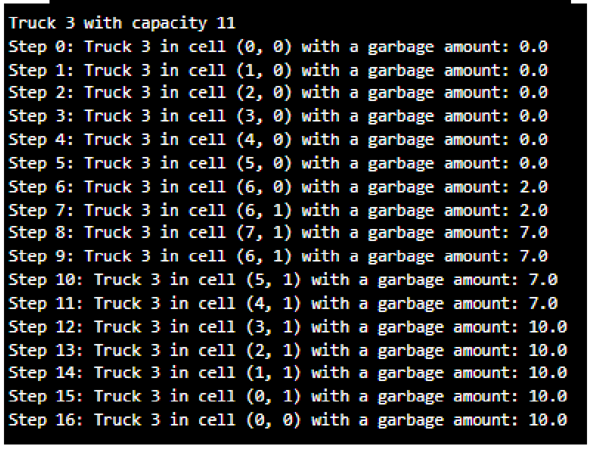
```
Presolve time: 0.48s
Presolved: 27977 rows, 19991 columns, 128051 nonzeros
Variable types: 500 continuous, 19491 integer (19487 binary)
...
Solution count 3: 73 75 77
```



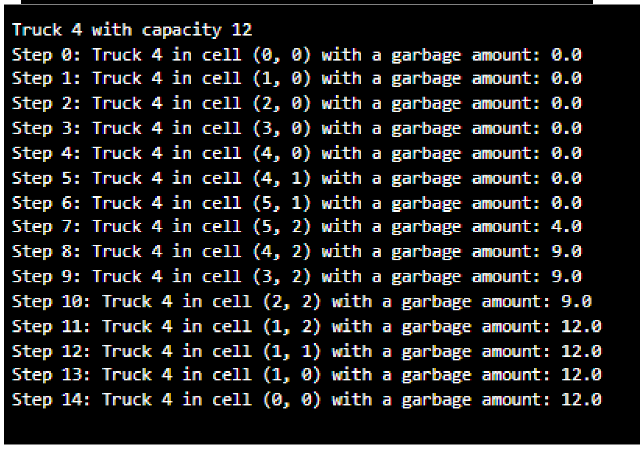
```
Truck 1 with capacity 9
Step 0: Truck 1 in cell (0, 0) with a garbage amount: 0.0
Step 1: Truck 1 in cell (0, 1) with a garbage amount: 0.0
Step 2: Truck 1 in cell (0, 2) with a garbage amount: 0.0
Step 3: Truck 1 in cell (0, 3) with a garbage amount: 0.0
Step 4: Truck 1 in cell (0, 4) with a garbage amount: 2.0
Step 5: Truck 1 in cell (1, 4) with a garbage amount: 2.0
Step 6: Truck 1 in cell (2, 4) with a garbage amount: 2.0
Step 7: Truck 1 in cell (3, 4) with a garbage amount: 2.0
Step 8: Truck 1 in cell (3, 5) with a garbage amount: 2.0
Step 9: Truck 1 in cell (4, 5) with a garbage amount: 2.0
Step 10: Truck 1 in cell (5, 5) with a garbage amount: 2.0
Step 11: Truck 1 in cell (6, 5) with a garbage amount: 5.0
Step 12: Truck 1 in cell (5, 5) with a garbage amount: 5.0
Step 13: Truck 1 in cell (4, 5) with a garbage amount: 5.0
Step 14: Truck 1 in cell (4, 4) with a garbage amount: 9.0
Step 15: Truck 1 in cell (4, 3) with a garbage amount: 9.0
Step 16: Truck 1 in cell (3, 3) with a garbage amount: 9.0
Step 17: Truck 1 in cell (3, 2) with a garbage amount: 9.0
Step 18: Truck 1 in cell (2, 2) with a garbage amount: 9.0
Step 19: Truck 1 in cell (2, 1) with a garbage amount: 9.0
Step 20: Truck 1 in cell (1, 1) with a garbage amount: 9.0
Step 21: Truck 1 in cell (1, 0) with a garbage amount: 9.0
Step 22: Truck 1 in cell (0, 0) with a garbage amount: 9.0
```



```
Truck 2 with capacity 10
Step 0: Truck 2 in cell (0, 0) with a garbage amount: 0.0
Step 1: Truck 2 in cell (0, 1) with a garbage amount: 0.0
Step 2: Truck 2 in cell (0, 2) with a garbage amount: 0.0
Step 3: Truck 2 in cell (0, 3) with a garbage amount: 0.0
Step 4: Truck 2 in cell (1, 3) with a garbage amount: 0.0
Step 5: Truck 2 in cell (2, 3) with a garbage amount: 4.0
Step 6: Truck 2 in cell (2, 4) with a garbage amount: 4.0
Step 7: Truck 2 in cell (2, 5) with a garbage amount: 7.0
Step 8: Truck 2 in cell (2, 6) with a garbage amount: 7.0
Step 9: Truck 2 in cell (3, 6) with a garbage amount: 7.0
Step 10: Truck 2 in cell (3, 7) with a garbage amount: 8.0
Step 11: Truck 2 in cell (2, 7) with a garbage amount: 8.0
Step 12: Truck 2 in cell (2, 6) with a garbage amount: 8.0
Step 13: Truck 2 in cell (1, 6) with a garbage amount: 10.0
Step 14: Truck 2 in cell (1, 5) with a garbage amount: 10.0
Step 15: Truck 2 in cell (1, 4) with a garbage amount: 10.0
Step 16: Truck 2 in cell (1, 3) with a garbage amount: 10.0
Step 17: Truck 2 in cell (0, 3) with a garbage amount: 10.0
Step 18: Truck 2 in cell (0, 2) with a garbage amount: 10.0
Step 19: Truck 2 in cell (0, 1) with a garbage amount: 10.0
Step 20: Truck 2 in cell (0, 0) with a garbage amount: 10.0
```



```
Truck 3 with capacity 11
Step 0: Truck 3 in cell (0, 0) with a garbage amount: 0.0
Step 1: Truck 3 in cell (1, 0) with a garbage amount: 0.0
Step 2: Truck 3 in cell (2, 0) with a garbage amount: 0.0
Step 3: Truck 3 in cell (3, 0) with a garbage amount: 0.0
Step 4: Truck 3 in cell (4, 0) with a garbage amount: 0.0
Step 5: Truck 3 in cell (5, 0) with a garbage amount: 0.0
Step 6: Truck 3 in cell (6, 0) with a garbage amount: 2.0
Step 7: Truck 3 in cell (6, 1) with a garbage amount: 2.0
Step 8: Truck 3 in cell (7, 1) with a garbage amount: 7.0
Step 9: Truck 3 in cell (6, 1) with a garbage amount: 7.0
Step 10: Truck 3 in cell (5, 1) with a garbage amount: 7.0
Step 11: Truck 3 in cell (4, 1) with a garbage amount: 7.0
Step 12: Truck 3 in cell (3, 1) with a garbage amount: 10.0
Step 13: Truck 3 in cell (2, 1) with a garbage amount: 10.0
Step 14: Truck 3 in cell (1, 1) with a garbage amount: 10.0
Step 15: Truck 3 in cell (0, 1) with a garbage amount: 10.0
Step 16: Truck 3 in cell (0, 0) with a garbage amount: 10.0
```



```
Truck 4 with capacity 12
Step 0: Truck 4 in cell (0, 0) with a garbage amount: 0.0
Step 1: Truck 4 in cell (1, 0) with a garbage amount: 0.0
Step 2: Truck 4 in cell (2, 0) with a garbage amount: 0.0
Step 3: Truck 4 in cell (3, 0) with a garbage amount: 0.0
Step 4: Truck 4 in cell (4, 0) with a garbage amount: 0.0
Step 5: Truck 4 in cell (4, 1) with a garbage amount: 0.0
Step 6: Truck 4 in cell (5, 1) with a garbage amount: 0.0
Step 7: Truck 4 in cell (5, 2) with a garbage amount: 4.0
Step 8: Truck 4 in cell (4, 2) with a garbage amount: 9.0
Step 9: Truck 4 in cell (3, 2) with a garbage amount: 9.0
Step 10: Truck 4 in cell (2, 2) with a garbage amount: 9.0
Step 11: Truck 4 in cell (1, 2) with a garbage amount: 12.0
Step 12: Truck 4 in cell (1, 1) with a garbage amount: 12.0
Step 13: Truck 4 in cell (1, 0) with a garbage amount: 12.0
Step 14: Truck 4 in cell (0, 0) with a garbage amount: 12.0
```

5. Conclusion

To conclude, we believe we have made significant progress throughout the process of modeling this problem, which initially seemed overly ambitious. We started from a point where we felt stuck and had to reconsider our initial formulation. Eventually, we developed a functional, although basic, solution that motivated us to continue refining the model.

The main challenges involved programming the truck to make step-by-step decisions within the matrix and integrating the complex process of garbage collection, which required handling multiple variables. Properly configuring and formulating these variables was a difficult task. However, through incremental improvements, we gradually enhanced the model to handle more complex scenarios, such as Case 2.

We are proud to present a model that simulates the entire garbage collection process with multiple trucks. It allows for an efficient distribution of tasks between trucks, minimizes time and distance for returning to the depot, and ensures that trucks stop at the depot when necessary—something we initially struggled to implement.

6. Bibliography

- CONTENUR, 2024. *Madrid estrena sistema de recogida de carga lateral con contenedores CONTENUR*. Disponible en: https://www.contenur.com/noticias/madrid-estrena-sistema-de-recogida-de-carga-lateral-con-contenedores-contenur#:~:text=Los%20nuevos%20contenedores%20para%20las,que%20hay%20en%20estos%20momentos

- Diario Madrid, 2024. *Madrid incorporará 36 nuevos camiones eco para la recogida de residuo orgánico*. Disponible en: https://diario.madrid.es/blog/notas-de-prensa/madrid-incorporara-36-nuevos-camiones-eco-para-la-recogida-de-residuo-organico/

- Ayuntamiento de Madrid, 2024. *Recogida de residuos mediante el sistema de carga lateral*. Disponible en: https://www.madrid.es/portales/munimadrid/es/Inicio/Medio-ambiente/Recogida-de-residuos/Recogida-de-residuos-mediante-el-sistema-de-carga-lateral/?vgnextfmt=default&vgnextoid=91b6a48e8cf1c510VgnVCM1000001d4a900aRCRD&vgnextchannel=f81379ed268fe410VgnVCM1000000b205a0aRCRD

- Koushki, P.A., 2012. *Benefits from GIS-Based Modelling for Municipal Solid Waste Management*. [online] ResearchGate. Disponible en: https://www.researchgate.net/publication/221914795_Benefits_from_GIS_Based_Modelling_for_Municipal_Solid_Waste_Management

- Erol, R. and Ferhatosmanoglu, H., 2016. *Solving the problem of vehicle routing by evolutionary algorithm*. ResearchGate. Disponible en: https://www.researchgate.net/publication/293907531_Solving_the_problem_of_vehicle_routing_by_evolutionary_algorithm