Cycle allocation in a parking lot

Presented By
Tanmay Nath(23N0457)
Vinayak Lal (22b2223)
Palusa sai Rohan (200050095)
Ditipriya Paul (23N0465)

IE501 : Course mini Project

Autumn, 2023-24

Presentation Outline

- Problem Statement
- Assumptions
- Reference Taken
- Tools used for modelling and solving
- Formulation
- Results
- Future Work

Problem Statement

- In a parking lot it is always seen that the cycles are kept in a very disordered manner and is sometimes seen that the students get late as there remains some cycle in their route or they have to remove the cycles from their position for taking out their own cycle, which is not very efficient.
- So, our problem is to find a optimal positioning of the cycle such that there is available route for the cycle to bring it out without affecting the parking of the other cycles.
- we are going to formulate a Integer linear program to find an optimal solution for the problem.

Design and Assumptions of the parking lot

- The Parking lot is assumed to be a rectangular field as seen in most of the hostel areas.
- We divide the cycle stand into rectanguar smaller areas where each grid is such that it is enough to hold the dimensions of each cycle and it is such that each such grid is a entry of a nXm matrix.
- Here we have assumed if the cycle is such that it has to move in row wise to come out, it can easily move but if it has to move in the column direction from its position of being parked then it can only move one step (i.e one position) in the column and not more than that.

Reference Text

- We looked upon various Linear Programming books for taking help mostly named in the course outline given to us.
- We searched up various implication of the Big M method for applying more than one 'or' kind of constraints as we were applying it for the first time
- We read more advanced research paper to get some idea, listed below:
  [1] A mixed integer linear programming model for optimal planning of bicycle sharing systems: A case study in Beijing
  [2] Evaluation of random parking layout SBA mall using integer linear programming

Tools Used for framework and solving

- We used python as the programming language for the whole implementing the formulation of the project.
- We used google collab as the programming software.
- e used pyomo to model the whole problem including decision variables, objective function and constraints
- We used "cbc" solver to solve this problem, Given the moderate scale of linear programming and mixed-integer programming problems encountered in the project, CBC was deemed suitable, offering a balance between performance and accessibility for optimization tasks.

Formulation

- Since the entire parking space is modelled as a grid. We have selected our decision variable as $x_{i,j}$ The variable x_{ij} is defined as:

$$x_{ij} = \begin{cases} 1 & \text{when a cycle is parked at (i,j)} \\ 0 & \text{if the space is empty} \end{cases}$$

- Then, for a cycle to be removed it needs a clear path. So a cycle in an interior position has 6 possible ways to go, which are up+right, up+left, left,right,down+left and down+right.

Formulation

The expression for each of the constraints are(for each of $x_{i,j}$-

$$(left)\sum_{k=1}^{j-1}\mathrm{X}_{i,k} + (j-1)\times \mathrm{X}_{i,j} \leq j-1+M\times(1-\mathrm{y1}_{i,j})$$

$$(right)\sum_{k=j+1}^{m}\mathrm{X}_{i,k} + (m-j)\times \mathrm{X}_{i,j} \leq m-j+M\times(1-\mathrm{y2}_{i,j})$$

$$(up+left)\sum_{k=1}^{j}\mathrm{X}_{i-1,k} + (j)\times \mathrm{X}_{i,j} \leq j+M\times(1-\mathrm{y3}_{i,j})$$

$$(up+right)\sum_{k=j}^{m}\mathrm{X}_{i-1,k} + (m-j+1)\times \mathrm{X}_{i,j} \leq m-j+1+M\times(1-\mathrm{y4}_{i,j})$$

Formulation

$$(down + left)\sum_{k=1}^{j} \mathrm{X}_{i+1,k} + (j) \times \mathrm{X}_{i,j} \leq j + M \times (1 - \mathrm{y5}_{i,j})$$

$$(down + right)\sum_{k=j}^{m} \mathrm{X}_{i+1,k} + (m - j + 1) \times \mathrm{X}_{i,j} \leq m - j + 1 + M \times (1 - \mathrm{y6}_{i,j})$$

$$y1_{i,j} + y2_{i,j} + y3_{i,j} + y4_{i,j} + y5_{i,j} + y6_{i,j} = 1$$

$$(objective)\text{Maximize: } \{\sum_{i,j} x_{ij}\}$$

Now for the top and bottom row, there will be only 4 constraints and 4 y_i 's since there are only 4 possible directions it could take.

The code can be found by clicking on the link below -
`https://colab.research.google.com/drive/16KpSMvuTLZO2nVXhbYQ217lG7C715PXQ?usp=sharing`

Few Results


```
------------------
|  |o|o|o|o|o|o|  |
   -------------
|  |o| | | | | |  |
   -------------
|  |o|o|o|o|o|o|  |
   -------------
|  |o|o|o|o|o|o|  |
   -------------
|  |o| | | | | |  |
   -------------
|  |o|o|o|o|o|o|  |
   -------------
```

6x6 parking space

Optimal Value = 26

```
-----------------
|  | |o|o|o|o|  |
   -----------
|  |o|o| | | |  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  | | |o|o| |  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  |o| | | | |  |
   -----------
|  |o|o|o|o|o|  |
   -----------
```

7x5 parking space

Optimal Value = 25

Few Results


```
-----------------
|  |o| |o|o|o|  |
   -----------
|  |o|o| | | |  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  | | | | |o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  | | | | |o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
```

8x5 parking space

Optimal Value = 28

```
-----------------
|  |o|o|o|o|o|  |
   -----------
|  | | | | |o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  | | | | |o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
|  | | | | |o|  |
   -----------
|  |o|o|o|o|o|  |
   -----------
```

9x5 parking space

Optimal Value = 35

Future Work

- The possible work in this could be to bring in time constraints as well where we built up a dynamic optimization allotment problem based on what time we need the cycle.
- Also we can remove our earlier assumption of allowing only one step in column and try for a more broader possibility by increasing the movement freedom in the columns as well.
- We can simulate various values of n and m and look for the maximum allotment and then accordingly plan for our parking lot
- Such dynamic programming can be used in the delivery boy packing problem as well where he has to pack the items based on the time at which he will reach the stops where he will deliver the items such that minimum number of items need to get displaced to take the reqd item out.

Thank You