PS3 Optimization of Warehouse operations

Team breakfastsearch
Chan Eu Ching
Alexander Ng
Mak Qin Xiang
Christopher Arif

Executive Summary

Currently, In the PSA warehouses, logistics flow is not optimized because of lack of proper resource allocation. There is little to no visibility of the availability of docking bays. This may lead to under usage or overuse of the docking bays. An under usage of docking bays would lead to the problem of the warehouse not fully optimizing their docking bays, and thus waste valuable time and space for the warehouses, while an overuse would waste valuable manpower resources from the haulier companies as they would be left waiting for a docking bay.

To solve this, we have developed a one-stop platform for both warehouses to assign available docking bays to hauliers, and for hauliers to check when and where to truck in and truck out containers.

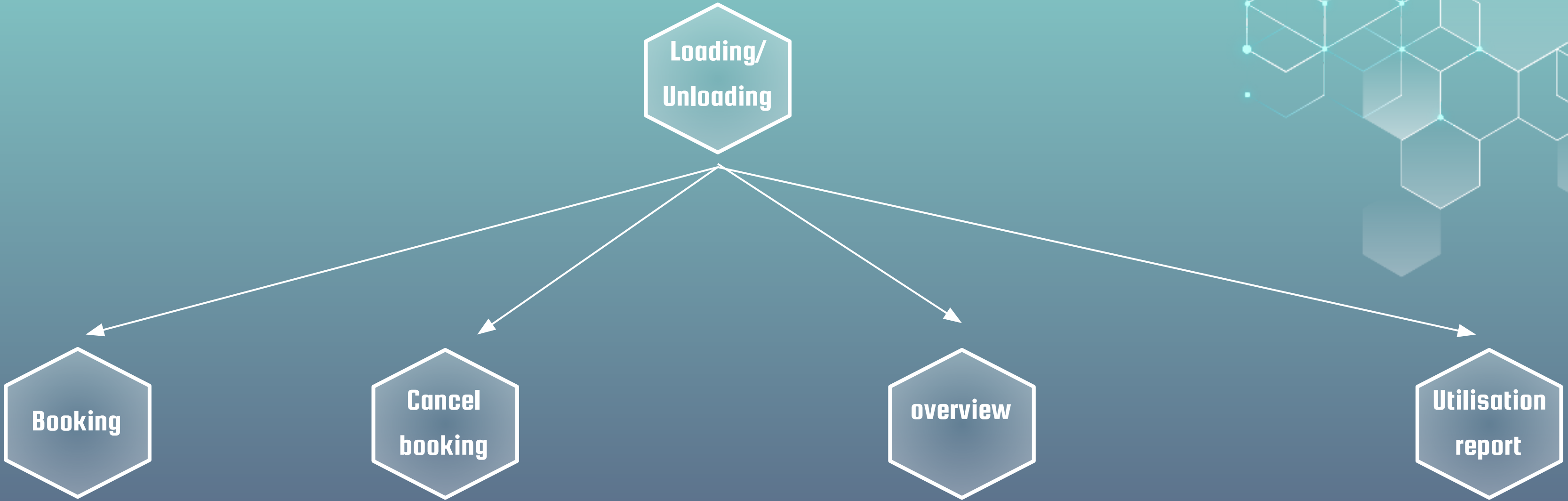
We also figured out that even with the visibility of available docking bays and scheduled timings, the warehouse managers would still have to manually schedule jobs on a given day. Therefore, we utilised a classical computer science optimisation algorithm, the multiple knapsack problem, to allocate jobs required in a day. This will greatly improve job efficiency, prevents the initial problem of idle or oversubscribed docking bays (we consider each docking bay as a single knapsack), and maximises warehouse utilisation rate (up to 100%) while lightening the burden of scheduling for the warehouse.

1.

Solution

1. Warehouse can first choose whether loading or unloading will take place.
2. Half of the docking bays will be allocated for loading and another half for unloading.
3. Our system will then show available timeslots in 30min intervals on each date for booking
4. The haulier will be then be able to view when and where to truck in and out the specific containers in advance of the booking timing.

Structure of Website (Warehouse)



Loading/ Unloading

Booking

Cancel booking

overview

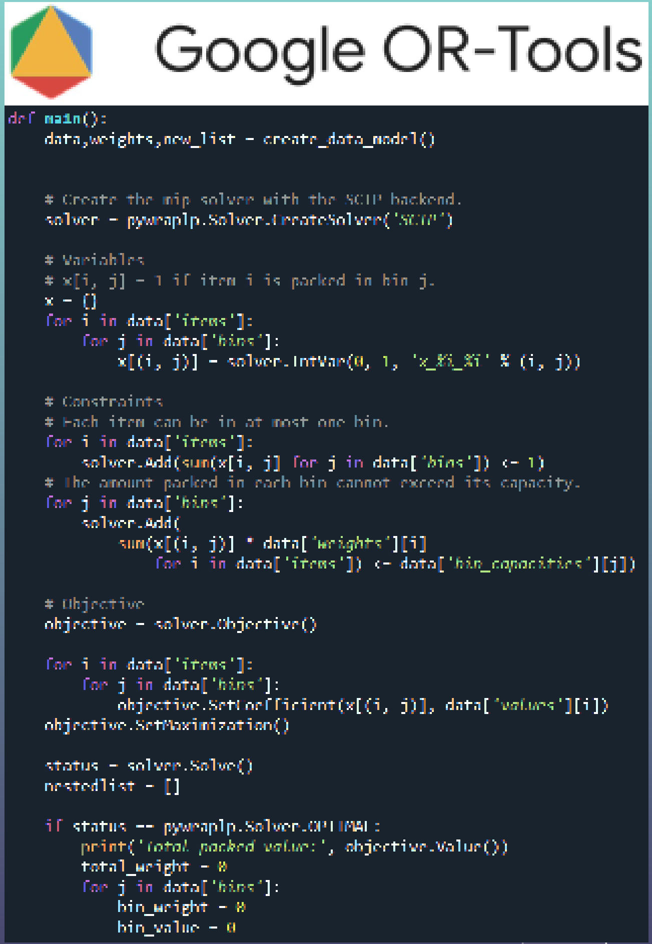
Utilisation report

Structure of Website (Haulier)

View jobs

Additional Solution

- Allocating jobs efficiently is to maximise the usage of the loading and unloading bays
- We treated this as a multiple knapsack problem.
    - The knapsack refers to the individual docking bays
    - Size of the knapsack is the total number of 30min intervals in a day (48)
    - The weight and value of the items (containers being loaded/unloaded) is the number of 30 min intervals required to complete the job.
- We then store the results in a csv file which will then be processed by the website for an easier view.



```python
def main():
    data,weights,new_list = create_data_model()


    # Create the mip solver with the SCIP backend.
    solver = pywraplp.Solver.CreateSolver('SCIP')

    # Variables
    # x[i, j] = 1 if item i is packed in bin j.
    x = {}
    for i in data['items']:
        for j in data['bins']:
            x[(i, j)] = solver.IntVar(0, 1, 'x_%i_%i' % (i, j))

    # Constraints
    # Each item can be in at most one bin.
    for i in data['items']:
        solver.Add(sum(x[i, j] for j in data['bins']) <= 1)
    # The amount packed in each bin cannot exceed its capacity.
    for j in data['bins']:
        solver.Add(
            sum(x[(i, j)] * data['weights'][i]
                for i in data['items']) <= data['bin_capacities'][j])

    # Objective
    objective = solver.Objective()

    for i in data['items']:
        for j in data['bins']:
            objective.SetCoefficient(x[(i, j)], data['values'][i])
    objective.SetMaximization()

    status = solver.Solve()
    nestedlist = []

    if status == pywraplp.Solver.OPTIMAL:
        print('Total packed value:', objective.Value())
        total_weight = 0
        for j in data['bins']:
            bin_weight = 0
            bin_value = 0
```

Unoptimised Solution

0000	ContainerID: c101 HaulierID: h001	ContainerID: c105 HaulierID: h005	ContainerID: c102 HaulierID: h002
0030	ContainerID: c104 HaulierID: h001	ContainerID: c105 HaulierID: h005	ContainerID: c103 HaulierID: h003
0100	ContainerID: c104 HaulierID: h004	ContainerID: c105 HaulierID: h005	ContainerID: c103 HaulierID: h003
0130			

Optimised Solution

0000	ContainerID: c101 HaulierID: h001	ContainerID: c104 HaulierID: h004	ContainerID: c106 HaulierID: h006
0030	ContainerID: c105 HaulierID: h005	ContainerID: c104 HaulierID: h004	ContainerID: c106 HaulierID: h006
0100	ContainerID: c105 HaulierID: h005	ContainerID: c103 HaulierID: h003	ContainerID: c106 HaulierID: h006
0130	ContainerID: c105 HaulierID: h005	ContainerID: c103 HaulierID: h003	ContainerID: c102 HaulierID: h002

Legend

Jobs	Time Required (hours)	Non optimized solution	Optimized solution
ContainerID: c101 HaulierID: h001	0.5	allocated	allocated
ContainerID: c102 HaulierID: h002	0.5	allocated	allocated
ContainerID: c103 HaulierID: h003	1.0	allocated	allocated
ContainerID: c104 HaulierID: h004	1.0	allocated	allocated
ContainerID: c105 HaulierID: h005	1.5	allocated	allocated
ContainerID: c106 HaulierID: h006	1.5	Not allocated	allocated

Thanks for your attention!

References:

1. https://developers.google.com/optimization/bin/multiple_knapsack
2. https://slidesgo.com/