ISyE 4133 Project Report

Group 6: Kevin Dee, Quentin Mot, Jake Paupe, Lea Seaberg

2023-04-23

(a)

See the attached csv named "PartA.csv" without the quotation marks.

(b)

Note that the start/end location is considered as item 0.

The data is the following:

$$c_j := \text{cost of item j } \forall j \in [[1, 56]]$$

$$d_{ij} := \text{ Time to go from item i to item j } \forall (i,j) \in [[0,56]]^2$$

$$C := 15 \text{ The capacity of the shopping cart}$$

$$T := 90 \text{ The time the shopper has to shop}$$

The optimization model has the following objective function, constraints, and decision variables.

Objective function:

$$\max \sum_{i=0}^{56} \sum_{j=1}^{56} c_j x_{ij}$$

Constraints, with an English description:

1. The shopper can spend at most T time traveling to and picking up items:

$$(1) \sum_{i=0}^{56} \sum_{j=0}^{56} d_{ij} x_{ij} + \sum_{i=0}^{56} \sum_{j=1}^{56} 2x_{ij} \leq T$$

2. The shopper can pick up at most C items:

$$(2) \sum_{i=0}^{56} \sum_{j=1}^{56} x_{ij} \leq C$$

3. The shopper must leave the start location once:

$$(3) \sum_{j=1}^{56} x_{0j} = 1$$

4. The shopper must enter the start location once:

$$(4) \sum_{i=1}^{56} x_{i0} = 1$$

5. The shopper cannot visit the same item it leaves:

$$(5) \sum_{i=1}^{56} x_{ii} = 0$$

6. Each item may be visited at most once:

$$(6) \sum_{i=1}^{56} x_{ij} \leq 1, \forall j \in [[1, 56]]$$

7. Each item may be left at most once:

$$(7)\sum_{j=1}^{56} x_{ij} \leq 1, \forall i \in [[1,56]]$$

8. Each item that is visited must be left:

$$(8)\sum_{i=0}^{56} x_{ij} - \sum_{k=0}^{56} x_{jk} = 0, \forall j \in [[1,56]]$$

9. No subtours which do not contain the start location are allowed. More precisely, any subset of locations S that is not the empty set and does not contains 0 can have at most $|S|-1$ nonzero x_{ij} variables:

$$(9)\sum_{i\in S}\sum_{j\in S} x_{ij} \leq |S|-1, \forall S \subseteq [[1,n]] | S \neq \emptyset$$

Decision variables:

$$x_{ij} \in \{0,1\}, \forall (i,j) \in [[0,56]]^2$$

(c)

See the attached Python files for our solution to the model using the lazy constraints approach. The pseudocode for how to determine if our solution contains an undesirable subtour is below:

1. Initialize a model m as in part (b) except without constraint (9).
2. Solve m. Let s equal the optimal solution to m.
3. If there exists a subtour that does not include the start location, i.e. a set of decision variables all equal to 1 $D := \{x_{ij}, x_{jk}, ..., x_{zi}\}$ such that $D \cap Z = \emptyset$ where $Z := \{x_{0j} \forall j \in [[0, 56]]\}$, continue. Otherwise, s is optimal so return s.
4. Add the constraint $\sum_{d \in D} d \leq |D| - 1$ to m.
5. Go to step 2.

(d)

The optimal path is:

$$0 \to 21 \to 22 \to 27 \to 26 \to 39 \to 38 \to 32 \to 33 \to 34 \to 35 \to 30 \to 25 \to 5 \to 2 \to 1 \to 0$$

The items picked, in order, are:

Start Node → Ibuprofen → Diapers → Paper Towels → Toilet Paper → Redbull → Gatorade → Detergent → Broom → Dog Treats → Air Freshener → Trash Bags → Shampoo → Granola → K-Cups → Coffee Beans → Start Node

The total value of those items is \$143.85.

During training, specifically while solving parts (e) and (f) we noticed our code was taking a long time to run. To deal with the issue, we eliminate subtours containing 0 whose objective value is strictly less than the solution to a model (called the incumbent in the code) with smaller capacity or shorter time, as applicable.

More formally, let $X := \{x_{0i}, x_{ij}, ..., x_{z0}\}$ and let o be the objective value of a previous solution. Then, if the objective function in part (b) evaluated on the variables in X is strictly less than 0, we eliminate X as a subtour.

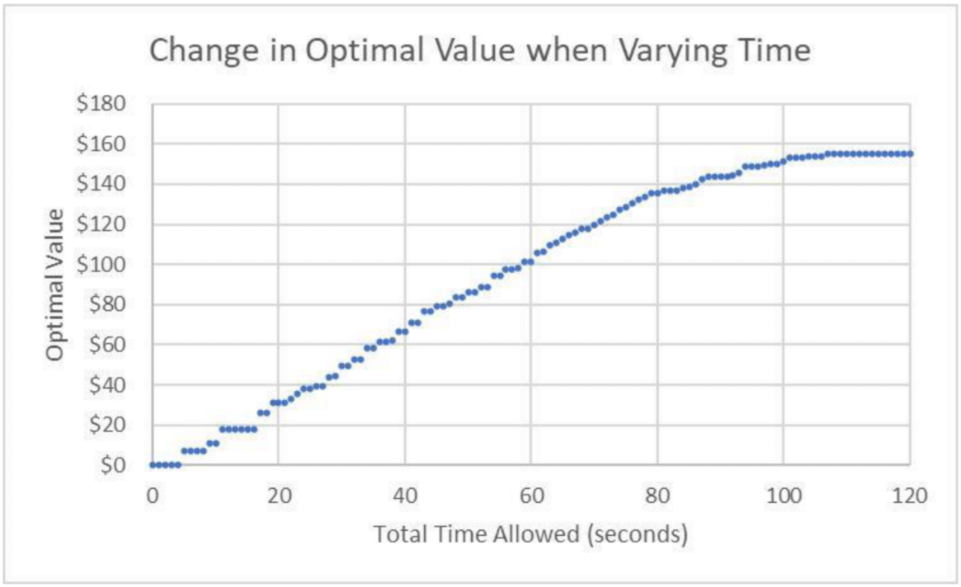
We discussed this solution with Professor Dahan.

See the Python file for the code which implements that functionality.

(e)

With low time, the shopper can only pick up items that are near the start node. Then, as time increases, the shopper can pick up the (potentially higher value) items that are farther from the start node. This continues until the capacity C of the shopping cart is reached with the shopping cart containing the top C highest value items. Specifically, with $C := 15$ the optimal solution has an objective value of 154.85, which is precisely the sum of the values of the 15 most expensive items. Hence, the optimal value of the Supermarket Sweet IP increases approximately linearly with respect to time.

See the graph below:

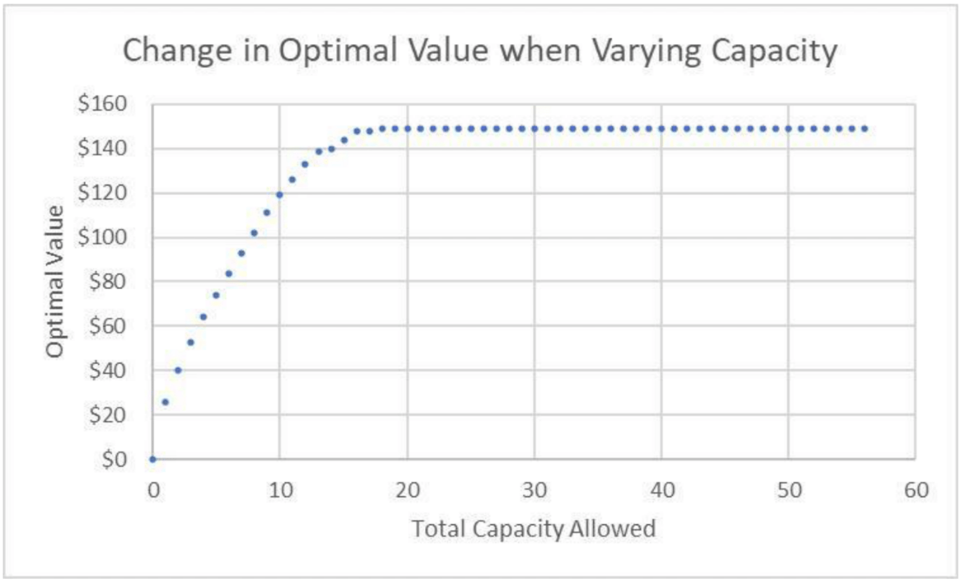

Change in Optimal Value when Varying Time
Optimal Value
$180
$160
$140
$120
$100
$80
$60
$40
$20
$0
0
20
40
60
80
100
120
Total Time Allowed (seconds)

#

(f)

With low capacity, the shopper picks up the highest value item it can reach in the allotted time. Then, as capacity increases, the shopper picks up more items in order of decreasing value until the cart contains as many of the high value items it can reach in the allotted time. Hence, the optimal value of the Supermarket Sweep increasing approximately like the cumulative distribution function of the exponential distribution.

See the graph below:


Change in Optimal Value when Varying Capacity
Optimal Value
$160
$140
$120
$100
$80
$60
$40
$20
$0
0
10
20
30
40
50
60
Total Capacity Allowed

#