WonderMarket
Section A – Internal Report

Kenton Lam

MATH3202 Assignment 3
Due 27/05/2019 1:00 pm

Abstract

Our long-time client WonderMarket has approached regarding their foray into the refridgerator space. Competition in the area is fierce and they need us to optimise their fridge logistics. Given information about their costs and requirements, we created and optimmised a dynamic programming model using Python. This report describes the model and its solution.

Model Definition

WonderMarket wishes to start selling fridges. For the time being, they are focusing on 3 bespoke fridge options—named "Alaska", "Elsa" and "Lumi".

Data

The following data has been provided to us by WonderMarket.

F	the set of fridge types.
Profit_f	profit made by selling one of fridge f.
$\text{Expected}_{f,n}$	expected number of fridge f sold if n are displayed (comm 1 only). $(0 \leq n \leq 4)$.
$\text{DemandProbs}_{f,n}$	probability that n units of fridge f will be sold (comm 2). $(1 \leq n \leq 6)$.
StoreCost	cost of storing one fridge for one week (comm 2).
FridgesPerTruck	maximum fridges transported by one truck (comm 3).
TruckCost	cost of one truck (comm 3).
MaxTrucks	maximum number of trucks per week (comm 3).
MaxStore	maximum number of each fridge type handled per week (comm 3).

Communication 1

WonderMarket wishes to display fridges. They can display up to 8 fridges and the amount of each fridge bought depends on how many fridges of that type are displayed.

Here, the stages are the fridge type and the state is the number of fridges remaining. The action describes the number of the current fridge to display. We write

$$f = \text{current fridge}$$
$$r = \text{remaining fridges}$$
$$A = \{0, \ldots, 4\} = \text{all possible number of fridges display}$$

Mathematically, we can write the value function as

$$V_f(r) = \max_{a \in A \mid a \leq r} \left\{ \text{Profit}_f \times \text{Expected}_{f,a} + V_{f+1}(r-a) \right\}$$

with the base cases of

$$V_3(r) = 0, \qquad V_f(0) = 0 \qquad \forall r, f.$$

To optimise for WonderMarket, we computed $V_0(8)$ which will iterate through the stage of each fridge (0 to 2) and with 8 available fridges. This resulted in a maximum profit of \$1863.20 which is achieved by displaying 2 Alaska fridges, 3 Elsa fridges and 3 Lumi fridges.

Note that we assumed at most 4 of a fridge type would be displayed. This is reasonable as any fridges more than 4 do not increase expected sales. Therefore, it will always be better to display some fridges of another type than more than 4 fridges of one type.

Communication 2

In this communication, the focus moved from direct-to-customer sales to delivery. WonderMarket is running a 4 week trial of their refridgerator branch and needs to order fridges to ship to their customers. There are costs and constraints associated with storing fridges during this process.

In addition, they cannot know the exact demand ahead of time so require us to optimise for maximum expected profit.

We can consider each fridge independently of the others, so we write a value function for one specific fridge then combine them later. For each fridge, the stage is number of weeks elapsed, state is number of fridges stored at the start of that week and actions represent the number of fridges bought. We also have a set of possible demands.

We write

$$\begin{aligned} t &= \text{weeks elapsed} \\ s &= \text{fridges of type } f \text{ stored at start of week} \\ A &= \{0, \ldots, 6\} = \text{all possible number of fridges to order} \\ D &= \{1, \ldots, 6\} = \text{all possible demands} \end{aligned}$$

The value function for a particular fridge can be written as

$$\begin{aligned} V_{f,t}(s) = \max_{a \in A} \Big\{ & -\text{StoreCost} \times (s+a) \\ & + \sum_{d \in D} \left(\text{DemandProbs}_{f,d} \times [\text{Profit}_f \times \min(d, s+a) + V_{f,t+1}(s+a-\min(n, s+a))] \right) \Big\} \end{aligned}$$

with base case $V_{f,4}(s) = 0$. This functions was memoized using Python's lru_cache decorator. We then consider all fridges using

$$V_t(a, e, l) = V_{0,t}(a) + V_{1,t}(e) + V_{2,t}(l).$$

Computing $V_0(0, 0, 0)$ returns \$5282.92 as maximum expected profit which is obtained by ordering 4, 5, 5 of Alaksa, Elsa and Lumi fridges respectively in the first week. Because this is a stochastic dynamic programming problem, the complete list of optimal actions cannot be known in advance. You can use the attached stochastic_explorer.py to explore the optimal actions given certain demands in each week.

Note that the summation computes an expected value over all demands. $\min(n, s+a)$ ensures that the number of fridges sold is limited by the smaller of n or $s+a$, the demand and number of fridges available respectively.

We assumed at most 6 of a fridge type can be ordered per week. This is reasonable because at most, 6 fridges are sold per week. If 7 fridges are ordered, one would always be left over to next week. However, it is cheaper to only order 6 then buy one next week if required. Also, because WonderMarket is only just beginning their fridge trial, we assume they have no fridge stock stored.

Communication 3

In addition to the requirements of communication 2, WonderMarket needs to consider the logistics of transporting fridges to their warehouse. At most, 7 fridges (possibly of mixed type) fit on a truck and at most 2 trucks can be ordered per week. Each truck costs a flat fee of $150. In addition, at most 8 fridges of each type can be in the warehouse at a time. This results in a maximum of 14 fridges ordered per week.

Again, the stage is number of weeks elapsed. Our state will now be a vector of fridges of each type currently stored. Similarly, the action is number of each fridge bought in the current week and the set of demands now considers all 3 fridge types.

Let $\sum \mathbf{x}$ be the sum of elements of the vector $\mathbf{x}$. We write

$t =$ weeks elapsed

$\mathbf{s} =$ fridges stored at start of week

(given $\mathbf{s}, s_f =$ fridges of type f stored)

$A = \left\{\mathbf{a} \in \{0, \ldots, 14\}^3 \mid \sum \mathbf{a} \leq 14\right\} =$ all possible permutations of fridges to order

(given $\mathbf{a} \in A, a_f$ is number of fridge f ordered)

$D = \{1, \ldots, 6\}^3 =$ all possible demands

(given $\mathbf{d} \in D, d_f$ is demand of fridge f)

To keep the notation neat, we define the element-wise minimum of two n-dimensional vectors as

$$\operatorname{emin}(\mathbf{v}, \mathbf{u}) := \left(\min\{v_1, u_1\}, \min\{v_2, u_2\}, \ldots, \min\{v_n, u_n\}\right).$$

Also note that Profit can be treated as a 3-dimensional vector. Then, we write the value function as

$$\begin{aligned} V_t(\mathbf{s}) = \max_{\substack{\mathbf{a} \in A \\ \max(\mathbf{s}+\mathbf{a}) \leq \mathrm{MaxStore}}} \Bigg\{ & -\mathrm{StoreCost} \times \sum(\mathbf{s}+\mathbf{a}) \\ & -\mathrm{TruckCost} \times \left\lceil \frac{\sum \mathbf{a}}{\mathrm{FridgesPerTruck}} \right\rceil \\ & + \sum_{\mathbf{d} \in D} \Big[\Big(\prod_{f \in F} \mathrm{DemandProbs}_{f,d_f} \Big) \times \Big(\mathrm{Profit} \cdot \operatorname{emin}(\mathbf{d}, \mathbf{s}+\mathbf{a}) \\ & \qquad + V_{t+1}(\mathbf{s}+\mathbf{a}-\operatorname{emin}(\mathbf{d}, \mathbf{s}+\mathbf{a})) \Big) \Big] \Bigg\} \end{aligned}$$

More Notation

For a vector $\mathbf{v}$, let v_i be the i-th component of the vector. We write the element-wise minimum of two n-dimensional vectors as

$$\operatorname{emin}(\mathbf{v}, \mathbf{u}) = (\min\{v_1, u_1\}, \min\{v_2, u_2\}, \ldots, \min\{v_n, u_n\})$$

Note that Profit is a 3-dimensional vector.

In our value function, we need to consider all possible permutations of demands of each fridge type. We write the set of demand permutations as

$$\mathrm{DemandPerms} = \{1, 2, \ldots, 6\}^3$$

and denote the probability of each permutation $\mathbf{d} \in \mathrm{DemandPerms}$ as

$$\mathrm{Prob}_{\mathbf{d}} = \prod_{f \in F} \mathrm{DemandProbs}_{f, d_f}.$$

Value Function

The value function represents the maximum expected profit given a starting time and state. It can be expressed as $V_4(\mathbf{s}) = 0$, then for $t < 4$,

$$\begin{aligned} V_t(\mathbf{s}) = \max_{\substack{\mathbf{a} \in \mathrm{Actions} \\ \max(\mathbf{s}+\mathbf{a}) \le \mathrm{MaxStore}}} \Bigg\{ & -\mathrm{StoreCost} \times \sum_{f \in F} (s_f + a_f) \\ & - \mathrm{TruckCost} \times \left\lceil \frac{\sum_{f \in F} a_f}{\mathrm{FridgesPerTruck}} \right\rceil \\ & + \sum_{\mathbf{d} \in \mathrm{DemandPerms}} \mathrm{Prob}_{\mathbf{d}} \times \Big(\mathrm{Profit} \cdot \operatorname{emin}(\mathbf{d}, \mathbf{s} + \mathbf{a}) \\ & \qquad + V_{t+1}(\mathbf{s} + \mathbf{a} - \operatorname{emin}(\mathbf{d}, \mathbf{s} + \mathbf{a})) \Big) \Bigg\} \end{aligned}$$

Derived Data

To simplify later calculations, we derived some data from the data given above.

$\mathrm{SurgeMultiplier}_{u,s}$	surge demand at store s during surge u, divided by regular demand at store s. $\mathrm{SurgeMultiplier}_{u,s} = \mathrm{SurgeDemand}_{u,s} / \mathrm{Demand}_s$
$\mathrm{NormalWeeks}$	number of weeks in the year which have no surge scenario. $\mathrm{NormalWeeks} = 52 - \sum_{u \in U} \mathrm{SurgeWeeks}_u$

Variables

The following variables were used in the Gurobi model.

B_d	binary variables for whether DC d is active. (1 means if d is new then d is built; if d already exists then d is not closed).
P_d	integer number of part-time teams employed at DC d year-round.
F_d	integer number of full-time teams employed at DC d year-round.
$C_{u,d}$	integer number of casual employeed employed at DC d during surge u. (each casual employee is only employed for the duration of the surge).
$A_{d,s}$	binary variable for whether DC d delivers to store s.
$X_{d,s}$	integer truckloads to be sent from DC d to store s during normal demand.
$Y_{d,s,u}$	integer truckloads to be sent from DC d to store s during surge scenario u. (X and Y will be 0 or exactly match demand, so will be integers.)

Objective

This calculates the total yearly cost, considering transport and labour costs. The objective is to *minimise* the following function.

$$\begin{aligned}
&\text{(normal transport cost)} && \sum_{s\in S}\sum_{d\in D} \mathrm{NormalWeeks}\cdot \mathrm{Cost}_{d,s} X_{d,s} \\
&\text{(surge transport costs)} && +\sum_{u\in U}\sum_{s\in S}\sum_{d\in D} \mathrm{SurgeWeeks}_u \mathrm{Cost}_{d,s} Y_{d,s,u} \\
&\text{(full/part-time labour costs)} && +52\sum_{d\in D} \mathrm{PTCost}\, P_d + 52\sum_{d\in D} \mathrm{FTCost}\, F_d \\
&\text{(casual labour costs)} && +\sum_{u\in U}\sum_{d\in D} \mathrm{CasualCost}\, \mathrm{SurgeWeeks}_u C_{u,d}
\end{aligned}$$

Constraints

First, we have the constraint that all variables are non-negative and certain variables are integers or binary. For all $d \in D,\ s \in S,\ u \in U$, we have

$$\begin{aligned}
B_d, P_d, F_d, C_{u,d}, A_{d,s}, X_{d,s}, Y_{d,s,u} &\geq 0 \\
B_d, A_{d,s} &\in \{0,1\} \\
P_d, F_d, C_{u,d} &\in \mathbb{Z}
\end{aligned}$$

We link X and Y variables via A and the known demand at each store. This ensures each store receives all its supplies from one DC.

$$X_{d,s} = \mathrm{Demand}_s A_{d,s} \qquad \forall\ d \in D,\ s \in S$$

We link X and Y via the surge multipliers data. This ensures for each store, it receives deliveries from the same DCs in each scenario as during normal demand.

$$Y_{d,s,u} = \mathrm{SurgeMultiplier}_{u,s} X_{d,s} \qquad \forall\ d \in D,\ s \in S,\ u \in U$$

We ensure the solution is valid for normal demand considering truckloads and capacities. Note that the northside capacity limit has been removed.

$$\begin{aligned}
\sum_{d\in D} X_{d,s} &\geq \mathrm{Demand}_s && \forall\ s \in S \\
\sum_{s\in S} X_{d,s} &\leq \mathrm{Capacity}_d && \forall\ d \in D
\end{aligned}$$

Ensuring at most 2 new DCs are built and there are 4 DCs in total (3 or 2 old).

$$\sum_{d \in \text{NewDCs}} B_d \leq 2$$
$$\sum_{d \in D} B_d = 4$$

Ensuring enough teams are employed to meet normal demand at each DC.

$$\sum_{s \in S} X_{d,s} \leq \text{FTCapacity}\, F_d + \text{PTCapacity}\, P_d \qquad \forall\ d \in D$$

We ensure our assignments can scale up to each surge scenario while remaining feasible with the given constraints. For each $u \in U$,

$$\sum_{d \in D} Y_{d,s,u} \geq \text{SurgeDemand}_{u,s} \qquad \forall\ s \in S$$
$$\sum_{s \in S} Y_{d,s,u} \leq \text{Capacity}_d \qquad \forall\ d \in D$$
$$\sum_{s \in S} Y_{d,s,u} \leq \text{FTCapacity}\, F_d + \text{PTCapacity}\, P_d + C_{u,d} \qquad \forall\ d \in D$$

Solution

Solving this MILP model in Gurobi returns the following solution. This would cost WonderMarket $12576018.00 each year ($241846.50 per week).

Store Assignments

Store	DC0	DC1	DC2	DC3	DC4	DC5	DC6
S0				100.0%			
S1				100.0%			
S2				100.0%			
S3						100.0%	
S4				100.0%			
S5		100.0%					
S6		100.0%					
S7			100.0%				
S8			100.0%				
S9				100.0%			

Distribution Centres

WonderMarket should close DC0 and build DC3 and DC5. DC1 and DC2 should remain open.

Labour

WonderMarket should hire the following teams for the whole year.

DC	Part-time	Full-time
DC1	0	2
DC2	0	2
DC3	0	8
DC5	0	2

Additionally, they should hire the following casual staff for certain surges.

Surge	DC	Casual
Surge 0	DC5	11
Surge 1	DC2	20
Surge 3	DC2	20
Surge 4	DC1	42
Surge 4	DC3	2