OPTIMIZATION OF PROFITS ON MULTI-WAREHOUSES VEHICLE ROUTING PROBLEM

Problem B

Ruolin Xiong, Hao Xue, Yan Nie

December 21, 2017

Abstract

Effective routing is an important and challenging task for the management of logistic industry. In this task, proper delivery routing and production plan are required to meet the demands of each shop in a timely and cost-effective fashion. Since there are two warehouses where capacity limited vehicles are housed, this problem can be characterized as a multi-depot vehicle routing problem (MDVRP).

To provide plausible solutions to this problem, we 1) predict the future sales and demands according to recorded data by uniform distribution; 2) formularize the problem mathematically by an objective function with constraints; 3) use Clark and Wright Savings Heuristic to generate feasible solution to the objective function proposed in step 2, in a straightforward but effective way; 4) approximate global optimum by Simulated Annealing; 5) simulate the change of demand and sale over one year to calculate the total profit; 6) apply Monte Carlo method to estimate the mean and standard deviation of annul profit to describe its distribution.

With this method, we are able to design routes that fullfill the demand of each shops with local minimal cost in a fixed time. We can also determine the daily delivery plan of each warehouse with respect to number of items and dates.

Key words: MDVRP; Clarke and Wright Saving Heuristic; Simulated Annealing; Objective Function

Contents

1 Introduction

1.1 Background

Logistics industry is vital for modern society since it ensures continuously flow of material goods from enterprises' plants to its warehouses and then further delivery to each sales shops. At present, the increasing logistic costs show great potentialities of improvements on the transportation quality. Therefore, the Vehicle Routing problem (VRP) arises to be a challenging combinatorial optimization tasks to seek the optimal routes of delivery or the collection from depots to its customers in this region, simultaneously satisfying carrying capacity and time constraints.

The Multi-Depot Vehicle Routing Problem (MDVRP), an extension of classical VRP, is an NP-hard problem for determining the routes for vehicles from multiple depots to customers and then returning back. During the past few decades, an increasing number of optimization techniques based on operations research are proposed, and together raised some implementation strategies of metaheuristics such as Simulated Annealing (SA).[3] These contribute much to the effective management of the provision of goods and services in distribution systems.

1.2 Restatement of the Problem

An industry owns two plants, two warehouses and twenty shops. Products (air-conditioners and heaters) are manufactured in plants and stored in warehouse. Plants and warehouses show no differences while dealing with different products. Products will eventually be delivered to shops according to customer demands.

- Location

  The industry owns 2 plants, 2 warehouse and 20 shops, all on a connected transport network shown as bellow:

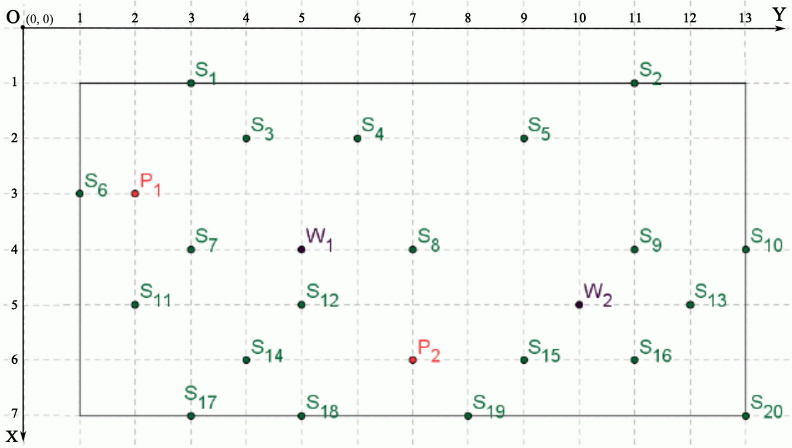

Figure 1: P=plant, W=warehouse, S=shop

Each segment of the small squares represents 1 kilometer. Vehicles should follow these line segments while traveling to different points.

- Sales

Price of two products: air-conditioners, 300 Euros; heaters, 100 Euros. All shops open six days a week, whose sales volume in 2015 is given. For the upcoming year, the sales of each shop are predicted to be within ±10% of the observed sale of the same time in 2015, following a uniform distribution.

- Storage Capacity

For each warehouse, the storage capacity of each product is fixed at 650 items. The warehouse is expected to be restocked once a week.

For each shop, its storage capacity is limited by the table shown below. According to this table we could expect that every shop is to be restocked on a daily basis:

Shop	P1	P2	Shop	P1	P2
1	15	15	11	30	30
2	15	15	12	30	30
3	20	20	13	30	30
4	20	20	14	35	35
5	15	15	15	30	30
6	20	20	16	40	40
7	30	30	17	25	25
8	30	30	18	20	20
9	35	35	19	15	15
10	25	25	20	20	20

Figure 2: Storage capacity for each shop

- Delivery tour

  The garages are inside the two warehouses. Deliveries are done by trucks with a volume capacity of 20 m^3. One heater occupies 0.4 m^3 and an air-conditioner occupies 0.8 m^3. The cost on road is 1 Euro per km, which is the only cost of our concern.

- **Questions**

Our aim is to organize a production and delivery planning. We will prioritize our concern on the delivery process from warehouses to shops. This problem is divided into several sub-problems to tackle:

1. Prediction of the weekly production of two products
2. Decide on each day and how many items should be transported to each shop
3. Define the truck rooting for each specific day
4. Find the probability law of the expected profit over one year
   1.0

In the whole modeling process, we give full consideration to maximize the expected profit for the industry over one year.

2 Model preparation

2.1 Assumptions

For our model, there are several assumptions here:

- Each plant could produce enough products as required.
- Two warehouses have enough space to store all products needed by twenty shops in the following week.
- At the beginning of one day, the replenishment process is always before the delivery tour, which means the first truck loads adequate goods and leaves for shops after all items have been unloaded and stored in the warehouses.

2.2 Definitions and Variables

Sets

I - Set of all warehouses

J - Set of all shops

K - Set of all trucks

Indices

i - warehouse index

j - shop index, ($j1$ - for product1,$j2$ - for product2)

k - route index

Parameters

N - Number of trucks

C_{ij} - Distance between point i and j, $i, j \in I \cup J$

V_1 - Volume of product 1

V_2 - Volume of product 2

W_i - Capacity of warehouse i

d_j - Demand (Sales) of shop j

Q_k - Capacity of one route (truck) k

Decision variables

$$x_{ijk} = \begin{cases} 1, & \textit{if } i \textit{ immediately preceeds } j \textit{ on route } k \\ 0, & \textit{otherwise} \end{cases}$$

$$z_{ij} = \begin{cases} 1, & \textit{if shop } j \textit{ is allotted to warehouse } i \\ 0, & \textit{otherwise} \end{cases}$$

3 Methodology

3.1 Prediction model

3.1.1 Analysis of sales data

Based on given data, we can draw a 2015 p1 and p2 sales table shown as below:

week	1	2	3	4	5	6	7	8	9	10	11	12	13
p1	34	68	67	67	68	128	126	120	114	172	182	161	169
p2	86	195	181	190	198	116	122	123	110	71	72	73	77

week	14	15	16	17	18	19	20	21	22	23	24	25	26
p1	210	214	211	206	223	243	246	246	258	246	220	242	245
p2	52	36	36	34	27	8	9	8	8	1	1	1	1

week	27	28	29	30	31	32	33	34	35	36	37	38	39
p1	249	247	231	236	252	203	215	212	227	185	180	172	183
p2	6	7	7	7	11	38	35	36	32	64	74	69	65

week	40	41	42	43	44	45	46	47	48	49	50	51	52	53
p1	153	121	128	128	124	61	61	62	67	13	1	1	1	1
p2	100	114	124	113	118	201	176	180	180	246	227	245	242	152

Figure 3: Weekly sales of two products in year 2015

For the first and last week, only sales for 3 or 4 days are available. For other weeks, sales of 6 days are accessible. For any week, if we multiply the numbers with 110%. These would imply a sufficient outputs of plants for each week of the new year.

Then we can get a demand trend figure shown as below:

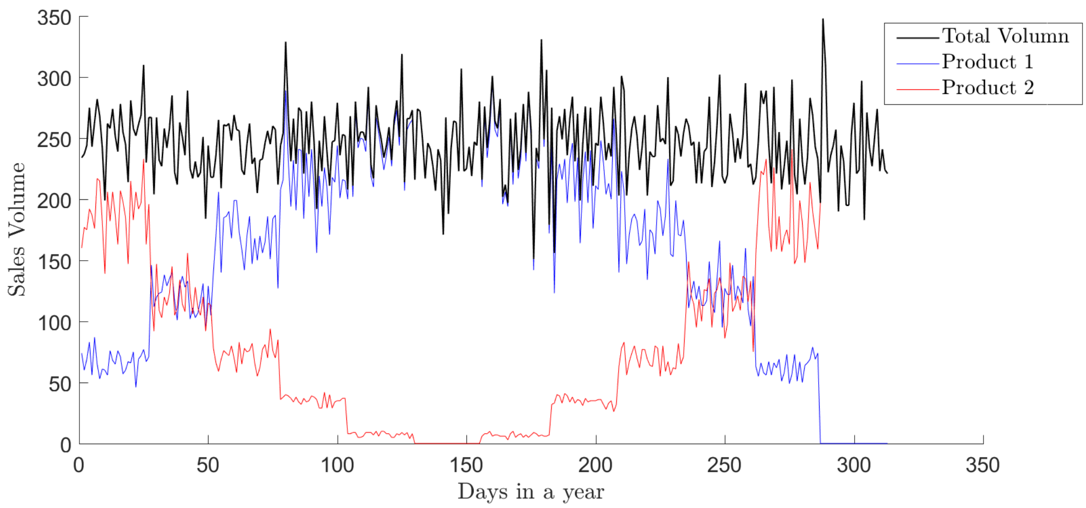

Figure 4: Demand trend of two products in year 2015

From the figure above, we can see that product 1 (air-conditioner) achieves high sales in summer while product 2 (heater) is sold more in winter. Thus, both of them are demonstrated to be seasonal. Although the given daily sales are different from one shop to another, the actual total sales per day fluctuates slightly around an intermediate value of 250. This implies the amount of work for plants tends to be relatively stable through the year.

3.1.2 Sale Prediction of 2018

From the question, we know for each day, each product and each shop, the sales will be given by a uniform law on the interval [-10%, +10%] centered at the observed sales. We get function *rand* to generate a random number in the interval [0, 1] so the interval [-0.9, 1.1] could be represented by $(\frac{rand-0.5}{5}+1)$.

$$Predicted\ Daily\ Sales = Recorded\ Daily\ Sales \times (\frac{rand - 0.5}{5} + 1) \quad (1)$$

Taking two arbitrary shops as instances, we can see the predicted daily sales of one year shown as below:

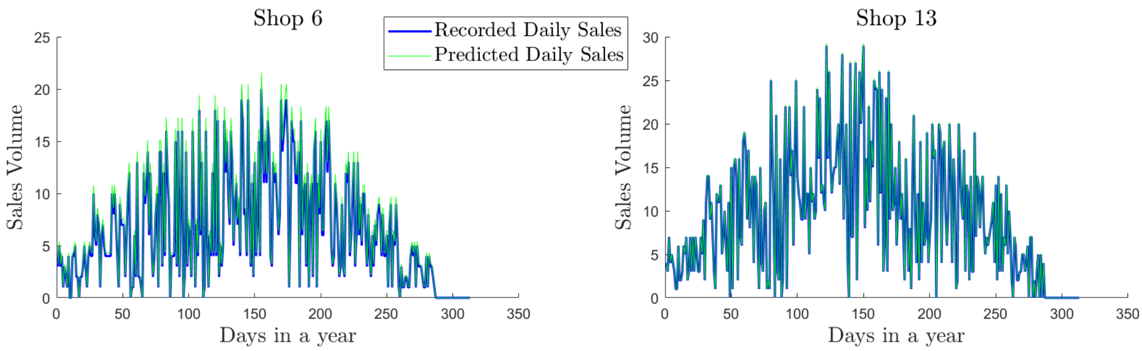

Figure 5: prediction of sales for shops over a year

For the figure above, blue lines are illustrated from the data we got, and green lines represent for casts of the coming year. Green lines are not fixed since the prediction changes with a random number.

3.1.3 Delivery Strategy

Generally, people would consider to restock shops according to the sales of next day but this might not be a wise decision. Because in that case, trucks need to carry goods to travel through all shops every day, the transportation costs could be high.

In our model, considering each shop has a capacity, our daily delivery aims to keep all shops well stocked. Therefore, except for the first day when trucks should go to every shop to replenish, trucks could merely travel to those shops whose stocks are nearly exhausted. And each time a truck arrives at a shop, it would not just replenish the sparse product but replenish both of the two products in one time. This is to ensure no shop is in short supply and each time of the delivery is made full use of.

Therefore, the condition for replenishment is *remnant inventory* $-1.1\times$ *recorded sales of the next day* <0. If this condition is satisfied, the demand is computed as

$$\textit{demand of a shop} = \textit{shop's capacity} - \textit{remnant inventory} \tag{2}$$

Otherwise, the demand for the next day will be zero.

3.2 Mathematical model

The objective function is to minimize the total distance of all trucks, that is:

$$min \sum_{i\in I\cup J}\sum_{j\in I\cup J}\sum_{k\in K} C_{ij}x_{ijk} \tag{3}$$

Every shop must be assigned to a single route, shown below:

$$\sum_{k\in K}\sum_{i\in I\cup J} x_{ijk} = 1, \quad j\in J \tag{4}$$

Each shop can be restocked at most once a day, the equation is

$$\sum_{i\in I}\sum_{j\in J} x_{ijk} \le 1, \quad k\in K \tag{5}$$

Because the direction of moving from one shop to the other does not matter, the route graph shown latter is undirected and it can be expressed mathematically as:

$$\sum_{j\in I\cup J} x_{ijk} - \sum_{j\in I\cup J} x_{ijk} = 0, \quad k\in K, i\in I\cup J \tag{6}$$

The demand of one shop consists of the demand of product 1 (d_{j1}) and product 2 (d_{j2}). If there could exists a new variable d_j to combine these two efficiently, deliveries of the two products could be treated together. So here we find d_j.

We already know the volume of p1 and p2 is $0.8m^3$ ans $0.4m^3$. Thus

$$2V_1 = V_2 \tag{7}$$

We assume the new demand variable have the same volume with V_1 without loss of generality. So the total volume of them could have equivalent effect:

$$d_j V_1 = d_{j1} V_1 + d_{j2} V_2 \tag{8}$$

Taking $V_2 = 2V_1$,

$$d_j V_1 = d_{j1} V_1 + 2 d_{j2} V_1 \tag{9}$$

Divide both sides of the equation by V_1,

$$d_j = d_{j1} + 2 d_{j2} \tag{10}$$

Thus we get the representation for d_j. Each d_j has the same volumes as V_1 as we assumed.

The total demand(sales) of all shops of all routes is subject to the capacity of a set of trucks, hence the capacity constrain is:

$$\sum_{j \in J} d_j \sum_{i \in I \cup J} x_{ijk} \leq Q_k, \quad k \in K \tag{11}$$

According to the providing material, the sales of all shops in one week are restricted by the capacity of the sum of both warehouses:

$$\sum_{j \in J} d_j z_{ij} \leq W_i, \quad i \in I \tag{12}$$

since the assumption, the capacity of the warehouses is not under consideration, illustrates our focus, the equation actually means:

$$\sum_{j \in J} d_j z_{ij} \leq W_i < \infty, \quad i \in I \tag{13}$$

3.3 Decisions on MDVRP

Since garages are located at warehouses, we can regard warehouses as two depots. Our goal is to find the optimal route that satisfies the demand of each shop and minimize path. Hence is problem can be solved by techniques of solving Multi-Depots Vehicle Routing Problem (MDVRP), we follows the flow chart exhibiting below:

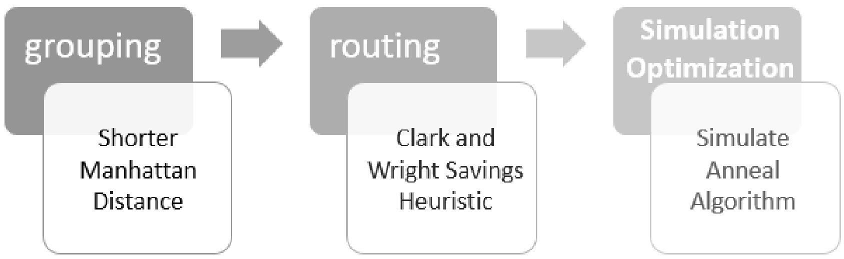

Figure 6: Decisions on MDVRP

Firstly, we perform grouping by assigning shops to the nearest warehouse according to their Manhattan distance (absolute sum of difference of coordinates) with respect to warehouses. After grouping, designing delivery routes based on daily demands of different shops is achieved by Clark and Wright Savings Heuristic. After that, shops are regrouped based on different combinations of assignments and the new groups will process the same procedure mentioned above. Meanwhile, Simulated Annealing algorithm is implemented to approximate the optimal routing. Further details are provided in the following three subsections.

3.3.1 Grouping

In a standard vehicle routing problem, there is only one depot (warehouse) which all vehicles start from and end at. In our question, there are two depots (warehouse1 and warehouse2) and shops can be served by any of the warehouses. By grouping, we can transfer multi-depot vehicle routing problem to several standard vehicle routing problems. Grouping steps are: At first, calculate the Manhattan distance between each shop and two warehouses. Secondly, for each shop, compare two distances (one to warehouse1 and one to warehouse2) and choose the shorter one. After selection, group the shops connecting to the same warehouse.

In this question, based on the coordinates of the shops, we obtain two groups. One group consists of warehouse1 and shop1, shop3, shop4, shop6, shop7, shop8, shop11, shop12, shop14, shop17 and shop 18. The other group includes warehouse2 and shop2, shop5, shop9, shop10, shop13, shop15, shop16 shop19 and shop20. This grouping can be visualized by two clusters in Figure 8:

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
Distance to W1	5	9	3	3	6	5	2	2	6	8	4	1	8	3	6	8	5	3	6	11
Distance to W2	11	5	9	7	4	11	8	4	2	4	8	5	2	7	2	2	9	7	4	5
Assignments	W1	W2	W1	W1	W2	W1	W1	W1	W2	W2	W1	W1	W2	W1	W2	W2	W1	W1	W2	W2

Figure 7: Group Result

3.3.2 Clark and Wright Savings Heuristic

Clark and Wright (1964)[2] proposed the savings concept by computing savings for combining two shops into the same route. Since 1964, the Clark and Wright Savings algorithm gradually becomes a well-known heuristic method for solving vehicle routing problem (Pichpibul and Kawtummachai, 2013)[4]. The heuristic begins by setting the depot node as the start point. We then postulate that there are n-1 trucks where n is the number of total nodes. Each truck starts from the depot straightly to a node and then returns to the depot. The distance between each shop and the depot exhibits on the directed line. Figure 9 displays this for a simple network, four-node-network.

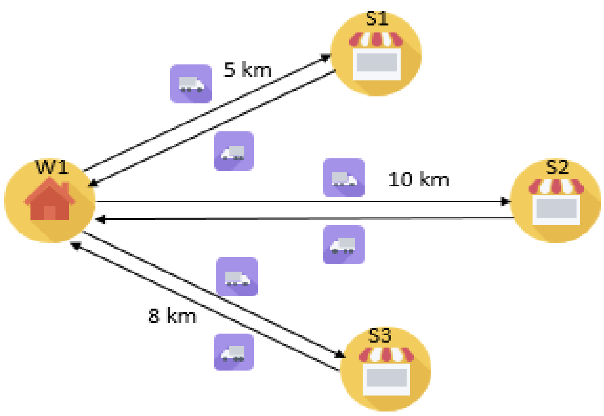

Figure 8: Clark and Wright Initial Step

For the second step in Clark and Wright Heuristic, shown in Figure 10, the solid lines are used in the first step and the dashed lines show the routes that may be used in the future. Compute the savings obtained from connecting S2 and S3: (10km+8km) - (5km) – The 10km and 8km are the distances of the return trip from S2 and S3 to the depot respectively; 5km is the length of the route between S2 and S3. In the same way, the savings of linking S1 and S2 is 12km. Similarly, for the last pair of nodes to be connected is S1 and S3 which yields a savings of 6km.

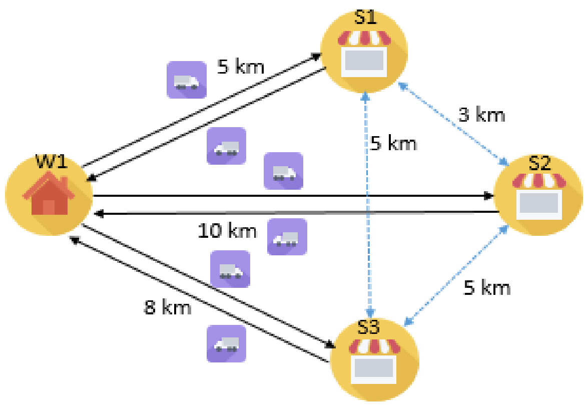

Figure 9: Computing the Savings

Next, we rank the savings for each pair of nodes which not connected yet. The first procedure to specify a route is to connect the nodes with the largest savings, that is, connecting S2 and S3. The result is shown in Figure 11 (a). The proceeding step is to seek the second largest savings - savings of linking S1 and S2, shown in Figure 11 (b)

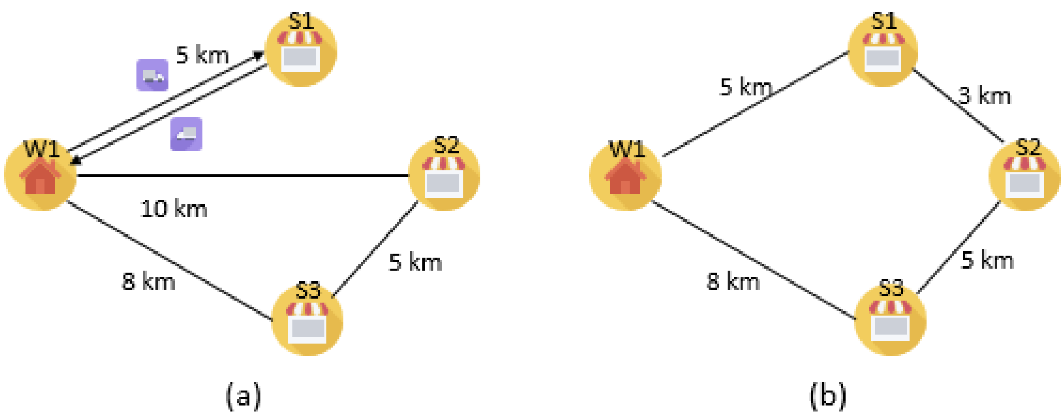

Figure 10: Rank and Substitution

In summary, the whole procedures of Clark and Wright Savings Heuristic algorithm is as follows:

1. Select the depot node and connect other nodes with the depot node
2. Compute the savings, S_{pq}, for nodes p and q:

$$S_{pq} = c_{ip} + c_{iq} - c_{pq}, \quad for \quad p, q \in J$$

   where c_{pq} = the cost of traveling from p to q

3. Rank the savings from largest to smallest
4. Beginning with the largest savings, form larger sub-routes by linking appropriate nodes p and q. Stop when a complete route is formed.

3.3.3 Simulated Annealing Algorithm

Since VRP is an NP hard problem, searching for global optimum by enumerating all possible combinations would take tremendous time, hence , we employ simulated annealing (SA), which is inspired by annealing of solids, to approximate the global optimum of our system in a fixed time interval. In SA method, an annealing schedule is specified by the temperature, which decreases as iteration proceeds. Together with the goal function of current state and future state, temperature determines an acceptance probability, which enables us to accept another less promising optimization candidate when it exceeds a random number.Therefore, SA tends to drag the optimization out of a local optimum by accepting an 'underperformed' candidate in random, however, as iteration proceeds, the acceptance probability decrease. The pseudocode is given below[1].

- Let $s = s_0$
- For $k = 0$ through $k_{\max}$ (exclusive):
  - $T \leftarrow \mathrm{temperature}(k / k_{\max})$
  - Pick a random neighbour, $s_{\mathrm{new}} \leftarrow \mathrm{neighbour}(s)$
  - If $P(E(s), E(s_{\mathrm{new}}), T) \geq \mathrm{random}(0, 1)$:
    - $s \leftarrow s_{\mathrm{new}}$
- Output: the final state s

Figure 11: Pseudocode of SA

In this case, to generate the initial state of the system, the shops are grouped according to their Manhattan distance with respect to two warehouses and assigned to the nearest warehouse. The temperature drop follows a reciprocal function k_{max}/k, where k_{max} is the maximal number of iterations and k is the number of iterations. The next state to explore is generated by randomly swapping, reversing or and reinserting an arbitrary part of the current groupings. The generation procedure is illustrated as following:

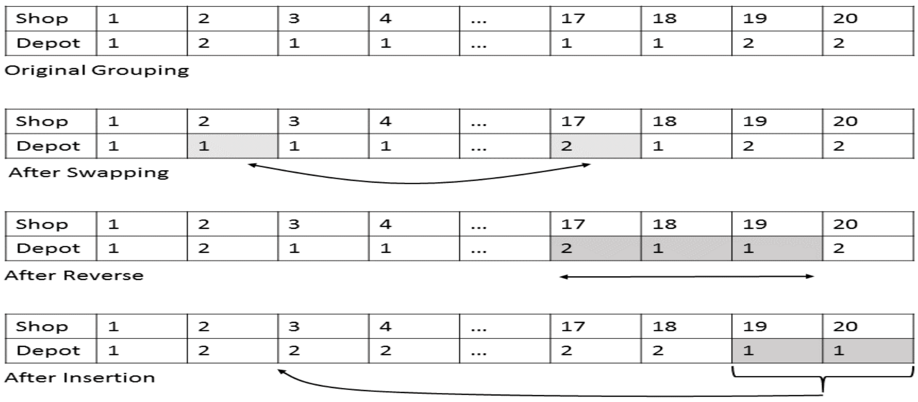

Figure 12: Illustration of Generating New States

As for acceptance probability, it is 1 when the total cost of new state c' is less than the current state c, otherwise it is computed by the formula $exp(-(c' - c)/T)$ proposed by Kirkpatrick et al.

4 Results

4.1 Decisions on daily delivery from warehouses to shops

Because our delivery strategy is always to satisfy the daily demand of different shops, the daily delivery is based on the daily demand. By using the formula mentioned in 3.1.3, we can obtain daily demands of all shops in one year. Due to the space limitation, we take first 6 days as an example, the following figure displays the demands for shops:

	Day 1		Day 2		Day 3		Day 4		Day 5		Day 6	
	p1	p2	p1	p2	p1	p2	p1	p2	p1	p2	p1	p2
s1	15	15	0	0	5	9	0	0	1	9	0	0
s2	15	15	2	7	1	8	0	0	0	0	3	10
s3	20	20	0	0	0	0	0	0	9	18	0	0
s4	20	20	1	12	0	0	0	0	5	12	2	12
s5	15	15	0	0	0	0	0	0	6	9	3	7
s6	20	20	0	0	0	0	0	0	11	17	4	10
s7	30	30	4	11	0	0	0	0	0	0	12	27
s8	30	30	0	0	0	0	0	0	0	0	0	0
s9	35	35	0	0	9	31	0	0	0	0	13	25
s10	25	25	0	0	0	0	0	0	10	22	0	0
s11	30	30	6	17	0	0	0	0	0	0	15	25
s12	30	30	0	0	8	24	0	0	3	12	0	0
s13	30	30	0	0	7	22	0	0	7	14	4	20
s14	35	35	0	0	8	17	0	0	0	18	3	16
s15	30	30	0	0	15	19	0	0	1	12	0	0
s16	40	40	0	0	0	0	0	0	0	0	0	0
s17	25	25	0	0	0	0	0	0	9	19	0	0
s18	20	20	0	0	3	10	0	0	4	12	3	15
s19	15	15	0	0	2	11	0	0	0	0	0	0
s20	20	20	4	12	0	0	0	0	0	0	0	0

Figure 13: demand of 20 shops in the first 6 days

4^{th} Jan. 2015 is a day for rest so there is no demand there. For the other days, only the first day get fully demand for all shops. Others just require several shops to have distribution.

4.2 Decisions on routes and schedules for trucks

After deciding the daily delivery of different shops, by following the process mentioned in 3.3, the result of one day, for example, displays in Figure 14:

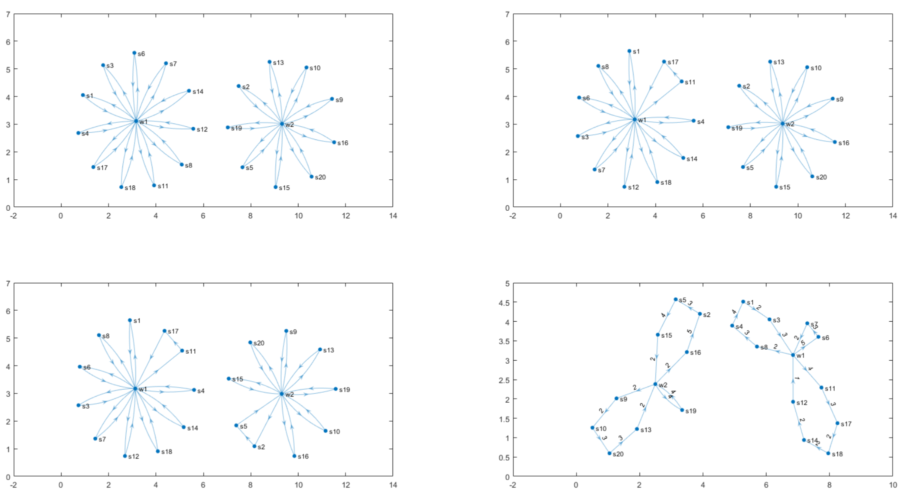

Figure 14: Decision on routes and schedules

Sub-picture in position (1,1)-the left top one- in Figure 14 demonstrates the initial step of the Clark and Wright Savings Heuristic algorithm and it can also shows the grouping result. The picture in position (1,2) and the one in position (2,1) explain the second step in each group. The one at the right bottom corner designates the result given by the Clark and Wright Savings Heuristic algorithm.

For another day, daily delivery alters, consequently the decisions on routing are totally different. To clearly illustrate this situation, another route pattern is provided.

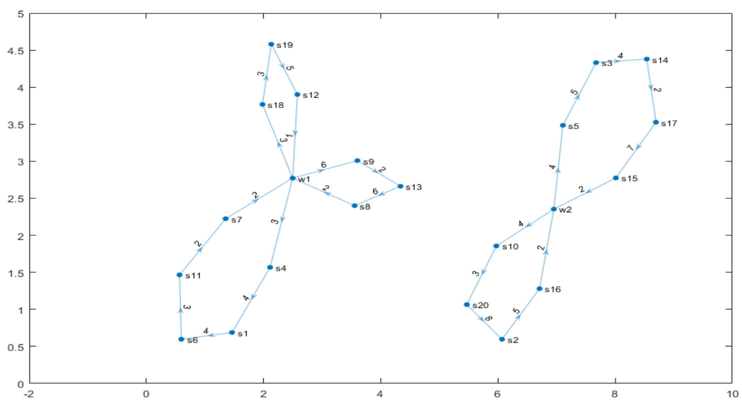

Figure 15: Route Decision on Another Day

However, it is likely that this result is the global optimum. Therefore we use simulated annealing algorithm to try to optimize the result.

To exemplify the result of SA, we sketch the curve of the accepted minimum cost with respect to number of update accepted.

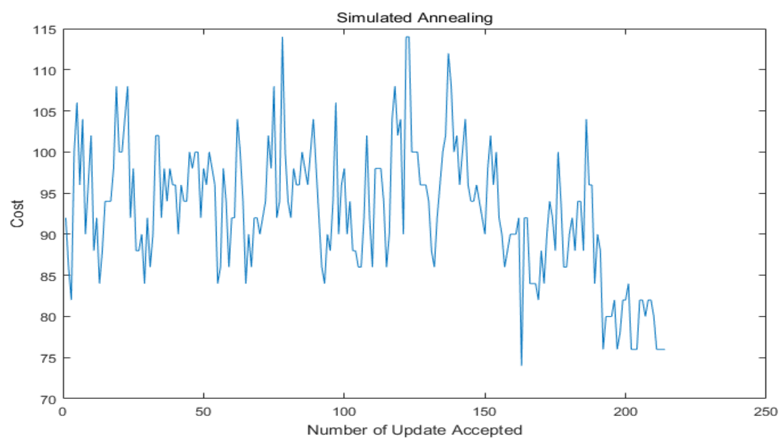

Figure 16: Illustration of Generating New States

As shown by Figure 15, SA enables current state to jump out of a local optimum by accepting a worse state with damping probability. Once iteration ends, the state associated with lowest cost will be recorded and the routing of this state will be regarded as our optimal routing.

4.3 Probability Law of Expected Profit

By using Monte Carlo Simulation, we obtained the mean and the standard deviation of expected profit over one year with 100 replicates.

$$Mean(profit) = 1.2902 \times 10^7$$

$$Std(profit) = 7.4841 \times 10^5$$

5 Strengths and Weaknesses

5.1 Strengths

- Our model covers plenty of information and serves as the fundamental step of solving the rest of questions.
- The CW Saving heuristic we employed is straightforward and easy to implement even without mathematical background. Though simple as it is, it is found one of the most effective heuristic which adequately approximate the global optimum.
- Because CW Saving heuristic and SA are so simple that our model is efficient enough to provide a good enough local optimum to NP problem in a short period, which is advantageous to be applied to a more complex system in the future.

5.2 Weaknesses

- In the model, we ignore the capacity of two warehouses and simply assume them to have enough space. However, in real case, the shops would fail to be replenished during some peak weeks because of the lack of warehouse stocks.

Under this condition, customers might get disappointed and then some part of the sales profits may lose.

- During the past few decades, researchers raised several implementation strategies of bio-inspired intelligent heuristics such as Simulated Annealing (SA), Genetic Algorithm (GA), Ant Colony Optimization (ACO) etc. The report would be more complete if more than one heuristic are tried. Comparing different results may lead us to find a relatively better algorithm or to better recognize the advantages and disadvantages of each.

- For many heuristics, including SA used in our model, it is merely possible to find a local minimum solution, which means a relative optimal path, for such kind of vehicle routing problem. But this local minimum does not necessarily converges to the global minimum. Finding the global minimum requires consideration of all the possible situations. This could be time consuming and unrealistic for the current time.

- Our algorithm shows it sensitiveness to the grouping process at the first step, but the method utilized for grouping here is rather simple, only determined by their distances.

6 Conclusion

We abstract the MDVRP from the context by regarding warehouses as depots. Techniques involving searching algorithms and operational research are applied to provide feasible solutions to this problem. Apart from providing mathematical formulas, simulation is run since the expected demand of each day of 2018 can be predicted according to observed data. To satisfy demand of each shop as well as reduce the routing cost, Clark and Wright Savings Heuristic is used to generate a feasible solution because of its simplicity and efficiency. To optimize the solution, simulated annealing method is utilized to approximate the global maximum in that this method yields a plausible local optimum within a reasonable period. In light of this model, we will be able develop delivery strategies of each day for trucks in each warehouse with a reasonable cost. This procedure can be applied to each day and eventually we can compute the expected profit over one year. Monte Carlo method is considered to compute the mean and standard deviation of annul profit, therefore making estimation of distribution of annul profit possible.

7 Further Investigations

Several questions have not been solved in our report so there still requires lots of further work. First, it remains to be classified how many items should be produced in each of the plant every day. At this time, we could only calculated the total production of them but the difference for plant 1 and plant 2 with respect to warehouse 1 is unclear. Second, triggered by the first question, the productivities of both plants should be limited. Thus, the productions of all items needs to be allocated to each plant in a reasonable way to satisfy both demands and productivities. Third, the capacity of warehouse is not as large as what we assumed. Sometimes the shops would be in short supply so the sales would not be such high. There also lacks a design of "penalty rule" to describe the situation of people get disappointed if he could not find any one shop to buy a product. All of the above needs to be refined in our further investigation.

References

[1] Emile Aarts, Jan Korst, and Wil Michiels. Simulated annealing. *Search methodologies*, pages 187–210, 2005.

[2] Geoff Clarke and John W Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.

[3] Scott Kirkpatrick, Mario P Vecchi, et al. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[4] Tantikorn Pichpibul and Ruengsak Kawtummachai. A heuristic approach based on clarke-wright algorithm for open vehicle routing problem. *The Scientific World Journal*, 2013, 2013.

Appendix

Calculate the demands of 20 shops

```
function [newDelivery,profit] = delivery(n)
dbstop if error;
analysisPath = pwd;
% open sales.xls and extract all data
dataPath = ('C:/Users/bear.pc/Desktop');
cd( dataPath );
fileName = ('sales','.xls');
dataFromFile = importdata(fileName);
textData = dataFromFile.textdata;
numData = dataFromFile.data;
cd( analysisPath );
% extract columns of p1 and p2
column_p1 = 1:2:39;
column_p2 = 2:2:40;

sales = zeros(20,2*n);
for i = 1:n
    if rem(i+3,7) ~= 0
        sales(:,2*i-1) = numData( i, column_p1 )';
        sales(:,2*i) = numData( i, column_p2 )';
    end
end
daySales = sum(sales);
sixDaySales = daySales;
sixDaySales([7:14:end,8:14:end])=[];

capacity = numData(1:20,49:50);
delivery = zeros(20,2*n);
delivery(:,1:2) = capacity;
rest = zeros(20,2*n);

for i = 1:n
```

```
    if i == 1
        rest(:,1:2) = delivery(:,1:2) - sales(:,1:2);
    else
        a = rest(:,2*i-3:2*i-2) - 1.1*sales(:,2*i-1:2*i);
        shopRestock_p1 = find( a(:,1)<0 );
        shopRestock_p2 = find( a(:,2)<0 );
        % see one day for demand situation
%           if i == 7
%               shopRestock = unique([shopRestock_p1;shopRestock_p2]);
%           end
        if isempty(shopRestock_p1) && isempty(shopRestock_p2)
            delivery(:,2*i-1:2*i) = zeros(20,2);
        else
            delivery(shopRestock_p1,2*i-1:2*i) = capacity(shopRestock_p1
            ,:) - rest(shopRestock_p1,2*i-3:2*i-2);
            delivery(shopRestock_p2,2*i-1:2*i) = capacity(shopRestock_p2
            ,:) - rest(shopRestock_p2,2*i-3:2*i-2);
        end
        % remaining stock
        rest(:,2*i-1:2*i) = rest(:,2*i-3:2*i-2) + delivery(:,2*i-1:2*i)
        - sales(:,2*i-1:2*i);
    end
end
dayDelicery = sum(delivery);

delivery(:,7:14:end)=[];
delivery(:,7:13:end)=[];
sales(:,7:14:end)=[];
sales(:,7:13:end)=[];

newSales = round(sales .* ((rand(20,626)-0.5)/5+1));
allDayNewSales = sum(newSales);
profit = allDayNewSales(1:2:end)*100 + allDayNewSales(2:2:end)*300;

```

```
newDelivery = zeros(20,313);
newDelivery = delivery(:,1:2:end)+ 2 * delivery(:,2:2:end);

% set up an figure
fI = figure( 'Color', 'w', 'InvertHardCopy', 'off', 'Position',
[ 100 100 1100 500 ] );
hold on;
plo(1)= plot( sixDaySales(1:2:end)+sixDaySales(2:2:end), '-k',
'lineWidth', 1 );
plo(2)= plot( sixDaySales(1:2:end), '-b', 'lineWidth', 0.5 );
plo(3)= plot( sixDaySales(2:2:end), '-r', 'lineWidth', 0.5 );
indsLegend = 1 : 3;
hleg = legend( plo( indsLegend ), { 'Total Volumn',...
    'Product 1',...
    'Product 2'},...
    'Location', 'Northoutside', 'fontSize', 15, 'FontWeight',
    'b', 'interpreter', 'LaTeX' );
set( hleg, 'position', [ 0.82, 0.8, 0.17, 0.07 ] );
set( gca, 'fontSize', 15 );
xlabel( 'Days in a year', 'interpreter', 'LaTeX' );
ylabel( 'Sales Volume', 'interpreter', 'LaTeX' );
set( fI, 'paperPositionMode', 'auto' );
hold off;
```