CHAPTER IV

THE SINGLE ITEM SINGLE DEPOT PROBLEM

4.1 Introduction

The problem defined in chapter III is very complex. In order to get insight in the problem this chapter presents a simplified version of the problem proposed in chapter III. The problem is simplified by consideration of single item and single depot instead of multi-item and multi-depot and a vehicle with unlimited capacity is used to make deliveries to multiple outlets. This simplified problem is important for solution approach development of the problem proposed in chapter III since it can be understood more easily.

The solution of this simplified problem is obtained by a heuristic which decomposes the main problem into two sub problems: Lot-Sizing Problem and Vehicle Routing Problem. Each of them is solved iteratively. This heuristic is developed using idea of solving Lot-Sizing Problem of each outlet. A modified algorithm for solving Lot-Sizing Problem is developed to get replenishment quantity and then Vehicle Routing Problem are iteratively solve by cheapest insertion and two-opt improvement.

The remainder of this chapter is organized as follows. Problem description is provided in Section 4.2. A solution approach will be presented in Section 4.3. The computational results are reported in Section 4.4. The conclusions of this work and some recommendations for the next are presented in Section 4.5.

4.2 Problem description and model formulation

The system which is considered in this chapter consists of a depot and multiple outlets. A depot distributes an item to outlets via a vehicle with unlimited

capacity. Each outlet faces a periodically deterministic demand. The depot works as a transshipment point without holding any inventory, therefore the inventory is carried at outlets only. The objective of managing this system is to determine replenishment quantity for all outlets during a finite period and provide delivery route for the replenishment with the minimal total inventory transportation cost. The costs, which are associated to the total cost, are inventory holding cost at outlets and distance dependent routing cost. It is assumed that all demands must be satisfied so stock out is prohibited. The illustration of this problem is shown in Figure 4.1 and the information of demand and holding cost of each outlet is shown in Table 4.1 and 4.2.

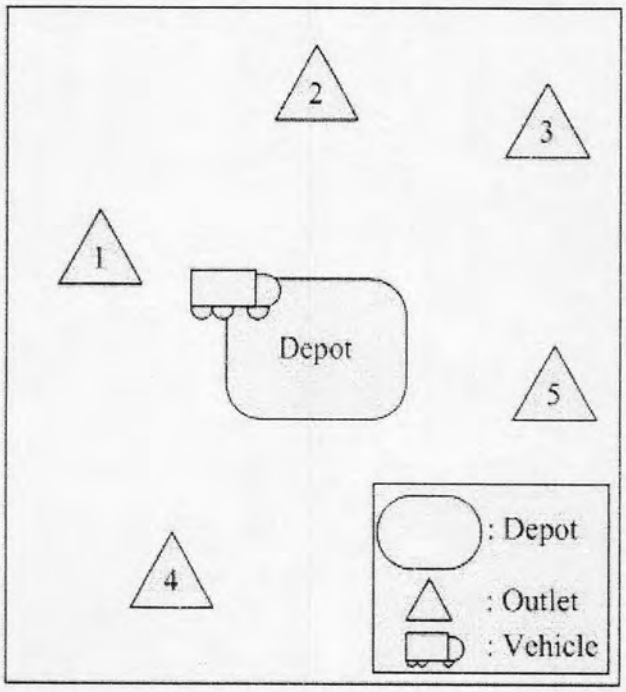


Figure 4.1 Illustration of the single-item single-depot inventory routing problem.

Table 4.1 Demand of all outlets

Period / Outlet	1	2	...	p
1	12	8	...	10
2	11	9	...	10
3	10	8	...	9
4	8	10	...	11
5	9	11	...	10

Table 4.2 Holding cost of all outlets

Outlet	Holding Cost
1	0.20
2	0.30
3	1.70
4	0.70
5	1.50

This problem simplifies the problem proposed in chapter III by considering a depot instead of several one. Moreover, both storage capacity at outlets and carrying capacity of vehicle is unlimited in this problem. However, the no stock out policy is still held in this problem. Another difference is the objective function. This problem does not consider the fixed-charge cost so there are only inventory holding cost and routing cost included in objective function. The model formulation of this simplified problem is shown as follow.

Decision Variable

I_i^t	Inventory level of Outlet i at period t.
o_i^{tk}	1 if Order Outlet i at period t cover demand from period t to k-1, and 0 otherwise
a_{ij}^t	1 if there is an arc for delivery from Node i to Node j in period t, and 0 otherwise
f_{ij}^t	Flow of Item from Node i to Node j at period t.

Set

DP={0}	Depot Set
O={1..n}	Outlet Set
N=DP $\cup$ O	Node Set :consists of Outlet and Depot
P={1..p}	Period Set

Parameter

H_i	Holding cost of Outlet i.
C_{ab}	Distance between Node a and b.
B_i	Beginning Inventory level of Outlet i.

D_i^t Demand of Outlet i at period t.

$CumD_i^{tk}$ Demand of Outlet i at from period t to k-1.

The model of single-item single-depot inventory routing problem (SSIRP) can be formulated as follows:

Objective function

$$\text{Minimize} \quad \sum_{i\in I}\sum_{t\in P} H_i I_i^t + \sum_{i\in N}\sum_{j\in N}\sum_{t\in P} C_{ij} a_{ij}^t \tag{0}$$

Constraints

$$I_i^t = I_i^{(t-1)} + CumD_i^{tk} o_i^{tk} - D_i^t \qquad \forall i \in O, t \in P \tag{1}$$

$$\sum_{j\in P} o_i^{jt} = \sum_{k\in P\cup\{p+1\}} o_i^{tk} \qquad \forall i \in O, t \in P\text{-}\{1\} \tag{2}$$

$$\sum_{j\in P} o_i^{jt} = 1 \qquad \forall i \in O, t = \{p+1\} \tag{3}$$

$$\sum_{k\in P\cup\{p+1\}} o_i^{tk} = 1 \qquad \forall i \in O, t = 1 \tag{4}$$

$$o_i^{tk} = 0 \qquad \forall i \in O, t, k \in P\cup\{p+1\}, t >= k \tag{5}$$

$$\sum_{i\in N\text{-}\{j\}} a_{ij}^t = \sum_{k\in P\cup\{p+1\},k>t} o_j^{tk} \qquad \forall j \in O, t \in P \tag{6}$$

$$\sum_{j\in N\text{-}\{i\}} a_{ij}^t = \sum_{k\in P\cup\{p+1\},k>t} o_j^{tk} \qquad \forall i \in O, t \in P \tag{7}$$

$$\sum_{i\in N\text{-}\{j\}} a_{ij}^t = \sum_{h\in N\text{-}\{j\}} a_{jh}^t \qquad \forall j \in O, t \in P \tag{8}$$

$$\sum_{i\in N\text{-}\{j\}} f_{ij}^t \geq \sum_{h\in N\text{-}\{j\}} f_{jh}^t \qquad \forall j \in O, t \in P \tag{9}$$

$$\sum_{i\in N\text{-}\{j\}} f_{ij}^t = \sum_{h\in N\text{-}\{j\}} f_{jh}^t - CumD_j^{tk} o_j^{tk} \qquad \forall j \in O, t \in P \tag{10}$$

Description of Constraints

(0) Objective Function: Inventory holding and Vehicle routing.

(1) Inventory balance.

(2) Continuity of Replenishment Period.

(3) There must be exact one link between the first period to an exact period.

(4) There must be exact one link from an exact period to period p+1.

(5) There is no link between later periods to former periods.

(6)-(7) Each Outlet is exactly visited by a vehicle only equal to order variable

(8) Route Continuity.

(9) Sub-tours Elimination.

(10) Relationship between flow variable and replenishment quantity.

4.3 Solution approach

4.3.1 Solution concept

The Inventory Routing Problem (IRP) is usually solved by the decomposition technique, due to its complexity that includes transportation and inventory decision. These decisions are known as NP-Hard problem so the IRP can be classified as a NP-hard problem too. As a result, the IRP is decomposed into two sub problems: Vehicle Routing Problem (VRP) and Lot-Sizing Problem (LSP). These sub problems are commonly solved by heuristic and the solution of a sub problem will be input parameters of another sub problem. The concept of the decomposition and updating information between sub problems is due to the coordination of transportation decision and inventory decision is the key for cost reduction in logistic system. In many situations the total logistic cost can be reduced by trading off between the transportation and inventory cost.

The cost reduction can be achieved by find the balance of inventory cost and transportation cost. Under this concept, we will discuss about the optimal solution and the two obvious solutions of this system. In this system the optimal solution may be in between the solution using Lot for Lot as replenishment policy and the solution which replenishes with the sum of demand for each outlet at the first period to cover all demands in considered horizon. With the Lot for Lot policy the delivery route will occur at every period, while the other one the delivery route will be set at only the first period.

These solutions can be the optimal solution for some conditions. The solution which is obtained by using Lot for Lot policy can be the optimal solution in case that the ratio of holding cost and routing cost is very high. In this case even the holding inventory of the smallest amount of demand can lead to excessively increasing in inventory holding cost while the routing cost can be barely reduced. In the other hands, the solution which is obtained by making replenishment with all demands can be the optimal solution in case that the ratio of holding cost and routing

cost is very low. In this case even the maximum inventory level can lead to slightly increasing in inventory holding cost while the routing cost can be greatly reduced.

In other cases, the ratio of holding cost and routing cost may be in between the value of those two extreme solutions. In these cases, the optimal solution can be achieve by trading off the routing cost and holding cost and solution may be that the outlets which the ratio of holding cost and routing cost is high will be replenished in almost every period while the outlets which the ratio of holding cost and routing cost is low will be replenished in only a few periods. The challenge of finding solution is to figure out that which outlets will be considered to replenish for almost every period or a few periods. The relationship between routing cost, holding cost and total cost is shown in Figure 4.2.

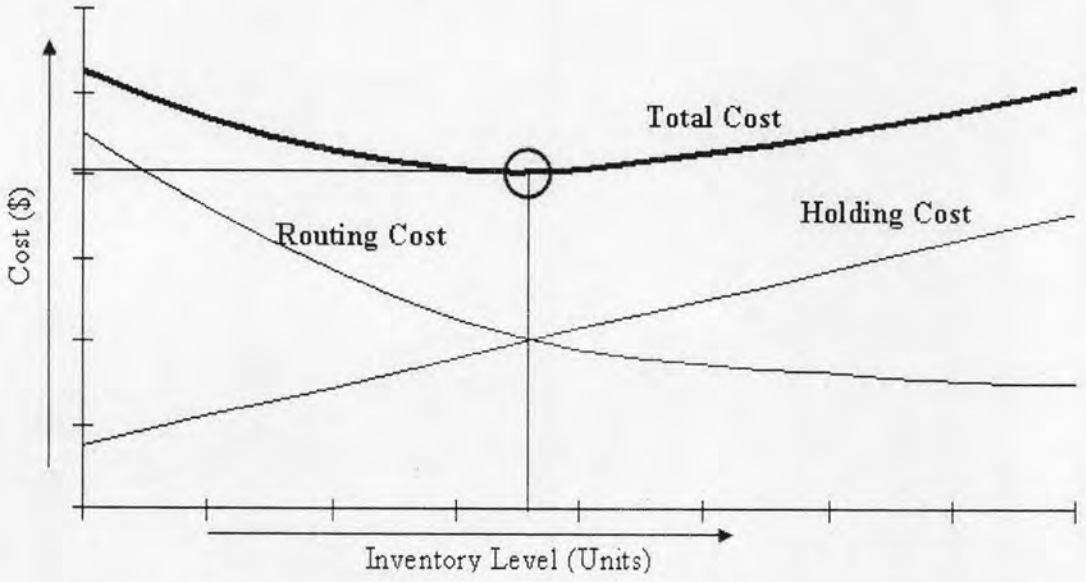


Figure 4.2 Relationship between routing cost, holding cost and total cost

As a result of study on optimal condition, a heuristic for solving the problem is developed under the concept that to consider the ratio of holding cost and routing cost of each outlet to determine the replenishment quantity and delivery route of them. This heuristic adapts an algorithm for Lot-Sizing problem (LSP) and algorithm for Vehicle routing problem (VRP) to solve the Inventory routing problem.

The basic knowledge to understand this heuristic is the knowledge about the sub problems after the main problem of Inventory routing problem is

decomposed. The sub problems are Lot-Sizing problem (LSP) and Vehicle routing problem (VRP).

Lot-Sizing problem (LSP) is the problem to find the replenishment quantity for a single outlet under the deterministic demand and given parameters. Parameters which involve in this problem are holding cost per period per unit and ordering cost per purchasing or production order. The optimal lot size is the solution that balances the ordering cost and holding cost under the planning horizon which can be infinite horizon. The solution of this problem has the condition that the dominant set of solution will not allow the demand to be spilt for replenishment in more than one period. So the problem is consider as a shortest path problem and can be efficiently solve by the algorithm that can solve shortest path problem.

Vehicle Routing Problem (VRP) is the problem to find the delivery routes for a group of location under the deterministic demand and given parameters. Parameters which involve in this problem usually are distance between location, number of vehicle and carrying capacity. The solution of this problem is a set of routes and assignment of each route to vehicles. Every route starts from a depot to visits all locations assigned to the route and return to the depot. Each of location must be exactly visit once. Each route is assigned to a vehicle to make delivery. This problem is known as an NP-Hard Problem that means the excessive computational effort is needed to obtain the optimal solution. Due to its complexity, heuristics are usually used to solve it.

Two basic problems described in previous paragraph are modified into a heuristic for the inventory routing problem. The lot-sizing problem is used to determine replenishment period and quantity of each outlet while the vehicle routing problem is used for providing the setup cost for the lot-sizing problem of the considered outlet. The main difference between the basic lot-sizing and the lot-sizing in this problem is the setup cost. The setup cost in the basic lot-sizing model is the constant and can be known in advance as a parameter of the model-sizing model used in this problem is the lot-sizing model with the dynamic setup cost. The setup cost in

this problem depends on the structure of route which is determined by the vehicle routing model. The vehicle routing problem will provide the setup cost by construct two routes which are the route including the considered outlet and excluding it and the setup cost for the lot-sizing problem of this outlet is the difference on route cost between the two routes. The illustration of use of lot-sizing problem and vehicle routing problem to solve the problem is shown in Figure 4.3.

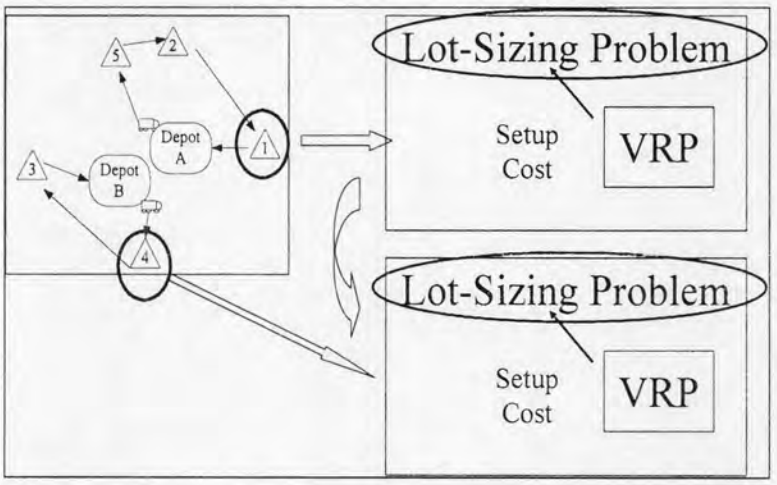


Figure 4.3 Use of lot-sizing problem and vehicle routing problem to solve the problem

4.3.2 Solution procedure

Briefly, the proposed heuristic consists of three main processes. Firstly, the algorithm starts with finding initial solution. To determine replenishment quantity of initial solution, the algorithm uses Lot-For-Lot policy, and then vehicle routing problems are solved by the cheapest-cost insertion and later improved by two-opt in each period. Secondly, the algorithm determines replenishment quantity and delivery period of an outlet by solving Lot-Sizing Problem of it. In this phase, the algorithm defines Arcs and Nodes to perform the shortest path algorithm for Lot-Sizing Problem. After the delivery period of all outlets is obtained, delivery routes are set up for each period. By solving the shortest path problem using updated information about routes and replenishment quantity from the solution of previous phase, total cost is iteratively reduced. Finally, a termination criterion is checked and decision variable and solving information are updated. Overall algorithm of this heuristic can be elaborately written into six steps as follows.

Step 1: Set up an initial phase

In this step the heuristic obtains an initial solution by using Lot-For-Lot as a replenishment policy. Thereby, the initial replenishment quantity is equivalent to customer demand and the initial routing is set by using cheapest insertion and improving by two-opt algorithm.

Step 2: select an outlet

The heuristic randomly select an outlet to determine delivery period and replenishment quantity of it in Step 3. The outlet can be selected with random sequence because the sequence of selecting an outlet takes no effect to the final solution. This can be explained through illustrations and proofs. Firstly, the proof will be presented for the case of a depot and two outlets which is the simplest case and easy to understand. The illustration of the example system is shown in Figure 4.4. In the Figure 4.4, a depot supply an item to two outlets name "1" and "2" via a vehicle. Distance between the depot and outlet "1" and "2" is "b" and "c" respectively and the distance between the two outlets is "a".

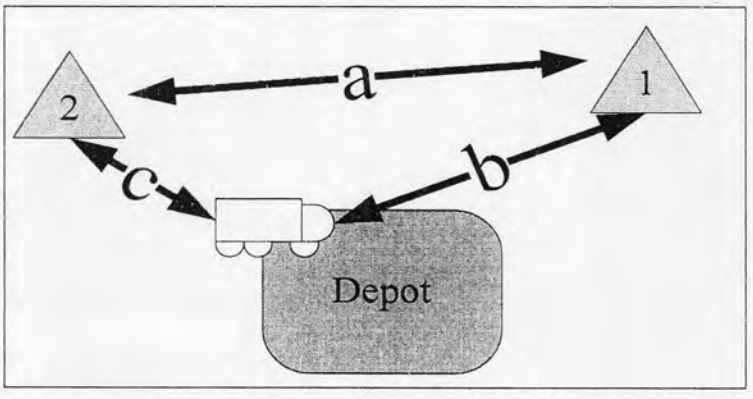


Figure 4.4 The example system for proof on sequence effect

The proof starts by assuming that the outlet "1" is considered first. The route cost that can be reduced as shown in Figure 4.5 is a+b-c. If the replenishment at period 1 to cover demand of outlet "1" in period 1 and 2 is the effective choice, the value of a+b-c which is the setup cost in lot-sizing problem of outlet "1" must greater than the increased inventory holding of outlet "1" in making

replenishment in period with the amount of demand at period 1 plus period 2. Assume the increased holding cost is Δh so the value a+b-c is greater than Δh. This can be concluded in equation (4.1).

$$a+b-c > \Delta h \tag{4.1}$$

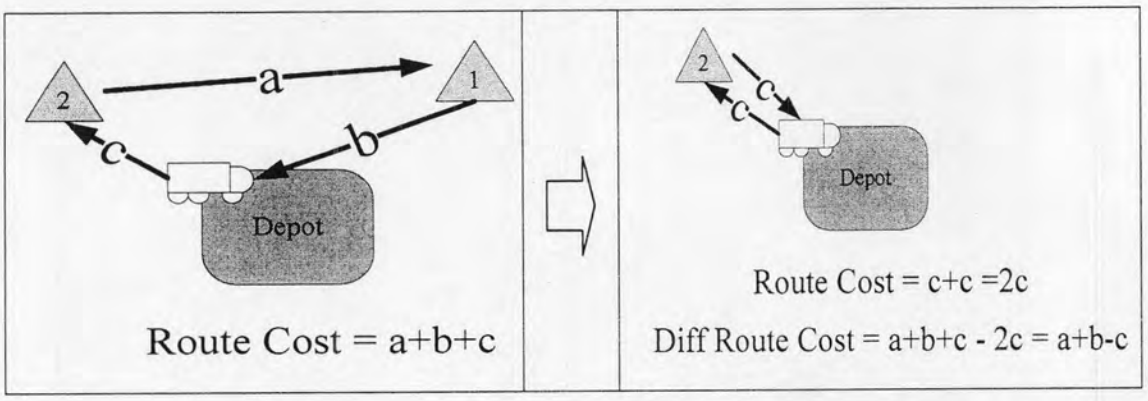


Figure 4.5 The example system for proof on sequence effect2

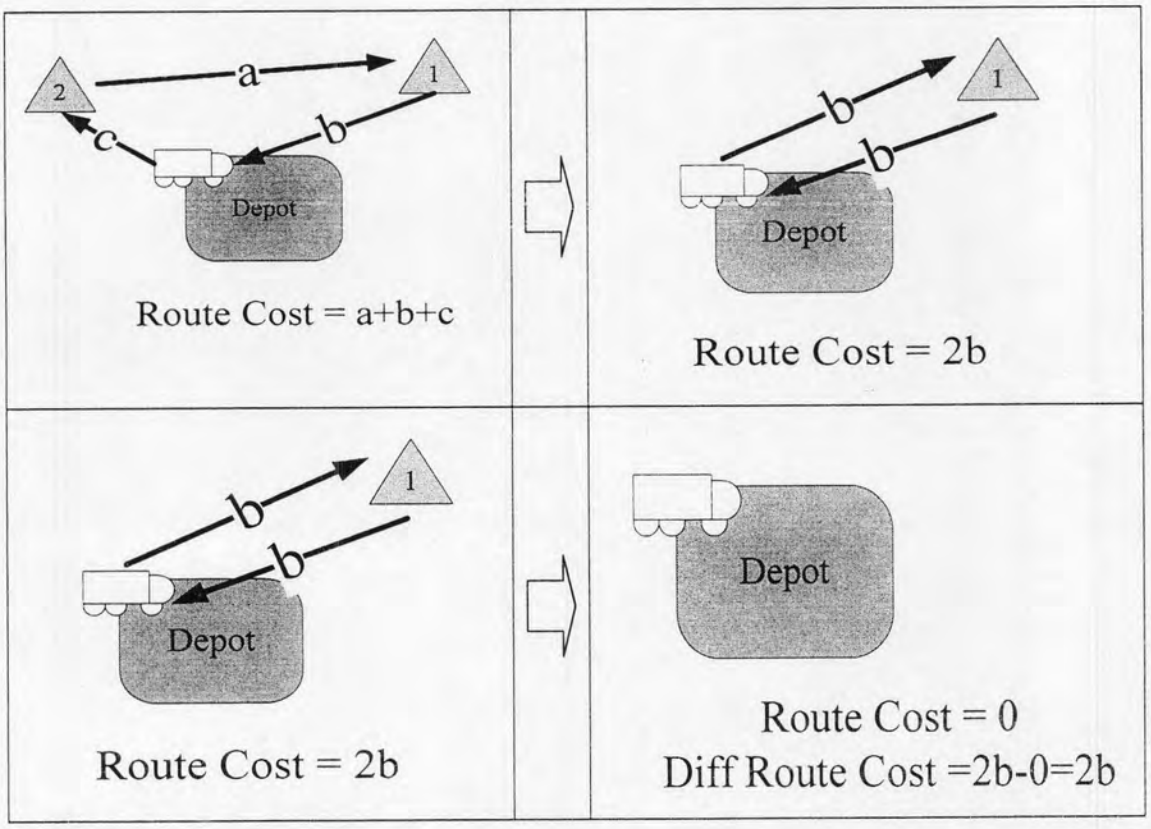


Figure 4.6 The example system for proof on sequence effect3

The sequence will take effect on the final solution if the replenishment on outlet "1" in the case that the outlet "2" is considered before outlet

"1"is not the same as the case that outlet "1" is considered before outlet "2". According to the Figure 4.6 the route cost that can be reduced is 2b. If the replenishment at period 1 to cover demand of outlet "1" in period 1 and 2 is the ineffective choice, the value of 2b which is the setup cost in lot-sizing problem of outlet "1" must less than the increased inventory holding of outlet "1" in making replenishment in period with the amount of demand at period 1 plus period 2. Assume the increased holding cost is Δh so the value 2b is less than Δh. This can be concluded in equation (4.2).

$$2b < \Delta h \tag{4.2}$$

By combining equation (4.1) and (4.2) the equation (4.3) and (4.4) can be derived.

$$2b<a+b-c \tag{4.3}$$

$$b+c<a \tag{4.4}$$

From the triangle inequality that the sum of length of any two sides of triangle must be longer than the length of remaining side. The route in this example has the shape similar to triangle so this property of triangle can be used for this example. From triangle inequality the relationship between a, b and c can be presented by equation (4.5)

$$b+c>a \tag{4.5}$$

Hence it is found that the equation (4.4) is wrong due to it contradicts to the triangle inequality as shown in equation. From the proof it can be concluded that the sequence takes no effect to the final solution.

Step 3: Determine delivery period using shortest path algorithm.

Step 3.1 Define Node and Arc.

This step begins with creating the shortest path networks for every outlet which each period is represented by a node and each

replenishment period is represented by an arc. To exemplify, an arc from k to t means considered outlet inventory level is replenished at period k and then period t. These networks are solved by an efficient shortest path algorithm in order to determine delivery period of each outlet. The illustration of shortest path network is presented in the Figure 4.7.

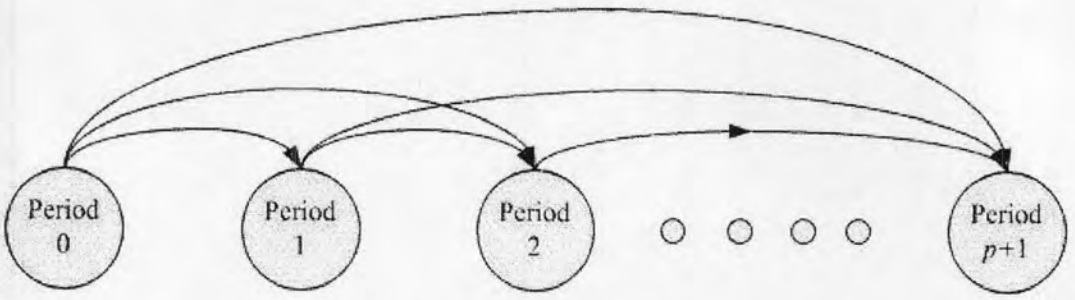


Figure 4.7 Illustration of shortest path network

For figure 4.7, each circle stands for node, period 0, 1, 2, ..., p+1 and each arrow line stands for arc, replenishment period 0-1, 0-2, 0-3, ..., 0-p+1, 1-2, ..., p-p+1. Hence, we will name node 0, node 1, node 2, ..., and node p+1, and arc 0-1, arc 0-2, arc 0-3, ..., and arc p-p+1.

Step 3.2 Define Arc cost.

Cost of every arc is the summation of increasing holding cost and the difference of route cost of selecting the considered arc. An increasing holding cost of arc k-t is determined by sum all inventory holding cost of the considered outlet from period t to period k+1. Route cost is determined by calculating the difference between route cost including considered this outlet and route cost without considered this outlet at period k.

Step 3.3 Solve by an efficient shortest path algorithm.

The well-known Dijkstra's algorithm (Dijkstra, 1959) is used to solve shortest path problem. The solution will be used to determine replenishment quantity in the next step.

Step 3.4 Determine decision variables using results from the shortest path algorithm.

Heuristic determines delivery period using solution from shortest path problem. The shortest path results in delivery period of considered outlet set. For example, delivery period of the considered outlet are period *k* and then period *t* if arc *k-t* is a part of shortest path solution.

Step 4: Update replenishment quantity and route

Heuristic determines replenishment quantity using solution of shortest path solution from previous step. The replenishment quantity is sum of demand from period *k* to period *t* of arc *k-t* that includes in shortest path solution. For delivery route, heuristic uses information of delivery period to determine route. For example, if there is delivery schedule of considered outlet at period 2, this outlet will be included into consideration with other outlet existing in this period. After that the route will be set up using cheapest insertion and two-opt improvement. There can be no updated because shortest path solution results in the same delivery schedule as the solution in previous step.

Step 5: Check whether all outlets are considered.

Heuristic will repeat the step 2 to 4 until all outlets are considered. After all outlets are considered, the heuristic will continue to the next step.

Step 6: Check terminating condition

Heuristic will check new total cost compare to the current best total cost. The new total cost will be accepted and heuristic will repeat the step 2 to 5, if the new one is lower than the current one. If the new cost is not less than the current cost, the heuristic is terminated.

4.3.3 Numerical example

In this section, a simple example is illustrated. This example consists of a depot and four outlets. Decisions are made for three periods. Figure 4.8 illustrates the graphical location of the depot and outlets in the example problem. Table 4.3 shows the distance between all locations. Demand of all outlets is given in Table 4.4. Table 4.5 gives the information about the holding cost of the outlets. Beginning inventory of all outlets are assumed to be zero.

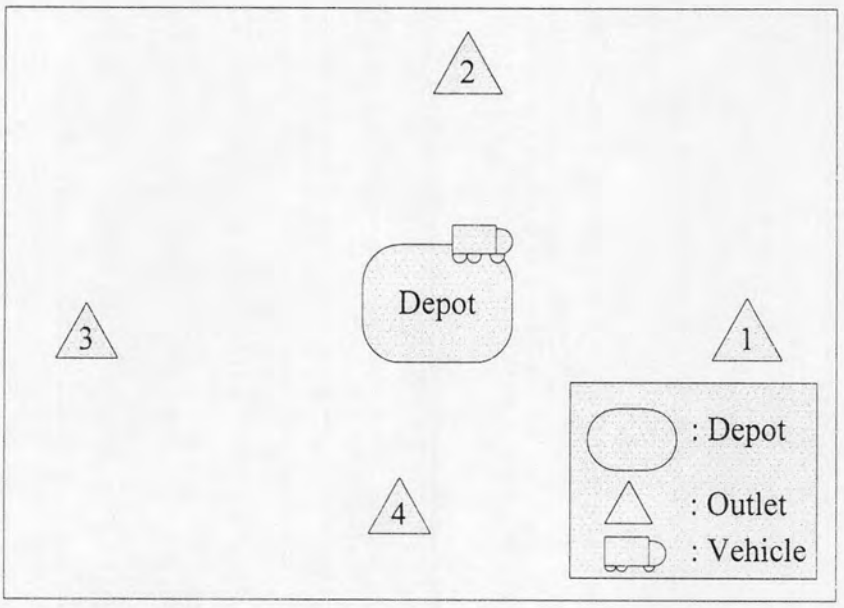


Figure 4.8 Illustration of the graphical location of the depot and outlets in the example problem.

Table 4.3 Distance between all locations

	Depot	Outlet 1	Outlet 2	Outlet 3	Outlet 4
Depot	0	7	8	8	9
Outlet 1	7	0	10	15	10
Outlet 2	8	10	0	13	17
Outlet 3	8	15	13	0	11
Outlet 4	9	10	17	11	0

Table 4.4 Demand of all outlets

Outlet \ Period	1	2	3
1	12	8	10
2	11	9	8
3	10	8	9
4	8	10	11

Table 4.5 Holding cost of all outlets

Outlet	Holding Cost
1	0.10
2	0.15
3	0.25
4	0.07

Step 1: Set up an initial phase

The initial solution is obtained by the Lot-for-Lot policy and shown in Table 4.6 and Figure 4.9. In Table 4.6, the replenishment quantity is presented. It can be indicated that the replenishment quantity is equivalent to customer demand. In Figure 4.9 shows the illustration of delivery route in each period. The route for all periods starts from depot indexed by "0" and goes to outlet "4", "1", "2", "3" and return to "0".

Table 4.6 Replenishment quantity and delivery route of initial solution

Outlet \ Period	**1**	**2**	**3**
1	12	8	10
2	11	9	8
3	10	8	9
4	8	10	11
Route	0-4-1-2-3-0	0-4-1-2-3-0	0-4-1-2-3-0

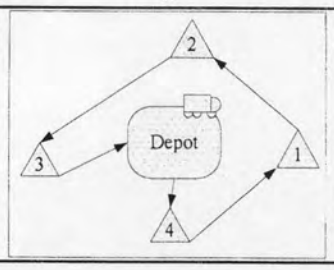
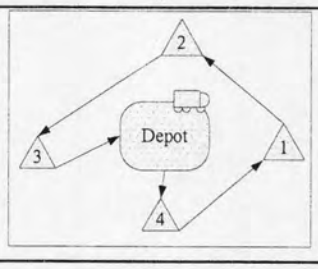
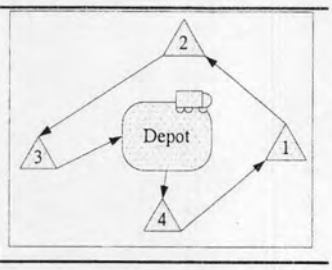
Period	1	2	3
Route			

Figure 4.9 Initial Route

Step 2: select an outlet

For the example problem the outlet "1" is randomly selected as shown in Figure 4.10.

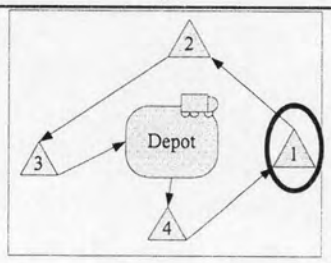
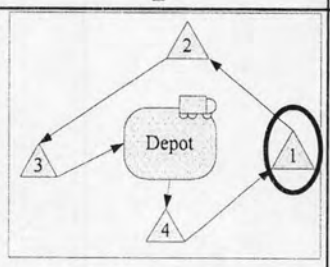
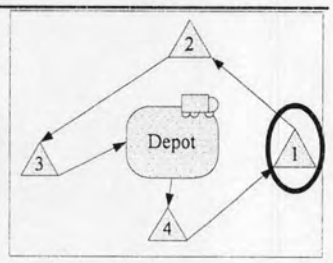
Period	1	2	3
Route			

Figure 4.10 Outlet named "1" is selected

Step 3: Determine delivery period using shortest path algorithm.

Step 3.1 Define Node and Arc.

The illustration of shortest path network of the outlet "1" in this iteration of heuristic procedure is presented in the Figure 4.11.

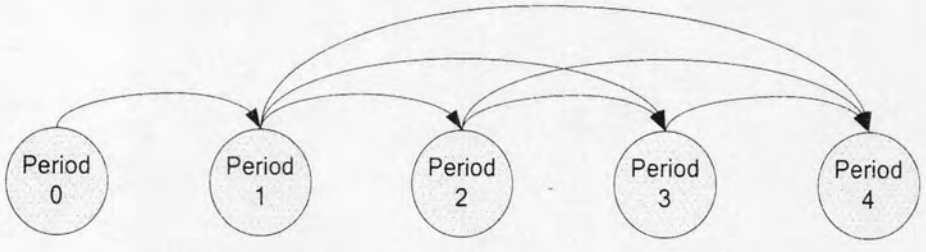


Figure 4.11 Illustration of shortest path network

For figure 4.11, each circle stands for node, period 0, 1, 2, ..., 4 and each arrow line stands for arc, replenishment period 0-1, 1-2,..., 3-4. Hence, we will name node 0, node 1, ..., and node 4, and arc 0-1, arc 1-2, arc 1-3, ..., and arc 3-4. The meaning of arc k-t is there is replenishment at period t after replenishment at period k. For example arc 2-3 means the outlet "1" is replenished at period 3 after replenishment at period 2. For the case of arc 0-t means the outlet "1" is first replenished at period t ex. an arc 0-1 means the first replenishment to outlet "1" is made at period 1. For the case of arc k-4, which node 4 is the last node in this problem, means the outlet "1" is last replenished at period k ex. an arc 2-4 means the last replenishment to outlet "1" is made at period 2 to cover demand of period 2 to period 3. It can be notice that there are no arc 0-2, 0-3 and 0-4 because these arcs violate the stock out constraint. For example arc 0-2 violates the stock out constraint since the first replenishment is made at period 2 which the demand at period 1 is not been satisfied.

Step 3.2 Define Arc cost.

Cost of every arc is the summation of increasing holding cost and the difference of route cost of selecting the considered arc. An increasing holding cost of arc k-t is determined by sum all inventory holding cost of the considered outlet from period t to period k+1. Rout cost is determined by calculating the difference between route cost including considered this outlet and route cost without considered this outlet at period k. The illustration of arc cost of shortest path network of the outlet "1" is shown in Figure 4.13.

Table 4.7 Holding Cost of Replenishment Period following Arc

To Period / From Period	1	2	3	4
0	0	-	-	-
1	-	0	8x0.1=0.8	(18+10)x0.1=2.8
2	-	-	0	10x0.1=1
3	-	-	-	0

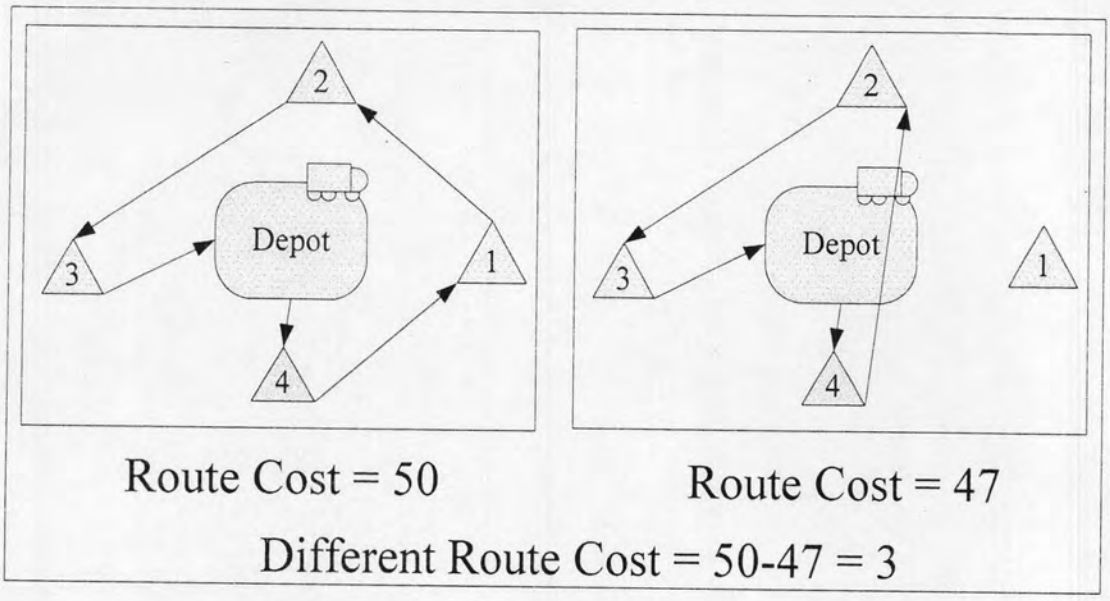


Figure 4.12 Illustration of different route cost in period 2

Table 4.8 Arc Cost of "1"

To Period / From Period	1	2	3	4
0	0	-	-	-
1	-	0+3=3	0.8+3=3.8	2.8+3=5.8
2	-	-	0+3=3	1+3=4
3	-	-	-	0+3=3

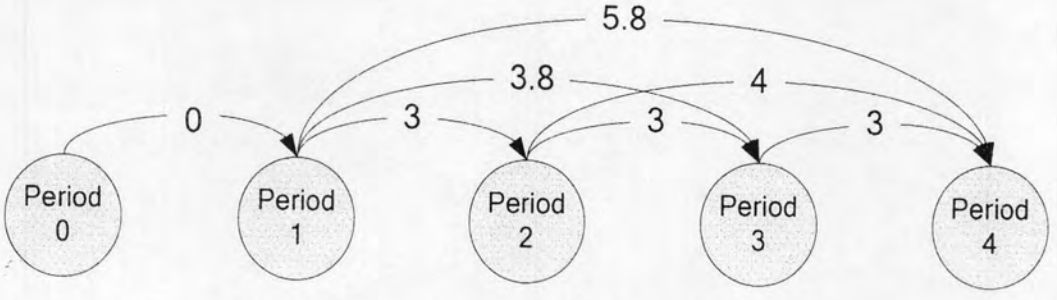


Figure 4.13 Illustration of arc cost of shortest path network

Step 3.3 Solve by an efficient shortest path algorithm.

From the example the result of solving shortest path is the path 0-1-4.

Step 3.4 Determine decision variables using results from the shortest path algorithm.

For example, delivery period of the considered outlet are period k and then period t if arc k-t is a part of shortest path solution. From the example the path is 0-1-4 so there is delivery schedule for "1" on period 1 with the replenishment quantity which covers demand of the outlet "1" from period 1 to period 3.

Step 4: Update replenishment quantity and route

The solution after updating is shown in Table 4.9 and Figure 4.14. In Table 4.9, the replenishment quantity is presented. It can be indicated that the replenishment quantity of the outlet "1" in period 1, 2 and 3 are changed. The value at period 1 is 30 which is derived from summation of demand from period 1 to period 3. The period 2 and 3 has no replenishment quantity since the replenishment quantity in period 1 can cover demand in this period already. In Figure 4.13 shows the illustration of delivery route in each period. The routes in period 2 and 3 are changed from previous state. The routes in period 2 and 3 become 0-4-2-3-0 that means the vehicle starts from depot indexed by "0" and goes to outlet "4", "2", "3" and return to "0".

Table 4.9 Replenishment quantity and delivery route after updating

Period / Outlet	1	2	3
1	12+8+10=30	0	0
2	11	9	10
3	10	8	9
4	8	10	11
5	9	11	10
Route	0-4-1-2-3-0	0-4-2-3-0	0-4-2-3-0

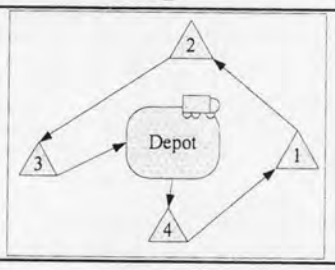
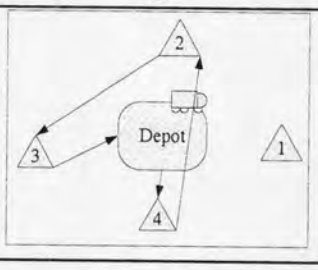
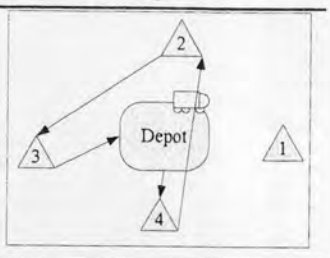
Period	1	2	3
Route			

Figure 4.14 Route after Updating

4.4 Computational experiments

4.4.1 Test problems

Seven test instances from standard library of Christofides and Eilon (1969), which contains problem size as shown in Table 4.10, are used for performance testing. The test instances from standard library of Christofides and Eilon (1969) provide demand and distance information of outlet and warehouse. However the inventory holding cost is not provided in the instances so it will be randomly generated using uniform distribution with range (0.1, 1). For each test instance are tested under the number of time horizon (H) 2,5,10 and 20 periods. Hence there are twenty eight test combinations for performance testing. Four problems are generated for each test combinations.

For the performance testing the solutions from the proposed heuristic are compared to solution from CPLEX and from Lot-For-Lot policy (LFL) which the replenishment quantity follows demand and delivery route is solved by the cheapest insertion and two-opt algorithm. The performance of heuristics will be tested in two aspects; solution quality, and computational time. The performance of the heuristic will be presented in form of the percentage gap from the solution of CPLEX, the improvement percentage from the solution of CPLEX and the improvement percentage from the Lot-For-Lot policy (LFL). The calculation of percentage gap from the optimal solution is shown in the following equation:

$$\% \text{ gap from the CPLEX solution (Gap\% vs. CPLEX)} = \frac{Sol_{heu} - Sol_{cplex}}{Sol_{cplex}} \times 100 \qquad (4.1)$$

where Sol_{heu} is the solution obtained by a given heuristic, and Sol_{cplex} is the solution obtaining by using CPLEX. The calculation of the improvement percentage from the solution of CPLEX is shown in the following equation:

$$\% \text{ improvement from the CPLEX solution} = \frac{Sol_{cplex} - Sol_{heu}}{Sol_{cplex}} \times 100 \qquad (4.2)$$

The calculation of the improvement percentage from the Lot-For-Lot policy is shown in the following equation:

$$\% \text{ improvement from the Lot-For-Lot} = \frac{Sol_{lfl} - Sol_{heu}}{Sol_{lfl}} \times 100 \qquad (4.3)$$

where Sol_{heu} is the solution obtained by a given heuristic, and Sol_{lfl} is the solution obtaining by using Lot-for-Lot policy.

The heuristic has been implemented in the Visual Basic 6 programming language on a PC with an Intel Pentium 4 1.8 GHz CPU and 256 MB of RAM. The solutions of CPLEX are solved by AMPL/CPLEX 8.0.0 solver with the initial solution option which the solution obtained by the Lot-For-Lot policy. However, the CPU time is limited to at most 3 hours.

Table 4.10 Problem code and size

Problem Code	Number of outlet
E-n7-k2	6
E-n13-k4	12
E-n22-k4	21
E-n31-k7	30
E-n51-k5	50
E-n76-k7	75
E-n101-k8	100

Table 4.11 Summary of computational results for the proposed heuristic: compare to optimal solution

Problem		Gap (%)	CPU Time (Sec.)	
Name	Period	vs. CPLEX	CPLEX	Heuristic
E-n7-k2	2	0.00%	0.02	0.01
	5	0.00%	0.04	0.04
	10	0.00%	1.34	0.20
E-n13-k4	2	0.00%	0.85	0.05
E-n22-k4	2	2.94%	377.43	0.20

Table 4.12 Summary of computational results for the proposed heuristic: compare to best known solution

Problem		Improvement (%)		CPU Time (Sec.)		
Name	Period	vs. Lot-for-Lot	vs. CPLEX	Lot-for-Lot	CPLEX	Heuristic
E-n7-k2	20	66.53%	27.94%	0.01	10,800.17	0.98
E-n13-k4	5	11.70%	4.18%	0.01	10,800.17	0.38
	10	16.13%	13.08%	0.02	10,800.23	1.32
	20	10.81%	10.81%	0.03	10,800.30	6.14
E-n22-k4	5	28.81%	15.51%	0.04	10,800.10	2.70
	10	31.81%	29.01%	0.18	10,800.20	10.07
	20	30.54%	30.54%	0.21	10,800.50	45.67
E-n31-k7	2	10.87%	10.87%	0.04	10,800.10	0.70
	5	15.21%	15.21%	0.11	10,800.20	8.54
	10	13.12%	13.12%	0.21	10,800.43	32.55
	20	13.91%	13.91%	0.41	10,801.20	124.83
E-n51-k5	2	13.20%	13.20%	0.28	10,800.20	4.19
	5	15.79%	15.79%	0.71	10,800.62	45.61
	10	15.99%	15.99%	1.42	10,801.32	155.65
	20	15.27%	15.27%	2.76	10,806.32	702.61
E-n76-k7	2	20.31%	20.31%	0.74	10,800.60	15.03
	5	21.99%	21.99%	1.85	10,802.75	180.63
	10	22.28%	22.28%	3.70	10,804.62	754.32
	20	23.51%	23.51%	7.38	10,805.80	2,735.15
E-n101-k8	2	20.70%	20.70%	1.61	10,800.98	30.22
	5	23.94%	23.94%	3.98	10,802.28	297.55
	10	26.31%	26.31%	7.77	10,805.18	1,198.02

4.4.2 Result

The results of experiment for the proposed heuristic are given in Table 4.11 and 4.12. The results from experiment indicates that the proposed algorithm can

improve cost 8.05%-71.47% on average from Lot-for-Lot policy for problems which its optimal solution is found, and can improve 10.81%-66.53% for problems which optimal solution cannot be found. The proposed heuristic also yield an average of 0%-2.94% GAP compared to optimal solution from CPLEX 8.0, and result in 4.18%-30.54% better than the best known solution from CPLEX. For computational time aspect, the proposed heuristic can determine its solution much faster than CPLEX 8.0 solver.

4.4.3 Discussion

The experiments show that the proposed heuristic can perform with a good result compare to Lot-for-Lot and CPLEX. This may be due to the concept of cost reduction of routing cost from Lot-for-Lot policy while inventory holding is increasing with fewer amounts compared to reduction on routing cost. It can be noticed from the table of results that the CPLEX can not improve the solution from the initial solution obtained by the Lot-for-Lot policy. It could be due to the complexity of the problem. However the CPLEX can improve the lower bound and percentage of gap between upper bound and lower bound continuously during the solving process. Hence if the time limit of CPLEX is expanded, the CPLEX may achieve the better solution than Lot-for-Lot policy.

4.5 Conclusion

This chapter studies a simplified version of the problem described in chapter III. This chapter begins with the problem description section and a mathematical model that represents the problem is presented in section 4.2. However, due to the complexity of the problem the exact solution technique is not suitable for medium to large-sized problem. A heuristic algorithm which is developed to solve this problem within reasonable time is described in section 4.3. The performance of the proposed heuristic is test in section 4.4. The computational results show that the heuristic can perform well in various problem sizes within the reasonable time. In the next chapter this heuristic will be modified to solve the multi-item multi-depot inventory routing problem.