Journal of Engineering Science and Technology
Vol. 14, No. 5 (2019) 2703 - 2715

THRESHOLD BASED ALGORITHMS FOR THE MULTI-PRODUCT MULTI-PERIOD INVENTORY ROUTING PROBLEM

FADILLAH RAMADHAN[1], ARIF IMRAN[1,*], AFRIN F. RIZANA[2]

[1]Department of Industrial Engineering, Institut Teknologi Nasional, Bandung, Indonesia
[2]Department of Industrial Engineering, Telkom University, Bandung, Indonesia
*Corresponding Author: arifimr@yahoo.com

Abstract

Inventory management and transportation are important factors to enhance the organization competitive advantage in the area of supply chain management. The integration of these aspects is known as the Inventory Routing Problem (IRP). In this paper, we address the multi-product, multi-period IRP. Here, a depot houses homogenous vehicles that deliver multi-period of times to an assembly plant, from multi-supplier. The aim is to find the least total inventory and transportation cost. We develop Threshold Acceptance (TA) and Record to Record Travel (RTR) based algorithm to solve the IRP. The initial solution for both algorithms is obtained by using an adaptation of the least cost insertion procedure called the least cost insertion with vehicle capacity maximization (LCI-VCM). A number of local searches and LCI-VCM are applied when conducting downhill moves to search the lowest total cost. The proposed algorithms are then tested to solve the dataset from the literature and competitive results are obtained.

Keywords: IRP, Record-to-record, SCM, Threshold-acceptance, Transportation.

1. Introduction

Cost-effectiveness is essential for a company to be more profitable. Furthermore, a company must continually increase its competitive advantage due to the competition among companies. It has been proven that a company's operating cost can be minimized by conducting an efficient implementation of supply chain management [1, 2]. Thus, achieving product availability, shipping consistency, accuracy in inventory, and increasing effectiveness of the distribution process with the Inventory Routing Problem (IRP) is crucial. The term IRP is the integration of inventory management and transportation.

The characteristics of the IRP model can vary, in terms of types of vehicle that are homogeneous or heterogeneous, transferring single or multiple types of product over a given planning horizon without stocking out at a set of *n* customers [3]. In a number of previous studies by Abdelmaguid et al. [4], Coelho et al. [5] and Hiassat et al. [6], this has been addressed. Moin et al. [1] investigated an IRP model that consider capacitated homogeneous vehicles (vehicles housed at a depot) in a finite horizon (transport products from supplier to assembly plant) used for multi-period, multi-suppliers, and products. Moin et al. [1] developed a hybrid genetic algorithm that used a new set of crossovers, mutation operators, and chromosome representations to tackle the IRP. Mjirda et al. [2] developed a two-phase variable neighbourhood search (VNS) metaheuristic to solve a similar IRP model. In the first phase, VNS is used to address the vehicle routing problem. In the second phase, the variable neighbourhood descent (VND) is applied to iteratively improve the initial solution to minimize transportation and inventory costs. Their algorithm produced better results than the one from Moin et al. [1].

Dror et al. [7] put forth a certain computational comparison of algorithms for the IRP, while Agra et al. [8] created a hybrid heuristic approach, which was more promising than the classical algorithm. However, there was a penalty cost resulting from the creation of backlog, taking into account the volatility of multiple clients towards a single producer with a stochastic single item of IRP. Peres et al. [9] suggested an alternative IRP model, which sought the best outcome through exact formulation optimization, in which, there is a transshipment movement of vehicles, supplying more than one product to customers from factories over a multi-period. This is done through the extensive use of computer simulations method, on a real large-scale retail industry, thereby realising reductions for cost. Moin et al. [10] proposed an artificial bee colony and scatter search algorithm to solve multi-product IRP with time-varying demand. Wong and Moin [11] developed ant colony optimization to tackle the split delivery IRP. Xiao and Rao [12] proposed a fuzzy genetic algorithm for multi-product multi-period inventory routing constraints with time window constraints. Al-e-hashem and Rekik [13] developed mixed-integer linear programming for the IRP model that consider the greenhouse gas emission level with CPLEX to solve the model. Noor and Shuib [14] created the data set for the multi-depot IRP problem using clustering techniques.

Planning horizon, number of periods, number of products, deterministic or stochastic demand, homogeneous or heterogeneous capacitated vehicles, are different criteria in an IRP model [15-17]. Metaheuristic, hybrid heuristic or exact formulation optimization are various algorithms executable for the IRP model. Although it takes more time to compute than the rest, the results from exact formulation optimization are the most reliable. Hence, heuristic, metaheuristic or hybrid heuristic approach is usually used where there is a time constraint, albeit with bad quality results due to the poor local minimum of the algorithm that is trapped in.

Threshold Acceptance (TA) algorithm can be used as an alternative approach to avoid becoming trapped at a poor local minimum, and it can get a better solution [18]. Tarantilis et al. [18] used threshold acceptance for solving the heterogeneous fleet vehicle routing problem. TA can make the uphill moves based on the upper bound of the objective function [19]. By accepting a worse solution than the current solution with uphill moves, a better solution can be generated by adding downhill moves. TA used a variable number of the upper bound of the objective function for the uphill movement of the solution. Li et al. [20] proposed a variant of the threshold acceptance algorithm in a Vehicle Routing Problem (VRP) type. They used a Record-to-Record Travel (RTR) based algorithm for solving the heterogeneous fleet vehicle routing problem. RTR used a fixed number of the upper bound of the objective function so that the uphill moves can be done based on that number of upper bound.

To our knowledge, TA and RTR have never been used to solve the multi-depot multi-product IRP model. With the difference in characteristics between VRP and IRP, especially in terms of the inventory costs that must be considered, it is necessary to develop the TA and RTR algorithms. In this paper, Moin et al. [1] and Mjirda et al. [2] solved the IRP model similar to the one we solved.

There are five sections in this document. After the introductory section, which makes the first section, the IRP's mathematical formulation was stated in the second section. While the TA and RTR to be used shall be expounded upon in Section 3. In Section 4, findings from the computation of the metaheuristic methods put forth shall be declared, and then a conclusion shall make up the fifth and final section.

2. Mathematical Formulation

The IRP shall be dealt with in an identical manner to Moin et al. [1] and Mjirda et al. [2]. For this research, one depot, one assembly plant, and *n* suppliers make up a distribution network.

In addition, there are a number of homogeneous vehicles stationed at the depot that deliver products from multiple suppliers to the assembly plant and report to the depot to end the trip. This IRP calls for an unlimited number of vehicles, as dictated by the need of the assembly plant and the finite horizon, multi-period, multi-product parameters of the model, which are underscored by the forbidding of back-ordering or backlogging.

The cost of each trip is fixed based on the distance covered on the trip. Although, there can be additional supply exceeding the amount ordered than can also incur an inventory holding. Moin et al. [1] prescribed the formulation to be used in this study.

Objective function

$$Z = min \;\; V\left(\sum_{\substack{j\in S\\ j\neq i}}\sum_{i\in S\cup D} r_{ij}\left(\sum_{t\in\tau} x_{ijt}\right) + \sum_{i\in S} r_{i,n+1}\left(\sum_{t\in\tau} x_{i,n+1,t}\right)\right)$$

$$+ r_{n+1,0}\sum_{t\in\tau}\sum_{i\in S} x_{0it} \;\; + F \;\sum_{t\in\tau}\sum_{i\in S} x_{0it} + \sum_{i\in S} h_i\left(\sum_{t\in\tau} inv_{it}\right) \tag{1}$$

Subject to

$$inv_{it} = inv_{i,t-1} + a_{it} - d_{it}, \;\; \forall i \in S, \forall t \in \tau \tag{2}$$

$$\sum_{\substack{i\in S\cup D\\ i\neq j}} q_{ijt} + a_{jt} = \sum_{\substack{i\in S\cup A\\ i\neq j}} q_{jit}, \quad \forall j \in S, \forall t \in \tau \tag{3}$$

$$\sum_{i\in S} q_{i,n+1,t} = \sum_{i\in S} a_{it}, \quad \forall t \in \tau \tag{4}$$

$$\sum_{\substack{i\in S\cup D\\ i\neq j}} x_{ijt} = \sum_{\substack{i\in S\cup A\\ i\neq j}} x_{jit}, \quad \forall j \in S, \forall t \in \tau \tag{5}$$

$$\sum_{j\in S} x_{ijt} = \sum_{j\in S} x_{jkt}, \quad i \in D, k \in A, \forall t \in \tau \tag{6}$$

$$q_{ijt} \leq C, \quad \forall i \in S, \forall j \in S \cup A, i \neq j, \forall t \in \tau \tag{7}$$

$$inv_{it} \geq 0, \quad \forall i \in S, \forall t \in \tau \tag{8}$$

$$q_{it} \geq 0, \quad \forall i \in S, \forall t \in \tau \tag{9}$$

$$x_{ijt} \in \{0,1\}, \quad \forall i, j \in S, \forall t \in \tau \tag{10}$$

$$x_{0jt} \geq 0 \; and \; integer, \quad \forall j \in S, \forall t \in \tau \tag{11}$$

$$x_{i,n+1,t} \geq 0 \; and \; integer, \quad \forall i \in S, \forall t \in \tau \tag{12}$$

$$x_{ijt} = 0, \quad i \in D, j \cup A, \forall t \in \tau \tag{13}$$

$$x_{ijt} = 0, \quad i \in S, j \cup D, \forall t \in \tau \tag{14}$$

$$x_{ijt} = 0, \quad i \in A, j \cup S, \forall t \in \tau \tag{15}$$

$$q_{ijt} \geq 0, \quad \forall i \in S, \forall j \in S \cup A, \forall t \in \tau \tag{16}$$

$$q_{0it} = 0, \quad \forall i \in S, \forall t \in \tau \tag{17}$$

The mathematical formulation breakdown is Eq. (1)**.** The equation of objective function that consists of total travel cost based on travel distance, total fixed cost for all trips, and total inventory holding cost for all periods; Eq. (2) The equation of balancing total inventory level at the assembly plant for each supplier's product; Eq. (3) The equation of a balanced flow of product quantity transported; Eq. (4) The accumulative amount of picked-up quantities, from supplier to the assembly plant; Eqs. (5) and (6). Constraint equation for balancing a number of departure and vehicle arrival; Eq. (7) a maximum number of vehicle capacity for each trip in all periods; Eqs. (8) to (17) the remaining constraint equations for a non-negativity variable.

There are four main steps of the LCI-VCM algorithm. Initialization step is used to get an efficient route. Set the first node is depot and calculate the least travel distance of supplier $r_{ij,} \forall i \in D, \forall j \in S, i \neq j$.

Set the second node is the supplier with the least travel distance from the depot. Repeat the calculation until the accumulation demand of node candidate is bigger than the vehicle capacity. Exclude the last node that makes the total accumulation of the product pick-up bigger than capacity and generates the new route for the remaining node for all period. After all of the routes are determined, calculate the residual capacity for

each route and period. Repeat the addition number of product pick-up based on remaining capacity with a generated uniform distribution of random number. If the number of repeats reaches the maximum number of repeats, update the current route and total amount of product pick-up.

3. Threshold Acceptance and Record-to-record Travel for IRP

The search starts by generating the initial solution by using an adaptation of the least cost insertion called the least cost insertion with vehicle capacity maximization (LCI-VCM).

In the second stage, the TA and RTR based algorithms are applied to improve the initial solution. The least-cost insertion approach used in this study is different from previous studies, as this algorithm considers the remaining capacity of the vehicle, which will affect the cost of transportation and inventory.

3.1. Least cost insertion with vehicle capacity maximization

Li et al. [20] and Gendreau et al. [21] used the least cost insertion to determine the efficient route. Based on studies by Gendreau et al. [21], the least cost insertion algorithm here is a modification from the one applied to obtain the initial solution of RTR. The result obtained is an efficient route for the distribution process in multi-period times.

As this approach just focused on minimizing travel distance, some enhancement approach to integrate the least total cost of travel distance and inventory must be done. Sometimes the route generated from least cost insertion allows the vehicle to have a non-full capacity slot.

The larger capacity of a vehicle during a period, the less possibility to pick-up product from a particular supplier in the next period because demand has been fulfilled in the previous period. This condition can reduce the total distribution cost of the vehicle.

Because of the importance of maximizing vehicle capacity, we proposed some enhancement of least cost insertion named Least Cost Insertion with Vehicle Capacity Maximization (LCI-VCM). The LCI-VCM algorithm is shown in Fig. 1.

There are four main steps of the LCI-VCM algorithm. Initialization step is used to get an efficient route. Set the first node is depot and calculate the least travel distance of supplier $r_{ij,}\ \forall i \in D, \forall j \in S, i \neq j$.

Set the second node is the supplier with the least travel distance from the depot. Repeat the calculation until the accumulation demand of node candidate is bigger than the vehicle capacity. Exclude the last node that makes the total accumulation of the product pick-up bigger than capacity and generates the new route for the remaining node for all period.

After all of the routes are determined, calculate the residual capacity for each route and period. Repeat the addition number of product pick-up based on remaining capacity with a generated uniform distribution of random number. If the number of repeats reaches the maximum number of repeats, update the current route and total amount of product pick-up.

Step (0) Initialization. Get an efficient route solution based on the basic least cost insertion algorithm

Step (1) Check. Calculate the residual capacity (RC) of the vehicle at the end of period $RC_t = C - a_{it}$, $\forall i \in S, \forall t \in \tau$, set number of repeats $NR_{max,t} = RC_t$, $\forall t \in \tau$.

Step (2) Repeat. Do the following steps until $NR = NR_{max}$

a) **Random number.** Generate a random number (an integer value of uniform distribution) $\mathrm{RN}[0, \mathrm{RC_t}]$.

b) **Choose.** From a number of suppliers on the route, choose one supplier randomly.

c) **Accumulate.** Calculate the accumulation number of pick-up from the chosen supplier $\mathrm{a_{it}} = \mathrm{a_{it}} +$ RN (random number).

d) **Increment.** Set NR = NR + 1.

Step (3) Finish. Update the current route and total amount of product pick-up for each supplier.

Fig. 1. LCI-VCM algorithm.

3.2. Threshold acceptance (TA) algorithm

Dueck and Scheuer [22] proposed the TA, which is a variant of the simulated annealing approach. TA accepts every new solution configuration, which is worse than the current solution. The basic TA algorithm of Dueck and Scheuer [22] is shown in Fig. 2.

Step (0) Initialization. Choose an initial configuration and threshold $T > 0$.

Step (1) Opt. Choose a new configuration with a local search method.

Step (2) Compute. Compute ΔE = quality of new configuration - quality of old configuration, check if $\Delta E > -T$, then old configuration = new configuration.

Step (3) Lower threshold. Check if no increase in quality in many iterations then lower the threshold and back to Opt.

Step (4) Stop. Check if no change in quality anymore or the threshold is too small then stop.

Fig. 2. Basic step of TA algorithm.

A rule is applied to reduce and increase the threshold or number of acceptances in the TA algorithm as put forth. A better solution is attained by the integration of local searches. The proposed TA algorithm can be seen in Fig. 3.

Parameter values for *repeat* TA_{max}, *iter*$_{max}$, *percent* TA_{start}, *percent* $TA_{increment}$, NA_{min}, and $NA_{increment}$ in the initialization section are obtained based on experiments. For the TA updating scheme, *percent* TA_{start} = 0.05, which means 5% higher than the total cost generated from the solution per iteration can be accepted and will decrease in each iteration by 0.5%. This causes the acceptance limit will continue to reduce by ten times up to the acceptance deviation limit of 0%. For the repeat step, several local searches are applied. The explanation of local searches is as follows: (1) the one-point-move inter-route. Here, we choose a node from a particular period and route randomly, and then move the node to another route in the same period. The position of node displacement is done by thorough experiment. The move is done in every feasible position and the node position will be chosen, which resulted from the smallest travel distance; (2) Two-point-move, in this case, two adjacent nodes are chosen and move to every feasible position in another route; (3) Random-two-point-move is similar to the two-point-move, the difference is choosing the two-chosen node. The two-node position is randomly selected from one route (not adjacent) and move to another route similar to one-point-move; (4) Swap-route (1-1), this procedure starts with one node chosen randomly identical with one-point-move and tries to swap it systematically with another

node in another route; (5) Swap-route (2-2) is similar to the swap-route (1-1). Two random nodes are chosen from a particular route and swap it systematically with two other nodes in another route; (6) Imran et al. [23] used a similar approach in 2-opt intra and inter route. 2-opt-intra route is used to avoid a cross route in particular route, while the 2-opt-inter route is applied to prevent cross route among the entire route.

All the basic procedures of local search in this multi-period IRP model are the same where the first procedure is to generate the random number of period and then do the local search in that period. If the new configuration is feasible, then check with the threshold number as in Fig. 3. The route solution in the previous period is still the same, however, in the selected period, this algorithm uses additional vehicle capacity maximization to get a better solution. As the initial inventory in the next period is different from the previous solution, the next period until a maximum number of periods uses the LCI-VCM algorithm and gets overall results. In the TA algorithm, LCI-VCM is not only used in the initial solution, however, also used when conducting downhill moves in the local search (one-point-move, two-point-move, etc.) Specifically, in the implementation section of the local search algorithm, the integration process of LCI-VCM and local search in the proposed TA algorithm can be seen in Fig. 4.

Step (0) **Initialization**. Generate an initial solution (LCI-VCM) x, set $x_{best} = x$, and define $repeatTA_{max} = 2$, $iter_{max} = 500$, $percentTA_{start} = 0.05$, $percentTA_{increment} = 0.005$, $NA_{min} = 10$, $NA_{increment} = 1$, $percentTA = percentTA_{start}$, $x_{TA} = x_{best}$, $repeatTA = 1$, $iter = 0$.

Step (1) **Repeat.** Do the following step until $iter = iter_{max}$.

a) For each local search at random (one-point-move, two-point-move, random-two-point-move swap-route(1-1), swap-route(2-2), 2-opt-intra, 2-opt-inter), set $c_{deviation} = c(x_{TA}) \times percentTA$, $TA = c(x_{TA}) + c_{deviation}$, set $NA = 0$ and generate feasible solutions x', check if $c(x') - c(x) \leq TA$ then set $NA = NA + 1$ and $x = x'$, check if $c(x) < c(x_{best})$ then set $x_{best} = x$.

b) Increment of iteration, $iter = iter + 1$.

Step (2) **Update TA**.

if $NA \geq NA_{min}$, reduce threshold as follows:

$percent\ TA = percentTA - percentTA_{increment}$, $NA_{min} = NA_{min} - NA_{increment}$, check if $percent\ TA = 0$ then $repeatTA = repeatTA + 1$, check if $repeatTA > repeatTA_{max}$ then go to Step (3), else set $x_{TA} = x_{best}$, $NA = 0$, $iter = 0$, back to Step (1).

else, increase threshold as follows:

$percentTA = percentTA + percentTA_{increment}$,

check if $percentTA > percentTA_{start}$,

then $percentTA = 0.05$, $NA_{min} = NA_{min} + NA_{increment}$, If $NA > NA_{min}$, then $NA = 10$.

Step (3) **Report** *the best solution obtained,* x_{best}.

Fig. 3. Proposed of TA algorithm.

Do the following step until $iter = iter_{max}$:

(1.1) Generate a random number to choose in which, period to be repaired.

(1.2) After selecting the period to be repaired, do the local search using one-point-move, two-point-move, random-two-point-move, swap-route(1-1), swap-route(2-2), 2-opt-intra, 2-opt-inter in that period.

(1.3) Update the initial inventory and demand for the next period.

(1.4) Use the LCI-VCM algorithm in the next period until the last period (this route determination considers transportation costs and inventory costs).

Increment of iteration, $iter = iter + 1$.

Fig. 4. Integration process of LCI-VCM and local search algorithm.

3.3. Record-to-record travel (RTR) algorithm

To avoid being trapped at a poor local minimum, RTR algorithm makes uphill moves. RTR is a deterministic variant of simulated annealing that is similar to TA [20, 23]. Uphill moves in RTR are conducted based on a fixed number of an upper bound of the objective function. The procedure of RTR algorithm is as follows: x as the current solution with a record and x' as an alternative solution. The record deviation was set at y% as a fixed percentage number of records. According to Dueck [24], the record deviation is 1%. If the result of x' is less than record + record deviation, then x' is the new solution. RTR algorithm as proposed by Li et al. [20] is adjusted to solve the IRP. The proposed RTR algorithm can be seen in Fig. 5.

Based on Fig. 5, M represents the maximum number of iterations allowed when the global record has not been updated, and K is similar with M, however, that is only when the maximum number of iterations in the record has not been updated. I represent the maximum number of RTR iterations with uphill moves. Li et al. [20] proposed a similar value as the determination of M, K, I's. Finally, LCI-VCM is used to derive the initial solution whilst maintaining the initial search in the TA algorithm. Similar to the TA algorithm, in RTR algorithm, LCI-VCM is also used with the local search to improve the route in the next period up to the last period as described in Fig. 4.

Step (0) **Initialization.** $M = 30,\ K = 20, I = 60, global\ record = \infty, global\ deviation = 1\%\ x\ global\ record$, generate an initial solution using LCI-VCM, set $record = the\ objective\ function, deviation = 1\%\ x\ record, iteration = 0$.

Step (1) **Repeat.** Do the following step until $iteration = M$, *go to Step (5) if iteration > M.*

Repeat nested loop 1. Do the following step until count = *K*.

Repeat nested Loop 2. Do the following step until i = *I*.

Do the local search one-point-move, two-point-move, random-two-point-move, swap-route (1-1), swap-route (2-2), 2-opt-intra, 2-opt-inter if $solution\ status = no\ feasible$, then go to Step (3).

if $solution\ status = feasible$, then check if $current\ solution < record + deviation, then\ set\ record = current\ solution, deviation = 1\%\ x\ record, count = 0$.

Step (3) **Downhill moves.**

Set $count = count + 1$, and do the local search one-point-move, two-point-move, random-two-point-move, swap-route (1-1), swap-route (2-2), 2-opt-intra, 2-opt-inter.

if $solution\ status = feasible$, then check if $current\ solution < record + deviation, then\ set\ record = current\ solution, deviation = 1\%\ x\ record, count = 0$, go back to Step (1) *repeat nested loop 1 if count < K.*

Step (4) **Check.** $iteration = iteration + 1$, and check if $record < global\ record + global\ deviation, global\ record = record, global\ deviation = 1\%\ x\ global\ record, iteration = 0$, go back to Step (1) *repeat if iteration < M.*

Step (5) **Finish.** Output solution of global record.

Fig. 5. Modification of RTR algorithm.

4. Results and Discussions

A 4 GB RAM with *i*3 processor computer is used to run the Visual Studio 2017 coded algorithm. Moin et al. [1] and Mjirda et al. [2] used the same dataset in this study. There are 14 data sets; S12T5 (12 suppliers, five periods), S12T10, S12T14, S20T5, S20T10,

S20T14, S20T21, S50T5, S50T10, S50T14, S50T21, S12T5, S12T10, and S12T14. The value of *C*, *V*, *F*, the *x* and *y*, which are coordinate of the depot, supplier, assembly plant, and inventory holding costs for each supplier are contained in each data set, as well as the demand for multi-period and the planning horizon. The parameter values of this IRP model are similar for each type of supplier data sets, and the value can be seen in Table 1.

Table 1. Parameter value of IRP model.

Datasets	S12T14	S20T21	S50T21	S98T14
C	20	20	20	200
V	1	1	1	50
F	10	10	10	400
Range of demand	[3, 27]	[3, 27]	[1, 9]	[1, 44]
Range of holding costs	[1, 4]	[1, 4]	[0, 9]	[0.04, 393.33]
Depot coordinate	[0, 0]	[0, 0]	[0, 0]	[0, 0]
Assembly plant coordinate	[10, 20]	[10, 20]	[10, 20]	[42.31, 83.17]

The result obtained by Moin et al. [1] and Mjirda et al. [2] are compared with the result produced by the proposed TA and RTR. The comparisons are presented in Tables 2 and 3.

It can be seen from Table 2 that TA produces eight results better than GA and one result better than VND. RTR produces 11 results better than GA, three results better than VND, and this algorithm produces two solutions similar to the best-known solutions (S12T10 and S12T14). TA and RTR average deviation (AVD) is better than the AVD of GA. The AVD from both algorithms is larger than the AVD by Mjirda et al. [2]. The CPU time is given in Table 3. Although the CPU time cannot be compared directly due to the differences in hardware used, we can say that the proposed algorithms use reasonable CPU time. Table 3 shows that TA consumes more CPU time than RTR. The route of the best solution found by RTR can be seen in Table 4, Figs. 6 and 7.

Table 2. Comparison of IRP solutions.

Datasets	Best solution	Moin et al. [1]	Mjirda et al. [2]		TA	RTR
		GA	VND	VNS		
S12T5	**1961.71**	2096.75	**1961.71**	**1961.71**	2069.48	1961.73
S12T10	**4002.85**	4333.27	**4002.85**	**4002.85**	4320.54	**4002.85***
S12T14	**5635.77**	6115.19	**5635.77**	**5635.77**	6092.42	**5635.77***
S20T5	**2861.26**	3143.39	3045.08	**2861.26**	3121.12	3121.12
S20T10	**5944.72**	6499.4	6098.15	**5944.72**	6242.24	6242.24
S20T14	**8422.02**	9208.43	8589.36	**8422.02**	8683.44	8683.44
S20T21	**12700.27**	13948.41	12944.08	**12700.27**	12929.74*	12929.74*
S50T5	**5084**	5618.09	5144.23	**5084**	5548.94	5478.37
S50T10	**10694.6**	11642	10787.88	**10694.6**	11668.54	10956.73
S50T14	**15384.96**	16987	15399.82	**15384.96**	16797.98	16797.98
S50T21	**23497.4**	26448.77	23500.94	**23497.4**	25281.47	24427.61
S98T5	**557287.97**	561168.21	575790.06	**557287.97**	585810.02	585810.02
S98T10	**1119767.72**	1124797.57	1160921.87	**1119767.72**	1170913.23	1160921.87*
S98T14	**1567904.75**	1571652.32	1567904.75	1577951.04	1592512.08	1592512.08
Average deviation		9.49%	1.58%	0.05%	6.20%	3.77%

***notes: bold = best solution, underline = better than GA, * = same or better than VND.**

Table 3. CPU time (in second).

Datasets	Moin et al. [1]		Mjirda et al. [2]				TA/RTR			
	GA (#v)	CPU (s)	VND (#v)	CPU (s)	VNS (#v)	CPU (s)	(#v)	CPU (s)	(#v)	CPU (s)
S12T5	14	58.48	14	**0.10**	14	0.10	14	85	14	12
S12T10	29	111.23	29	**0.20**	29	0.20	29	366	29	8
S12T14	41	120.31	41	**0.30**	41	0.31	41	425	41	3
S20T5	21	31.81	23	**0.40**	21	27.20	22	209	22	32
S20T10	43	126.03	46	**2.74**	43	98.34	44	222	44	35*
S20T14	61	360.33	65	**8.45**	63	113.36	61	665	61	52*
S20T21	92	255.83	98	**17.79**	94	185.99	91	756	91	77*
S50T5	45	133.40	46	**3.55**	45	93.03	47	199	46	19*
S50T10	95	226.01	99	**27.68**	96	183.79	96	312	92	55*
S50T14	135	238.07	139	55.58	137	212.07	136	410	136	**48***
S50T21	209	496.72	214	140.51	211	370.40	209	789	210	**140***
S98T5	57	476.77	59	**76.50**	57	799.60	57	612	57	89
S98T10	113	1307.26	119	**330.39**	114	1128.34	113	1818	119	710
S98T14	159	1589.71	160	**1353.51**	161	1354.80	159	1976	159	1612

***notes: #v = number of vehicles, CPU = central processing unit (time in second), bold = best CPU time, underline = better than GA, * = same or better than VNS.**

Table 4. Best solution route of RTR algorithm.

Datasets	Period	Route number	Route	Length per trip	Product pick-up per trip	Cost per trip
S12T10	1, 3-10	1	0 – 9 – 8 – 6 – 3 – 13 – 0	134.82	10	154.82
		2	0 – 7 – 10 – 12 – 11 – 13 – 0	99.62	9	119.62
		3	0 – 2 – 4 – 5 – 1 – 13 – 0	113.78	9	133.78
	2	1	0 – 9 – 7 – 8 – 10 – 12 – 11 – 13 – 0	128.37	8	148.37
		2	0 – 2 – 6 – 3 – 4 – 5 – 1 – 13 – 0	160.48	10	180.48
S12T14	1, 3-14	1	0 – 9 – 8 – 6 – 3 – 13 – 0	134.82	10	154.82
		2	0 – 7 – 10 – 12 – 11 – 13 – 0	99.62	9	119.62
		3	0 – 2 – 4 – 5 – 1 – 13 – 0	113.78	9	133.78
	2	1	0 – 9 – 7 – 8 – 10 – 12 – 11 – 13 – 0	128.37	8	148.37
		2	0 – 2 – 6 – 3 – 4 – 5 – 1 – 13 – 0	160.48	10	180.48

***notes: total costs of S12T10 = 4002.85, total costs of S12T14 = 5635.77.**

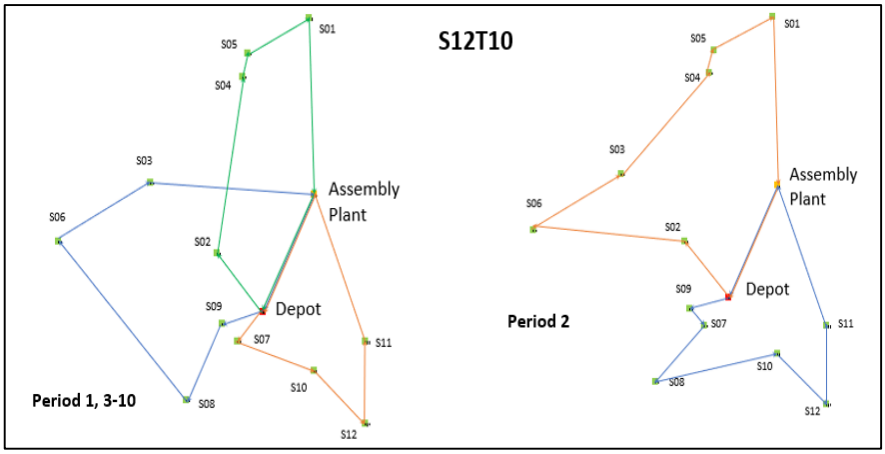

Fig. 6. Best solution route of S12T10.

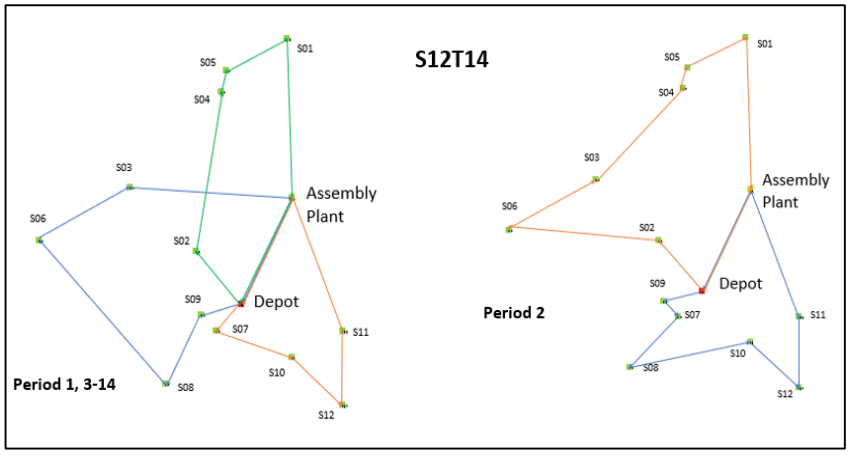

Fig. 7. Best solution route of S12T14.

5. Conclusions

IRP many-to-one distribution network is addressed in this paper. The initial solution was obtained from the LCI-VCM algorithm. TA and RTR based algorithm were used to improve the initial solution. TA and RTR algorithms produced good results when compared to the previous results from the literature. In this paper, we introduced the updating scheme of the threshold value for the TA and the integration between local search and LCI-VCM algorithms for TA and RTR. To improve the proposed algorithms further investigation can be done in determining the number threshold percentage, incremental value range, a rule for updating threshold scheme, number of acceptances for the TA algorithm, and finding better values of *M*, *K*, and *I* for RTR algorithm. Applying more local search in the search process can also be considered. The algorithms can also be developed to solve an IRP variant called the multi-depot, multi-depot and multi-product IRP.

Nomenclatures

$A = \{n + 1\}$	Assembly plant node
a_{it}	Total amount of product to be picked-up at supplier i in period t
C	Vehicle capacity per trip
$D = \{0\}$	Depot node
d_{it}	Demand from supplier i product i in period t
F	Fixed charge cost per trip
h_i	Inventory holding cost for product i supplier i per period
$iter_{max}$	Maximum number of iterations in repairing the route using local search
inv_{it}	Inventory level of product i supplier i at the end of period t
$NA_{increment}$	Amount of the reduction in the threshold value for each iteration
NA_{min}	Minimum number of solutions received each iteration
$percentTA_{increment}$	Amount of the reduction in threshold value for each iteration
$percenTTA_{max}$	Maximum number of percentage threshold for receiving a solution that is generated
q_{ijt}	Product quantity transported from node i to j in period t

$repeatTA_{max}$	Maximum number of repetitions made after the threshold value becomes 0
r_{ij}	Travel distance from node i to j where $r_{ij} = r_{ji}$ with $i \neq j$
$S = \{1, 2, \dots, n\}$	A set of suppliers that supplies product i where supplier $i \in S$
V	Travel cost per unit travel distance
x_{ijt}	Number of times visited from node i to j by vehicles in period t
Greek Symbols	
$\tau = \{1, 2, \dots, T\}$	Period t where $t \in \tau$

References

1. Moin, N.H.; Salhi, S.; and Aziz, N.A.B. (2011). An efficient hybrid genetic algorithm for the multi-product multi-period inventory routing problem. *International Journal Production Economics*, 133(1), 334-343.
2. Mjirda, A.; Jarboui, B.; Macedo, R.; and Hanafi, S. (2012). A variable neighborhood search for the multi-product inventory routing problem. *Electronic Notes in Discrete Mathematics*, 39, 91-98.
3. Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3), 345-358.
4. Abdelmaguid, T.F.; Dessouky, M.M.; and Ordonez, F. (2009). Heuristic approaches for the inventory-routing problem with backlogging. *Computers and Industrial Engineering*, 56(4), 1519-1534.
5. Coelho, L.C.; Cordeau, J.-F.; and Laporte, G. (2012). The inventory-routing problem with transshipment. *Computers and Operations Research*, 39(11), 2537-2548.
6. Hiassat, A.; Diabat, A.; and Rahwan, I. (2017). A genetic algorithm approach for location-inventory-routing problem with perishable products. *Journal of Manufacturing Systems*, 42, 93-103.
7. Dror, M.; Ball, M.; and Golden, B. (1985). A computational comparison of algorithms for the inventory routing problem. *Annals of Operations Research*, 4(1), 1-23.
8. Agra, A.; Requejo, C.; and Rodrigues, F. (2018). A hybrid heuristic for a stochastic production-inventory-routing problem. *Electronic Notes in Discrete Mathematics*, 64, 345-354.
9. Peres, I.T.; Repolho, H.M.; Martinelli, R.; and Monteiro, N.J. (2017). Optimization in inventory-routing problem with planned transshipment: A case study in the retail industry. *International Journal of Production Economics*, 193, 748-756.
10. Moin, N.H.; Halim, H.Z.A.; and Yuliana, T. (2014). Metaheuristics for multi-products inventory routing problem with time varying demand. *Proceedings of the 21st National Symposium on Mathematical Sciences (SKSM)*. Penang, Malaysia, 3-9.
11. Wong, L.; and Moin, N.H. (2017). Ant colony optimization for split delivery inventory routing problem. *Malaysian Journal of Computer Sciences*, 30(4), 333-348.
12. Xiao, N.; and Rao, Y.L. (2016). Multi-product multi-period inventory routing optimization with time window constraints. *International Journal of Simulation Modelling*, 15(2), 352-364.

13. Al-e-hashem, S.M.J.M.; and Rekik, Y. (2014). Multi-product multi-period inventory routing problem with a transshipment option: A green approach. *International Journal of Production Economics*, 157, 80-88.
14. Noor, N.M.; and Shuib, A. (2015). Multi-depot instances for inventory routing problem using clustering techniques. *Journal of Industrial and Intelligent Information*, 3(2), 97-101.
15. Bard, J.F.; and Nananukul, N. (2009). Heuristics a multi-period inventory routing problem with production decisions. *Computers and Industrial Engineering*, 57(3), 713-723.
16. Archetti, C.; Bertazzi, L.; Hertz, A.; and Speranza, M.G. (2012). A hybrid heuristic for an inventory routing problem. *INFORMS Journal on Computing*, 24(1), 31 pages.
17. Singh, T.; Arbogast, J.E.; and Neagu, N. (2015). An incremental approach using local-search heuristic for inventory routing problem in industrial gases. *Computers and Chemical Engineering*, 80, 199-210.
18. Tarantilis, C.D.; Kiranoudis, C.T.; and Vassiliadis, V.S. (2004). A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *European Journal of Operational Research*, 152(1), 148-158.
19. Tarantilis, C.D.; Kiranoudis, C.T.; and Vassiliadis, V.S. (2003). A list based threshold accepting metaheuristics for the heterogeneous fixed fleet vehicle routing problem. *Journal of the Operational Research Society*, 54(1), 65-71.
20. Li, F.; Golden, B.; and Wasil, E. (2007). A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Computers and Operations Research*, 34(9), 2734-2742.
21. Gendreau, M.; Hertz, A.; and Laporte, G. (1992). New insertion and postoptimization procedures for the traveling salesman problem. *Operations Research*, 40(6), 1086-1094.
22. Dueck, G.; and Scheuer, T. (1990). Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90(1), 161-175.
23. Imran, A.; Salhi, S.; and Wassan, N.A. (2009). A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, 197(2), 509-518.
24. Dueck, G. (1993). New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational Physics*, 104(1), 86-92.