Research Article

A Memory Integrated Artificial Bee Colony Algorithm with Local Search for Vehicle Routing Problem with Backhauls and Time Windows

Naritsak Tuntitippawan*
Department of Industrial Engineering, Faculty of Engineering, King Mongkut's University of Technology North Bangkok, Bangkok, Thailand

Krisada Asawarungsaengkul
Operations Research and Engineering Management Research Center
Department of Industrial Engineering, Faculty of Engineering, King Mongkut's University of Technology North Bangkok, Bangkok, Thailand

* Corresponding author. E-mail: nst@kmutnb.ac.th DOI: 10.14416/j.ijast.2018.03.001
Received: 1 June 2017; Accepted: 26 June 2017; Published online: 5 March 2018

Abstract
The vehicle routing problem is a logistics problem which receives much attentions in logistics management. This paper presents a Memory integrated Artificial Bee Colony Algorithm (MABC) to solve the Vehicle Routing Problem with addition of Backhauls and Time Windows, known as the VRPBTW. In VRPBTW, a homogenous fleet of vehicles are utilized to deliver goods to linehaul customer set and pick up goods from backhaul customer set. Vehicle capacity, sequence of linehaul/backhaul and time windows are the three of major constraints for this problem. The VRPBTW's objective is to determine the optimal routes with minimum of total distance that satisfies all constraints. The proposed algorithm was tested on Gelinas's VRPBTW benchmark problems. MABC is developed by adding the memory to Artificial Bee Colony (ABC). The local search algorithms including λ-interchange and 2-opt* are utilized to search for the better solutions. The computational results show that MABC significantly yields the good solutions in terms of total travelling distance. Finally, it can be concluded that the performance of the proposed MABC algorithm is superior to the existing studies in term of quality solution.

Keywords: Memory integrated artificial bee colony algorithm, Vehicle routing problem, Backhauls, Time windows, Local search

1 Introduction

Logistics management mainly focuses in cost, time, and quality. The Vehicle Routing Problem (VRP) is a vital operation for logistics. Fully utilization of the fleet of vehicle is a major concern for VRP. Assigning vehicle to pick up goods during backhaul can enhance utilization of the vehicle. In addition, the period for delivery and pick up goods is also the restriction in logistics since warehouse or customer usually schedule a specific period to pick up or receive goods. Vehicle Routing Problem with Backhauls and Time Windows (VRPBTW) is an extension of the classical Vehicle Routing Problem (VRP). The VRPBTW contains two subsets of customers, known as delivery customer (linehaul) and pickup customer (backhaul). Each linehaul customer requires a given quantity of goods from depot, and a given quantity of goods is collected from each backhaul customer and returned to the depot. Because deliveries are usually higher priority

Please cite this article as: N. Tuntitippawan and K. Asawarungsaengkul, "A memory integrated artificial bee colony algorithm with local search for vehicle routing problem with backhauls and time windows," *KMUTNB Int J Appl Sci Technol*, vol. 11, no. 2, pp. 85–92, Apr.–Jun. 2018.

than pickup; therefore, the linehaul customers must be visited before the backhaul customers in each route. Each customer must be visited within a specified time windows. The vehicle is allowed to arrive before the lower bound of time windows, and waits until the customer is available. However, the vehicle is not allowed to arrive after the upper bound of time windows. We consider this restriction as the hard time windows constraint for the vehicle routing problem with backhauls. Due to the finite capacity of each vehicle, the routes must be satisfied the capacity constrain. The objective is to minimize the total travelling distance of vehicles, while all constraints must be satisfied.

Literatures on logistics management can be found in [1], [2]. The decision making model on the excavated soil transportation with transportation cost and time constraint was proposed by [1]. [2] presented the location routing problem which intended to minimize the transportation and depreciation cost of collection and distribution in a rubber market. In this paper, the vehicle routing problem is our interest. The VRPBTW is a NP-hard problem [3]. Because it can be considered as the VRP, which is well-known NP-hard problem, only backhauls and time windows are additionally considered. Research on VRPBTW has been mainly focused on heuristic and meta-heuristic approaches which can produce high quality of solutions within reasonable computational time. The review on metaheuristic for VRPBTW are provided as follows. A Genetic Algorithm (GA) with greedy route construction heuristic for VRPBTW was performed by [3]. The routing construction heuristic with different local search heuristics for improving initial solution was proposed by [4]. A Tabu Search (TS) heuristic for VRPBTW was proposed by [5]. The result was compared with solution of other heuristic approach and the optimal solutions. An ant system was presented to solve the VRPBTW by [6]. This ant system approach was based on insertion algorithm proposed by [7]. A Guided Local Search Approach (GLSA) and section planning technique for VRPBTW were also presented by [8].

Recently, a Differential Evolution Algorithm (DEA) was presented by [9]. DEA was tested with benchmark problem and compared the results with the optimal and best known solutions. They could find some of the results that were better than the best known solutions. Moreover, an advanced Hybrid Metaheuristic Algorithm (HMA) was proposed by [10]. The HMA combines Simulated Annealing (SA) and Tabu Search (TS) meta-heuristic to solve VRPBTW. The HMA was applied to the benchmark problem and the results showed that HMA could also find better solution than the best known solutions. In addition, [11] proposed an Artificial Bee Colony (ABC) algorithm with λ -interchange local search technique for VRPBTW. The results shown that the proposed algorithm is comparable to HMA and GA.

In this paper, we present Memory integrated Artificial Bee Colony Algorithm (MABC) and additional local search techniques, which are λ -interchange and 2-opt* to reduce the time spent searching for solution and increase chances of finding better solution for VRPBTW. The Gelinas's benchmark problems in [12] are used to evaluate the performance of the MABC algorithm.

2 Problem Definition

VRPBTW is formulated based on the existing mathematical formulation for VRPB, with each vertex representing a customer [13]. Let $G(V, A)$ be a graph with a vertex set $V = \{0\} \cup L \cup B$, where the subset $\{0\}$, L and B represent to the depot, linehaul customer nodes, and backhaul customer nodes, respectively. A denotes all possible arcs that are connected between nodes. $\overline{G}(\overline{V}, \overline{A})$ be a graph in G which is defined as:

$\overline{A} = A_1 \cup A_2 \cup A_3$ and $\overline{V} = V$

$A_1 = \{(i, j) \in A : i \in L \cup \{0\}, j \in L\}$,

$A_2 = \{(i, j) \in A : i \in L, j \in B \cup \{0\}\}$,

$A_3 = \{(i, j) \in A : i \in B, j \in B \cup \{0\}\}$

The arc set $\overline{A}$ can be divided into three subsets. The arc set A_1 represents all of arcs from the depot to linehaul customer nodes and from linehaul customer nodes to linehaul customer nodes. A_2 represents all of arcs from linehaul customer nodes to backhaul customer nodes and from linehaul customer nodes to the depot. A_3 represents all of arcs from backhaul customer nodes to backhaul customer nodes and from backhaul customer nodes to the depot. Additionally, for each $i \in V$, $\Delta_i^+ = \{j : (i, j) \in \overline{A}\}$ and $\Delta_i^- = \{j : (j, i) \in \overline{A}\}$ define the forward and backward star of i, respectively. Forward star of i defines the set of vertices j that have direct path from vertex i and backward star of i defines the set of vertices j that have direct path to vertex i.

Using these definitions, the VRPBTW was formulated as following and it is classified as the mixed-integer programming model [10].

Notations

K : Set of vehicles
d_i : Demand/supply for customer $i : \forall i \in \overline{V} \setminus \{0\}$
c_{ij} : Distance from node to node $j : (i, j) \in \overline{A}$
t_{ij} :Travel time between node to node $j : (i, j) \in \overline{A}$
a_i : Earliest arrival time at customer $i : \forall i \in \overline{V} \setminus \{0\}$
b_i : Latest arrival time at customer $i : \forall i \in \overline{V} \setminus \{0\}$
s_i : Service time at customer $i : \forall i \in \overline{V} \setminus \{0\}$
u_k : Capacity of vehicle $k : \forall k \in K$
$T_{\max}$: Maximum route time allowed for vehicles
M : Large penalty value

Decision variables

x_{ijk} : 1 if vehicle k travels from customer i to j, and 0 otherwise
w_{ik} : Service start time of vehicle k for customer i.

Objective

$$Min\, z = \sum_{(i,j)\in \overline{A}} \sum_{k\in K} c_{ij} x_{ijk} \tag{1}$$

Subject to:

$$\sum_{i\in \Delta_j^-} \sum_{j\in L} d_j x_{ijk} \le u_k \qquad \forall k \in K \tag{2}$$

$$\sum_{i\in \Delta_j^-} \sum_{j\in B} d_j x_{ijk} \le u_k \qquad \forall k \in K \tag{3}$$

$$\sum_{i\in \Delta_0^+} x_{0jk} = 1 \qquad \forall k \in K \tag{4}$$

$$\sum_{i\in \Delta_0^-} x_{i0k} = 1 \qquad \forall k \in K \tag{5}$$

$$\sum_{i\in \Delta_j^-} \sum_{k\in K} x_{ijk} = 1 \qquad \forall j \in \overline{V} \setminus \{0\} \tag{6}$$

$$\sum_{j\in \Delta_i^+} \sum_{k\in K} x_{ijk} = 1 \qquad \forall i \in \overline{V} \setminus \{0\} \tag{7}$$

$$\sum_{i\in \Delta_j^-} x_{ijk} = \sum_{i\in \Delta_j^+} x_{jik} \qquad \forall j \in \overline{V} \setminus \{0\}, \forall k \in K \tag{8}$$

$$t_{0j} - w_{jk} \le M(1 - x_{0jk}) \qquad \forall j \in \overline{V} \setminus \{0\}, \forall k \in K \tag{9}$$

$$w_{ik} + s_i + t_{i0} - w_{0k} \le M(1 - x_{i0k}) \quad \forall i \in \overline{V} \setminus \{0\}, \forall k \in K \tag{10}$$

$$w_{ik} + s_i + t_{ij} - w_{jk} \le M(1 - x_{ijk}) \qquad \forall i \in \overline{V} \setminus \{0\}, \forall j \in \overline{V} \setminus \{0\}, \forall k \in K \tag{11}$$

$$a_i \le w_{ik} \le b_i \qquad \forall i \in \overline{V} \setminus \{0\}, \forall k \in K \tag{12}$$

$$0 \le w_{0k} \le T_{\max} \qquad \forall k \in K \tag{13}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in \overline{V}, \forall j \in \overline{V}, \forall k \in K \tag{14}$$

$$w_{ik} \ge 0 \qquad \forall i \in \overline{V}, \forall k \in K \tag{15}$$

According to the model, objective (1) is to minimize the sum of the route distance. Constraints (2) and (3) are the capacity restrictions which ensure that load of vehicle k will not exceed the capacity in linehaul and backhaul customers, respectively. Constraints (4)–(8) are the classic VRP constraints. Constraints (4) and (5) say that each route must leave from the depot and must return to the depot. Constraints (6) and (7) ensure that the exactly one arc enters and leaves each customer node. Constraint (8) states that a vehicle leaves from the same customer node it has entered. Constraints (9)–(13) define time windows that each customer must be serviced in the time windows and also these constraints prevent subtour. Finally, constraints (14) and (15) guarantee that all decision variables must be the proper variable types.

3 Proposed Algorithm

This section describes the MABC algorithm, generation of initial solution and local search operations which are λ -interchange and 2-opt*. These local searches improve the solution by exchanging customer nodes between 2 selected routes.

3.1 *Memory integrated Artificial Bee Colony Algorithm (MABC)*

The ABC algorithm is an evolutionary algorithm that is inspired by the natural foraging behavior of honey bees in finding food or nectar source around the hive. The solution of the problem are considered as food source and the groups of bees try to exploit the food sources in the hope of finding good quality nectar or high quality of solutions.

1. Generate a set of initial food sources (or initial solutions) X_i.
2. Evaluate the fitness $f(X_i)$ for each food source.
3. Set $v = 0$, $l_i = 0$, and *MEM* of $X_i = \varnothing$; $i = 1,2,...,Nb$
4. **While** ($v <= MaxIteration$) do
   a. **For** each X_i, generate X_i' by randomly select pair of routes ($\boldsymbol{r_i}$, $\boldsymbol{r_j}$) that do not appear in *MEM* of X_i and apply two local search operations,
      If $f(X_i') > f(X_i)$ **Then** $X_i = X_i'$, $l_i = 0$, apply *removing procedure* to ($\boldsymbol{r_i}$, $\boldsymbol{r_j}$) as in section 3.5, **Else** $l_i = l_i + 1$ and *add* to *MEM* of X_i **End If, Next**
   b. Set $f(\hat{X}) = 0$, F = $\varnothing$
   c. **For** each onlooker
      Select food source X_i by using roulette wheel selection method and generate X_i' by randomly select pair of routes ($\boldsymbol{r_i}$, $\boldsymbol{r_j}$) that do not appear in *MEM* of X_i and apply two local search operations,
      If $f(X_i') > f(\hat{X})$**Then** $\hat{X} = X_i'$ **End If, Next**
   d. **For** each food source X_i, **If** $f(X_i) < f(\hat{X})$ **Then** $i \in F$ **End If, Next**
      Set $\tilde{X} = X_j$ where $j = \arg[\max_{j \in F} \{l_j\}]$
      For each food source X_i, **If** $X_i = \tilde{X}$ **Then** Replace X_i with $\hat{X}$, $l_i = 0$, replace *MEM* of X_i with *MEM* of $\hat{X}$ and **Else** $l_i = l_i + 1$ **End If, Next**
   e. **For** each food source X_i, **If** $l_i = limit$ **Then** Replace X_i with a randomly generated solution, $l_i = 0$, and set *MEM* of $X_i = \varnothing$ **End if, Next**
   f. $v = v + 1$

Loop

Figure 1: Memory integrated artificial bee colony algorithm for solving the VRPBTW.

In the ABC algorithm, the bees are divided into three types including employed bees, onlookers and scouts [14]. Employed bees are responsible for exploiting available food sources and gathering required information. The information is shared to onlookers then the onlookers select existing food source to be further explored. Employed bee can abandon the old food source when the onlookers can find the best food source. In this case, the employed bee associated with the old food source will be assigned to the best food source found by onlookers. However, any food sources will also be abandoned if the iteration limit is reached. After that, the employed bee becomes a scout bee to look for new food source randomly.

MABC algorithm which adopts ABC algorithm in [11] with addition of memory and local search can be performed by following step 1 to 4. Step 1 to 3 are for the initial solutions and parameters setting. Step 4 describes an improvement on solution (or food source) X_i by generating X_i' with local search processes including λ-interchange and 2-opt*. The best results from both local searches will be used to generate X_i'. Next, the additional of memory (*MEM*) concept as in step 4 is proposed to avoid ABC algorithm to fall into the local optimal too fast. The memory can be considered as Tabu list or forbidden list. Additional of memory is used for memorizing the search paths in solution X_i that has been proceeded by local searches without any improvement. The MABC algorithm can be summarized as in Figure 1.

3.2 *Initial solutions*

An initial solution is constructed based on the weighted time oriented nearest neighbor heuristic process that was proposed by Solomon [7] and this technique was applied with ABC algorithm in [11]. The closet of node i to node j, c_{ij} is introduced to generate the initial solutions appropriately.

In term of closest, c_{ij} is determined by three cost factors [15]. The first factor as in (16) is distance from customer i's location to customer j's location, d_{ij}

$$d_{ij} = \sqrt{(y_i - y_j)^2 + (x_i - x_j)^2} \tag{16}$$

where (x_i, y_i) is geometric location of customer i.

The second factor is urgency, u_{ij} in (17). Urgency of customer j is time left to service customer j after it serves customer i.

$$u_{ij} = b_j - (\mathrm{T}_i + s_i + t_{ij}) \quad (17)$$

where T_i denotes the service start time of customer, s_i is defined as the service time of customer i, t_{ij} denotes travel time between customer i's location to customer j's location.

The third factor is the waiting time, w_{ij} as in (18) which is the time remaining until the vehicle's last possible service start.

$$w_{ij} = \max\{0, a_j - (T_i + s_i + t_{ij})\} \quad (18)$$

The closest term can be formulated as (19)

$$c_{ij} = \theta_d d_{ij} + \theta_u u_{ij} + \theta_w w_{ij} \quad (19)$$

where θ_d, θ_u, θ_w, are the weight of distance, urgency, and waiting time, respectively.

The lower value of c_{ij} is more preferable. To generate different initial food sources, the initial weight of distance, urgency, and waiting time are randomly searched from range 0 to 1.

The procedure of this heuristic starts every routes by finding the unrouted customer closest (in term of c_{ij}) to the depot. Next, calculate c_{ij} for unrouted customer j to the last customer of each route i and also calculate c_{ij} for unrouted customer j to the depot. Select the lowest c_{ij} that satisfies both capacity and time windows constraint then add customer j to corresponding route. If closest to the depot, the new vehicle route is introduced. The approach is repeated until all customers are assigned.

3.3 *λ -interchange local search*

λ -interchange local search was introduced by [16]. It improves the solution by exchange of nodes between pair routes that are selected randomly. The number of node using exchange ranged from 0 to λ. The exchange will be performed to all possible cases that can be exchanged. Then, the best result is selected after doing the λ -interchange local search. For example, if λ =2 all possible exchange will have 8 cases: exchange with (0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1) and (2,2). Let route i and route j are selected routes to be performed the local search by exchange with operator (1,2) which means that route i will use 1 node to exchange and route j will use 2 nodes to exchange. Moreover, assuming that route i has 6 nodes and route j has 4 nodes. In this case, there are 181 possible results then the best result will be the solution of this local search. Example of λ -interchange local search with operators (1,1) and (1,2) are shown in Figure 2.

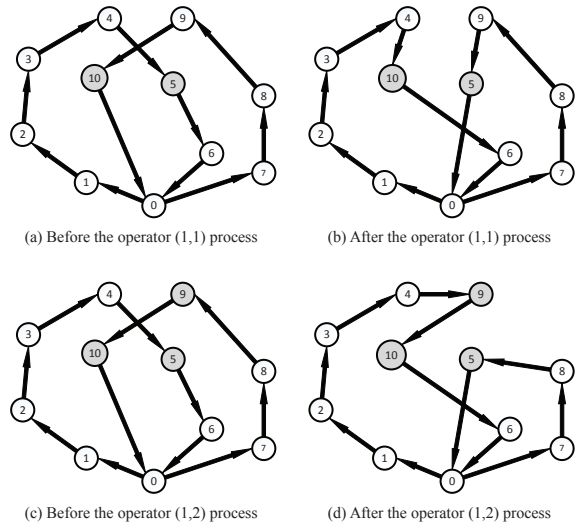

Figure 2: Interchange of nodes between two routes.

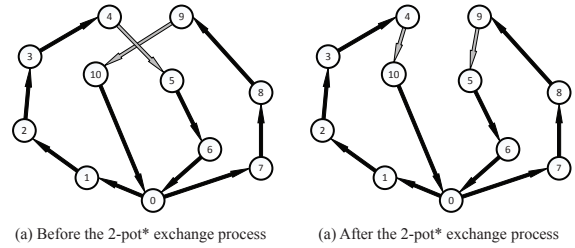

Figure 3: 2-opt* exchange.

3.4 *2-opt* exchange local search*

The 2-opt* procedure was proposed by [17]. It is able to improve the solution by exchange nodes in pair of routes that are randomly selected. After that select arc from selected route, for example, suppose that route i and route j are selected. Result of 2-opt* will occur by taking a node located next to selected arc on route i and taking a node located next to selected arc of route j and then exchange nodes between pair of routes. An example of 2-opt* procedure is depicted in Figure 3. Assuming that route i has 6 nodes and route j has 4 nodes, these lead to 35 possible results then the best result will be the solution of this local search.

3.5 *Removing procedure for the pair of routes*

Since the memory is integrated to ABC algorithm, the forbidden list of each food source is introduced. If route (r_i, r_j) is added to forbidden list, it should be removed after a certain period of time. The removing procedure will release all pairs of routes that are consisted of r_i or r_j from the forbidden list (or *MEM*).

4 Computational Results

The proposed algorithm, MABC with λ -interchange and 2-opt * local search, was implemented via Visual Basic programming language using a laptop PC intel Core i5, 2.3 GHz processor and 4 GB memory. To evaluate the performance, the instance of VRPBTW generated by Gelinas *et al.* [12] have been tested. These benchmark problems were selected from the first 5 problems of the r1 series developed by Solomon [7] and then randomly chooses 10, 30 and 50% of the 100 customer nodes to be backhaul customer, without any changes to the other attributes. Gelinas *et al.* generated additional test problems by considering only the first 25 and first 50 customer nodes. Therefore 45 test problems for the VRPBTW were generated. However, in this paper, we select 34 problems of which the optimal solutions are known to the computational experiments.

The parameters for MABC consist of: the Number of bees (*Nb*), *limit*, λ(Lambda) and maximum iteration (*MaxIteration*). Design of experiments was conducted by [11] in order to determine the appropriate parameters which are employed to these experiments. Therefore, the parameters for MABC are as following: $Nb = N\_node/10$, $limit = N\_node*5$, $\lambda = 2$ and *MaxIteration* = 40,000. The number of customer is denoted by *N_node*.

Table 1 to Table 3 display the optimal solution, the best result of MABC, and also the compared algorithms which are ABC II, HMA, GA, DEA referring to the proposed algorithms in [11], [10], [3] and [9] respectively. The $\%Gap_{opt}$ for Gelinas's benchmark problem sets with known optimal solution are shown as well. The $\%Gap_{opt}$ can be described by the following equation:

$$\%Gap_{opt} = \frac{metaheuristic\ solution - optimal\ solution}{optimal\ solution} \times 100\%$$

In the Table 1, the results for 25 customer nodes, show that the proposed MABC is able to find the optimum solutions for 10 problems and yields 1 new best solution in problem R104a. The total distance of 8,432.1 and $\%Gap_{opt}$ of 0.32% for proposed MABC reveal that MABC is the best for small-size problems

Table 1: Computational result for 25 customer nodes

Problem	Optimal Solutions	MABC			ABCII	HMA	GA	DEA	$\%Gap_{opt}$				
		Dist	NV	%CV	Dist	Dist	Dist	Dist	MABC	ABCII	HMA	GA	DEA
R101a	643.4	643.4	10	0.00%	643.4	643.4	643.4	643.4	0.00%	0.00%	0.00%	0.00%	0.00%
R101b	711.1	721.8	10	0.00%	721.8	721.8	721.8	721.8	1.50%	1.50%	1.50%	1.50%	1.50%
R101c	674.5	676.8	10	0.00%	676.8	676.8	682.3	676.8	0.34%	0.34%	0.34%	1.16%	0.34%
R102a	563.5	563.5	7	0.00%	563.5	563.5	563.5	565.3	0.00%	0.00%	0.00%	0.00%	0.32%
R102b	622.3	628.1	9	0.00%	628.1	628.1	622.3	629	0.93%	0.93%	0.93%	0.00%	1.08%
R102c	584.4	584.4	9	0.00%	584.4	584.4	584.4	585.3	0.00%	0.00%	0.00%	0.00%	0.15%
R103a	476.6	476.6	5	0.00%	476.6	478.8	476.6	489	0.00%	0.00%	0.46%	0.00%	2.60%
R103b	507	507	10	0.00%	507	507	507	510.9	0.00%	0.00%	0.00%	0.00%	0.77%
R103c	475.6	483	6	0.00%	483	483	483	495	1.56%	1.56%	1.56%	1.56%	4.08%
R104a	452.5	452.5	5	0.09%	453.8	453.8	452.8	459.1	0.00%	0.29%	0.29%	0.07%	1.46%
R104b	467.6	468.5	6	0.00%	468.5	468.5	468.5	469.6	0.19%	0.19%	0.19%	0.19%	0.43%
R104c	446.8	446.8	5	0.00%	446.8	446.8	446.8	458.7	0.00%	0.00%	0.00%	0.00%	2.66%
R105a	565.1	565.1	7	0.00%	565.1	565.1	565.1	565.1	0.00%	0.00%	0.00%	0.00%	0.00%
R105b	623.5	623.5	8	0.32%	628	623.5	630.2	630.2	0.00%	0.72%	0.00%	1.07%	1.07%
R105c	591.1	591.1	8	0.00%	591.1	592.1	592.1	598.5	0.00%	0.00%	0.17%	0.17%	1.25%
Total	8405	8432.1	0	0.00%	8437.9	8436.6	8439.8	8497.7	0.32%	0.39%	0.38%	0.41%	1.10%

Table 2: Computational result for 50 customer nodes

Problem	Optimal Solutions	MABC			ABCII	HMA	GA	DEA	%Gap$_{opt}$				
		Dist	NV	%CV	Dist	Dist	Dist	Dist	MABC	ABCII	HMA	GA	DEA
R101a	1122.3	1133.3	15	0.03%	1134	1135.8	1138.1	1138.3	0.98%	1.04%	1.20%	1.41%	1.43%
R101b	1191.5	1191.6	16	0.32%	1191.6	1191.6	1192.7	1245.8	0.01%	0.01%	0.01%	0.10%	4.56%
R101c	1168.6	1183.9	16	0.00%	1183.9	1183.9	1183.9	1183.9	1.31%	1.31%	1.31%	1.31%	1.31%
R102a	974.7	976.5	12	0.00%	976.5	976.8	976.8	978.7	0.18%	0.18%	0.22%	0.22%	0.41%
R102b	1024.8	1054.6	14	0.00%	1054.6	1046	1029.2	1046	2.91%	2.91%	2.07%	0.43%	2.07%
R102c	1057.2	1059.7	14	0.00%	1059.7	1061.6	1059.7	1115.3	0.24%	0.24%	0.42%	0.24%	5.50%
R103a	811.4	813.8	9	0.52%	821.6	815.5	813.3	831.1	0.30%	1.26%	0.51%	0.23%	2.43%
R103b	882.8	886.3	11	0.33%	887.1	889.3	892.7	895.1	0.40%	0.49%	0.74%	1.12%	1.39%
R103c	882.1	884.3	11	0.07%	885.1	887.7	885.5	887.7	0.25%	0.34%	0.63%	0.39%	0.63%
R104c	733.6	733.6	8	0.27%	739.3	738.2	741.4	742.2	0.00%	0.78%	0.63%	1.06%	1.17%
R105a	970.6	973.7	11	0.33%	985.2	978.5	1002.5	972.8	0.32%	1.50%	0.81%	3.29%	0.23%
R105b	1007.5	1022.5	14	0.27%	1024.7	1026.7	1047.8	1030	1.49%	1.71%	1.91%	4.00%	2.23%
R105c	993.4	993.4	11	0.00%	993.4	996.2	1018	1022.2	0.00%	0.00%	0.28%	2.48%	2.90%
Total	12820.5	12907.2	0	0.00%	12936.7	12927.8	12981.6	13089.1	0.68%	0.91%	0.84%	1.26%	2.10%

Table 3: Computational result for 100 customer nodes

Problem	Optimal Solutions	MABC			ABCII	HMA	GA	DEA	%Gap$_{opt}$				
		Dist	NV	%CV	Dist	Dist	Dist	Dist	MABC	ABCII	HMA	GA	DEA
R101a	1767.9	1813.3	24	0.09%	1818.6	1811.6	1815	1881.6	2.57%	2.87%	2.47%	2.66%	6.43%
R101b	1877.6	1887.7	25	0.39%	1904.5	1891.1	1896.6	1925.9	0.54%	1.43%	0.72%	1.01%	2.57%
R101c	1895.1	1909.4	25	0.54%	1928.2	1911.2	1905.9	1930.2	0.75%	1.75%	0.85%	0.57%	1.85%
R102a	1600.5	1625.5	21	0.53%	1640.7	1623.7	1622.9	1649.8	1.56%	2.51%	1.45%	1.40%	3.08%
R102b	1639.2	1700.7	22	0.30%	1717.3	1724	1688.1	1758.2	3.75%	4.76%	5.17%	2.98%	7.26%
R102c	1721.3	1734.7	21	0.29%	1752.2	1759.8	1735.7	1777.1	0.78%	1.80%	2.24%	0.84%	3.24%
Total	10501.6	10671.3	0	0.00%	10761.5	10721.4	10664.2	10922.8	1.62%	2.47%	2.09%	1.55%	4.01%

The results for 50 customer nodes as in Table 2 show that MABC can determine the optimum solutions for 2 problems and obtains 5 new best solutions out of 13 instances considered this papers. The total distance of 12,907.2 and %Gap_{opt} of 0.68% of MABC algorithm confirms that MABC is still efficient to find the solutions for medium-size problems.

Table 3 displays the results for 100 customer nodes. Although, the MABC can obtain only 2 new best solutions (R101b and R102c) for 6 instances but the total distance of MABC algorithm are close to the optimal values. %Gap_{opt} of MABC algorithm are not significant different from GA. Moreover, the overall %CV for MABC are very lower which indicate that the MABC is capable to yield the solution with low variation.

5 Conclusions and Discussions

The MABC is proposed to solve the extension of the vehicle routing problem with addition of backhauls and time windows, known as the VRPBTW. The proposed algorithm is based on ABC algorithm with additional memory for improving the effectiveness in finding the solution. The initial solutions are generated by random weighted time oriented nearest neighbor heuristic. The λ -interchange and 2-opt* local search are used as the neighborhood search mechanism. To evaluate the efficiency of the proposed algorithm, MABC is tested with Gelinas's VRPBTW benchmark problems. The numerical experiments reveal that proposed MABC yield the solutions which are close to the optimal solutions with 0.89% of overall %Gap_{opt}. Moreover, MABC has superior performance in solving VRPBTW by obtaining 8 new best solutions which are better than other reference algorithms. It can be concluded that the proposed MABC is out perform for VRPBTW. For the future research, the proposed algorithm may be applied to other variants of vehicle routing problem such as

vehicle routing problem with pickup and delivery and open vehicle routing problem, etc. Moreover, the hybrid algorithm between two metaheuristics can also be developed to determine better solutions for the large-scale problem of VRPBTW.

Acknowledgements

This research was funded by Faculty of Engineering, King Mongkut's University of Technology North Bangkok, Thailand. This support is gratefully acknowledged.

References

[1] W. Meethom and T. Triwong, "A multi-attribute urban metro construction excavated soil transportation decision making model based on integrated fuzzy AHP and integer linear programming," *KMUTNB Int J Appl Sci Technol*, vol. 9, no. 3, pp. 153–165, 2016.

[2] S. Kaewploy and S. Sindhuchao. (2017). Solving the location routing problem of the central rubber market by tabu search. *KMUTNB Int J Appl Sci Technol* [Online]. 10(2), pp. 145–151. Available: http://ojs.kmutnb.ac.th/index.php/ijst/article/view/806/pdf_111

[3] J. Y. Potvin, C. Duhamel, and F. Guertin, "A genetic algorithm for vehicle routing with backhauling," *Applied Intelligence*, vol. 6, pp. 345–355, 1996.

[4] S. R. Thangiah, J. Y. Potvin, and T. Sun, "Heuristic approaches to vehicle routing with backhauls and time windows," *Computers & Operations Research*, vol. 23, pp.1043–1057, 1996.

[5] C. Duhamel, J. Y. Potvin, and J. M. Rousseau, "A tabu search heuristic for the vehicle routing problem with backhauls and time windows," *Transportation Science*, vol. 31, no. 1, pp. 49–59, 1997.

[6] M. ReimannKarl, K. Doerner, and R. F. Hartl, "Insertion based ants for vehicle routing problems with backhauls and time windows," in *Proceedings of the Third International Workshop, ANTS*, Brussels, Belgium, 2002, pp.135–148.

[7] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time windows constraints," *Operations Research*, vol. 35, pp. 254–265, 1987.

[8] Y. Zhong and M. H. Cole, "A vehicle routing problem with backhauls and time windows: A guided local search solution," *Transportation Research Part E*, vol. 41, pp. 131–144, 2005.

[9] I. Kucukoglu and N. Ozturk, "A differential evolution approach for the vehicle routing problem with backhauls and time windows," *Journal of Advanced Transportation*, vol. 48, pp. 942–956, 2014.

[10] I. Kucukoglu and N. Ozturk, "An advanced hybrid meta-heuristic algorithm for the vehicle routing problem with backhauls and time windows," *Computers & Industrial Engineering*, vol. 86, pp. 60–68, 2015.

[11] N. Tuntitippawan and K. Asawarungsaengkul, "An artificial bee colony algorithm with local search for vehicle routing problem with backhaul and time windows," *KKU Engineering Journal*, vol. 43, pp. 404–408, 2016.

[12] S. Gélinas, M. Desrochers, J. Desrosiers, and M. M. Solomon, "A new branching strategy for time constrained routing problems with application to backhauling," *Annals of Operations Research*, vol. 61, no. 1, pp. 91–109, 1995.

[13] P. Toth and D. Vigoo, *The vehicle routing problem*, Philadelphia: Society for Industrial and Applied Mathematics, 2001, pp. 198–199.

[14] W. Y. Szeto, Y. Wu, and S. C. Ho, "An artificial bee colony algorithm for the capacitated vehicle routing problem," *European Journal of Operational Research*, vol. 215, no. 1, pp. 126–135, 2011.

[15] K. W. Pang, "An adaptive parallel route construction heuristic for the vehicle routing problem with time windows constraints," *Expert Systems with Applications*, vol. 38, pp. 11939–11946, 2011.

[16] I. H. Osman and N. Christofides, "Capacitated clustering problems by hybrid simulated annealing and tabu search," *Internatioal Transactions in Operational Research*. vol. 1, no. 3, pp. 317–336, 1994.

[17] J. Y. Potvin, T. Kervahut, B. L. Garcia, and J. M. Rousseau, "A tabu search heuristic for the vehicle routing problem with time windows," Quebec Centre de Recherche sur les Transports, Universite de Montreal, Technical Report CRT-855, 1993.