CUCKOO SEARCH ALGORITHM FOR THE VEHICLE ROUTING PROBLEM WITH BACKHAULS AND TIME WINDOWS

การพัฒนาวิธีการจัดเส้นทางเดินรถซึ่งมีข้อจำกัดด้านรถเที่ยวกลับและตารางเวลาด้วยวิธีการค้นหาคำตอบจากการเลียนแบบพฤติกรรมของนกกาเหว่า

Tanawat Worawattawechai[1] Boonyarit Intiyot[2] and Chawalit Jeenanunta[3]
[1,2]Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University
[3]School of Management Technology, Sirindhorn International Institute of Technology (SIIT), Thammasat University

Abstract

A Vehicle Routing Problem with Backhauls and Time Windows (VRPBTW) involves two different subsets of customers known as linehauls and backhauls. The demands of the linehauls must be delivered before the backhaul pickups. The total demands of customers must not exceed a vehicle's capacity, and the time that a vehicle arrives at every customer must be within the required time windows. In this study, we present a cuckoo search (CS) algorithm, which is inspired from aggressive breeding behavior of cuckoo birds to solve this problem. Moreover, we proposed the nearest neighbor with roulette wheel selection method (NNRW) as an initial solution algorithm. The proposed method was tested on a set of benchmark instances. The results indicated that NNRW gave equal or better solutions than the improved nearest neighbor algorithm (INN). Furthermore, CS algorithm was compared with other methods from existing studies. Computational results show that our algorithm gave equivalent solutions to or better solutions than the best known solutions for the majority of small and medium-size instances. Hence, it is a competitive method for solving small and medium size VRPBTW problems.

Keywords: vehicle routing problems, backhaul, time window, cuckoo search

Corresponding Author
E-mail: i.am.t.tanawat@gmail.com

บทคัดย่อ

ปัญหาการจัดการเส้นทางเดินรถโดยมีข้อจำกัดด้านรถเที่ยวกลับและตารางเวลานั้นเกี่ยวข้องกับกลุ่มลูกค้าสองประเภท ได้แก่ ลูกค้าเที่ยวไป และลูกค้าเที่ยวกลับ โดยเราจะต้องนำสินค้าไปส่งให้กับลูกค้าเที่ยวไปก่อนที่จะรับสินค้าจากลูกค้าเที่ยวกลับเสมอ ทั้งนี้ปริมาณสินค้าที่บรรทุกไปนั้นจะต้องไม่เกินความจุของรถ และส่งสินค้าภายในกรอบเวลาที่ลูกค้าสะดวกอีกด้วย ในการศึกษานี้เราได้นำเสนอขั้นตอนวิธีการค้นหาคำตอบเลียนแบบพฤติกรรมของนกกาเหว่า ซึ่งได้รับแรงบันดาลใจมาจากพฤติกรรมการสืบพันธุ์ที่ก้าวร้าวของนกกาเหว่าเพื่อแก้ปัญหานี้ โดยเราได้นำเสนอขั้นตอนวิธีการเลือกคำตอบที่ใกล้ที่สุดด้วยวงล้อรูเล็ตต์ (roulette wheel) สำหรับการสร้างคำตอบเริ่มต้น และได้ทดสอบขั้นตอนวิธีดังกล่าวกับตัวอย่างที่ใช้เปรียบเทียบประสิทธิภาพ ผลการศึกษาพบว่า ขั้นตอนวิธีการเลือกคำตอบที่ใกล้ที่สุดด้วยวงล้อรูเล็ตต์ได้คำตอบเทียบเท่าหรือดีกว่าการค้นหาคำตอบด้วยวิธีการเลือกคำตอบที่ใกล้ที่สุดที่ปรับปรุงแล้ว นอกจากนี้เรายังได้ทำการเปรียบเทียบขั้นตอนวิธีการค้นหาคำตอบเลียนแบบพฤติกรรมของนกกาเหว่ากับวิธีอื่นๆ ที่รวบรวมมาจากงานวิจัยต่างๆ จนถึงปัจจุบัน พบว่า ในส่วนใหญ่ของปัญหาที่มีขนาดเล็กและขนาดกลาง ขั้นตอนวิธีการค้นหาคำตอบเลียนแบบพฤติกรรมของนกกาเหว่าสามารถพบคำตอบที่เทียบเท่าหรือดีกว่าคำตอบที่ดีที่สุดเท่าที่เคยพบมา ดังนั้นขั้นตอนวิธีนี้จึงเป็นอีกทางเลือกหนึ่งที่ดีในการหาคำตอบของปัญหานี้ที่มีขนาดเล็กและขนาดกลาง

คำสำคัญ: การจัดเส้นทางเดินรถ รถเที่ยวกลับ กรอบเวลา วิธีการค้นหาคำตอบจากการเลียนแบบพฤติกรรมของนกกาเหว่า

Introduction

Since business has always been a highly competitive environment, many companies employ strategies for optimizing their logistics system. To effectively improve logistic service quality, several problems have been studied including vehicle routing problem (VRP). The objective of VRP is to find an optimal set of routes for delivery vehicles which minimizes total cost while being restricted by the capacity of the vehicles. This problem is widely applied in many applications such as logistics distribution, school bus routing, and mailing system. Many types of vehicle routing problem models have been developed due to varieties of real-world situations. One of them is the vehicle routing problem with time windows (VRPTW), which is a VRP with a specified time slot that a delivery is allowed for each customer. A waiting time occurs if a vehicle arrives before the specified time window. VRPTW is commonly found in distribution planning (Wang et al., 2016), material transportation (Pradhananga et al., 2014), and E-grocery delivery (Emeç, Catay & Bozkaya, 2016). Berger & Barkaoui (2002) presented a new memetic algorithm in the serial and parallel versions to address the VRPTW. Later, they presented a new parallel hybrid genetic algorithm for VRPTW (Berger & Barkaoui, 2004). The results showed that this algorithm was highly competitive and provided some new best known solutions. Bräysy & Gendreau (2002) presented tubu search algorithm for VRPTW and concluded that this algorithm is a one of the best techniques to tackle this problem. The hybrid version which consists of ant colony optimization (ACO) and tabu search was presented by Yu et al. (2011). The results showed that this algorithm was an effective tool for VRPTW when compared with some other published meta-heuristics. The vehicle routing problem with backhauls (VRPB) is one of the interesting variations of VRP where a vehicle does not only deliver goods to the linehaul customers but also picks up goods from the backhaul customers before going back to the depot. The benefit of doing so is to utilize the unused capacity of empty vehicle on the way back to the depot after delivery. For example, a coffee company delivers the goods to its customers and picks up their raw materials back to its factory (Casco, Golden & Wasil 1988). Osman & Wassan (2002) presented a reactive tabu search which was a new way to exchange neighborhood structures for VRPB. The results showed that this algorithm was robust and competitive with other algorithms that gave the best known solutions. Brandao (2006) presented a new tabu search algorithm for the VRPB. The computational results showed that this algorithm outperformed existing published algorithms. A memetic algorithm with different local search methods was presented by Tavakkoli-Moghaddam, Saremi & Ziacc (2006). The results exposed the effectiveness of exploiting power of this algorithm. Gajpal & Abad (2009) presented multi-ant colony system which used pheromone data to generate the solutions. This algorithm gave some better solutions than the others and five new best known solutions.

In this paper, we study the VRP combining with two variations, namely backhauls and time windows. This problem is called the vehicle routing problem with backhauls and time windows (VRPBTW). Since the VRPBTW is an NP-hard combinatorial optimization problem (Thangiah, Potvin & Sun, 1996), the exact algorithm is not always possible to find an optimal solution within a limited time. For larger problems, heuristics and meta-heuristics are more appropriate than exact methods.

Bio-inspired intelligence known as meta-heuristic methods is widespread for solving various problems during the last decade. Examples of these algorithms are Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Artificial Bee Colony algorithm (ABC), Bat Algorithm (BA), and Firefly Algorithm (FA). However, only few studies have been devoted to the VRPBTW. Provin, Duhamel & Guertin (1996) presented a genetic algorithm for solving VRPBTW. The results of this algorithm showed that, on average, 1% of the optimum were produced by this algorithm. Thangiah et al. (1996) described an insertion algorithm for the VRPBTW as well as other local search heuristics to improve the initial solutions. Reimann, Doerner & Hartl (2002) presented an ant system approach which is based on the well-known insertion algorithm proposed for the VRPTW by Solomon (1987). The results showed that the learning and computational time behavior of this algorithm were equivalent to the custom-made methods. Zhong & Cole (2005) presented a basic construction of an initial infeasible solution and then used a guided local search to improve the solution. Moreover, a new technique called section planning was used to enhance the feasibility and some of the results were better than the best known solutions in the literature. Ropke & Pisinger (2006) proposed a unified heuristic for VRPB and applied the local search heuristic to enhance the solution. This algorithm obtained 227 new best known solutions out of 338 problems. Küçükoğlu & Öztürk (2015) proposed an advanced hybrid meta-heuristic algorithm which combines tabu search algorithm and simulated annealing algorithm to obtain more effective solutions for the VRPBTW. The experiment results showed that some new best known solutions were obtained and were closed to optimal solutions.

Various heuristics and meta-heuristics have been applied to VRPBTW but this is not the case for Cuckoo Search (CS). CS is a meta-heuristic method introduced by Yang & Deb (2009). Inspiration of this algorithm is the parasitic spawn behavior of some cuckoo species. This algorithm was originally designed for solving continuous problem. Although discrete versions of CS have been applied to the travelling salesman problem (Ouaarab, Ahiod & Yang, 2014) and VRP (Zheng et al., 2013), to the best of our knowledge, it had never been applied to VRPBTW. Thus, we propose CS algorithm for VRPBTW in this study.

This paper is organized as follows. Firstly, we introduce a brief concept of CS, and then describe the main steps of the algorithm. Secondly, we explain the nearest neighbor with roulette wheel selection method for generating a set of initial solutions; and the 1-move intra-route exchange and λ-interchange for improving the solutions. Then, we report the computational results. Finally, we discuss and make the conclusions for this study.

The General Concept of Cuckoo Search

A cuckoo is an extraordinary bird because of its aggressive breeding behavior. The female cuckoos lay eggs in the nest of other host birds to let them hatch and brood young cuckoo chicks. If the host birds discover that the eggs are not theirs, they can either get rid of the cuckoo eggs or abandon their nests and build new ones. However, some cuckoo species can mimic color and pattern of eggs in a few chosen host species to reduce chance of their eggs being abandoned. In addition, a cuckoo chick always mimics the call of the host chick to gain more feeding opportunity.

The cuckoo search starts by generating a number of host eggs (initial solutions) and assign them to nests. In the simplest approach, each nest can always have only a single egg. A cuckoo randomly selects a host nest and lays its egg (neighborhood search) into the nest. The aim is to replace a not-so-good solution with a new and better solution (cuckoo egg). A cuckoo egg will be abandoned and the host bird will build a completely new one (generating a new solution) when it discovers the egg is not its own. In summary, there are three ideal rules for this: (1) each cuckoo lays one egg at a time and selects a nest randomly; (2) the best nest with a high quality egg will be carried over to the next generation; (3) the number of host nests is fixed and a cuckoo egg is discovered with a probability $p_a \in [0,1]$.

Main Steps of Cuckoo Search

The steps of the CS can be described as follows:

Step 1 Generate a set of initial solutions (host eggs) by the nearest neighbor with roulette wheel selection method and assign each egg to a host nest.

Step 2 Evaluate the fitness of each solution and remember the global best solution.

Step 3 Choose randomly a host nest and then apply the neighborhood search on the host egg to generate a cuckoo egg. The host egg will be replaced with the cuckoo egg if the new cuckoo egg is better than the old one.

Step 4 Abandon the worse nest with the probability p_a and generate a new one.

Step 5 Update the global best solution if a solution has better quality than the current best one.

Step 6 If the number of iterations reaches the maximum, then the algorithm finishes. Otherwise, go to Step 3.

Initial Solution Generation for CS

The nearest neighbor heuristic (NN) is one of the classical methods for solving the VRPBTW. This method finds the solution by choosing the closest customer from the last node to be next customer in the route while preserving the capacity, time windows, and backhaul feasibilities. In general, the closeness is the reciprocal of the Euclidean distance. Küçükoğlu & Öztürk (2015) presented an improved nearest neighbor heuristic (INN), which computed the closeness from the reciprocal of the weighted sum of three factors, namely the direct distance between the two customers, the urgency of the delivery of the next customer, and the time remaining until the vehicle's last possible service start.

The INN algorithm starts a tour with the depot. Next, it adds the feasible closest unassigned customer into the tour until no more unassigned customer can be added, in which case the tour is finished and the process repeated with a new tour. If all customers are assigned, the initial solution is obtained. The closeness of customer i to customer j, denoted by $closeness_{ij}$, is computed by determining the reciprocal of $proximity_{ij}$, which is defined as: $proximity_{ij} = \alpha c_{ij} + \beta h_{ij} + \gamma v_{ij}$, where $\alpha + \beta + \gamma = 1$, $\alpha, \beta, \gamma \geq 0$, c_{ij} denotes the distance expressed as time from customer i to customer j, h_{ij} denotes the idle time before servicing customer j after customer i, and v_{ij} denotes the urgency of delivery to customer j after customer i expressed as the time remaining until the vehicle's last possible service start for customer j.

In this paper, we propose the nearest neighbor with roulette wheel selection method (NNRW) which is a combination of a roulette wheel selection method (Holland, 1975) and the improved nearest neighbor (INN) heuristic (Küçükoğlu & Öztürk, 2015) for generating the initial solutions. The $closeness_{ij}$ which is the reciprocal of $proximity_{ij}$ is defined the same way the INN heuristic describes. The NNRW method can be explained as follows.

During a tour construction where customer i is our current customer, let p_j be the selection probability of customer j to be served next after customer i. Let U be the set of all unassigned customers. Then p_j is calculated by:

$$p_j = \frac{closeness_{ij}}{\sum_{h \in U} closeness_{ih}} \text{ for } j \in U$$

We define $q_j = \sum_{h=1}^{j} p_h$ for $j \in U$. Then a random number r which ranges between 0 and 1 is selected for spinning the roulette wheel. If $r \le q_1$, then choose the first customer in U to be the next customer for the vehicle. Otherwise, if $q_{j-1} < r \le q_j$, then choose the j^{th} customer in U to be the next customer where $2 \le j \le |U|$. The assigned customers are discarded from U to prevent duplicate customers in a tour.

The initial solution construction always starts a tour with the depot, and then finds the next customer by the nearest neighbor with roulette wheel selection method. If the next customer violates the constraints (the capacity constraints, the time windows constraints, and the backhaul constraints), we spin the roulette wheel again to find a new one. If the new one is still not feasible, we end this tour and begin a new tour. This process is repeated until all customers are served.

Neighborhood Search

The definition of a neighborhood of a solution in a continuous problem is well known, but this is not always the case for a combinatorial problem. In VRPBTW, a neighbor of a solution is generated by changing the order of visited customers. In this study, this can be accomplished by the 1-move intra-route exchange (Chiang & Russell, 1997) and the λ-interchange (Osman, 1993).

The idea of 1-move intra-route exchange is randomly removed one customer (linehaul or backhaul) from a route and inserted back to the same route in a different position. The solution is accepted if it can reduce the total cost while the capacity constraints, the time windows constraints, and the backhaul constraints are not violated. An example of 1-move is shown in Figure 1.

The λ-interchange is a technique which combines many methods such as insertion, swap, insert section, and swap section. The idea of λ-interchange is to interchange customers (linehauls or backhauls) between routes where λ is a limit on the number of customers to be exchanged. The operator (λ_1, λ_2) on routes (p, q) means exchanging λ_1 customers on route p with λ_2 customers on route q, where $\lambda_1, \lambda_2 \le \lambda$. The improved solution is accepted if the total cost is decreased while maintaining the capacity, time windows, and backhaul feasibility. An example of operator (1, 0) which removes customer 4 in the first route and then adds it in another route is given in Figure 2. This operator is similar to the insertion algorithm. As shown in Figure 3, the operator (1, 2) exchanges customer 4 in the first route with customer 8 and customer 9 in the second route. This operator is similar to the swap section algorithm.

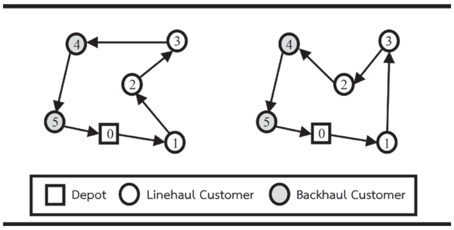

Figure 1 An Example of a 1-move

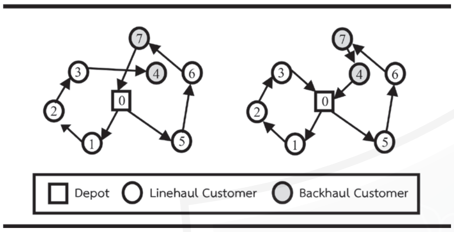

Figure 2 Example of operator (1, 0)

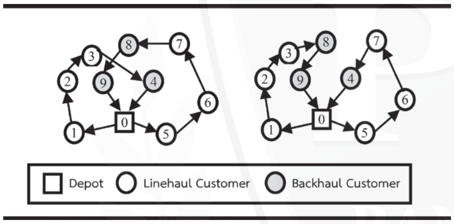

Figure 3 Example of operator (1, 2)

Computational Results

The proposed algorithm was coded in Microsoft Visual C# 2010 Express and executed on a PC with 2.5 GHz Intel Core2 Duo CPU and 4 GB memory. As for this experiment, the algorithm parameters were assigned as follows: α = 0.4, β = 0.3, γ = 0.3 (Küçükoğlu & Öztürk, 2015: 60-68), the number of host nest = 15, p_a = 0.25 (Yang and Deb, 2009: 210-214), the size of λ-interchange operator = 4, maximum number of iterations = 300.

We tested NNRW algorithms on the benchmark problems sets (R101-R105) developed by Gelinas et al. (1995) for the VRPBTW. For each problem, 100 customers are located uniformly over the service area with a short scheduling horizon. The small and medium problems are obtained by taking the first 25 and 50 customers respectively. Moreover, for each problem size, three problems are generated by randomly selecting 10%, 30% and 50% of the nodes to be backhaul customers without changing other attributes. The results are shown in Table 1.

In Table 1, the first column represents the number of customers in the problem, name of problems are shown in the second column, BH (%) denotes the percentage of backhauls, Dist shows the total distance of solution, NV indicates the number of vehicles used in the solution. The Average Dist and SD columns indicate the average and the standard deviation calculated from 10 independent runs of NNRW. The best solutions of NNRW algorithm from these runs are represented by Best Dist, and the computational time in seconds is presented in the CPU time column. The $\%Gap_{imp}$ is computed by the following formula:

$$\%Gap_{imp} = \frac{(\text{NNRW } \textit{solution}) - (\textit{INN solution})}{\textit{INN solution}} \times 100$$

$\%Gap_{imp}$ represents the quality of the NNRW solutions in terms of improvement percentage over the INN solution, where a negative value indicates that NNRW solution is better than the INN solution, zero value indicates that NNRW solution is equal to the INN solution, and a positive value indicates the NNRW solution is worse than the INN solution.

Table 1 Comparison of the NNRW solutions with NN and INN for VRPBTW

Size	Prob	BH (%)	Nearest Neighbor Solutions (NN)			Improved Nearest Neighbor Solutions (INN)			Nearest Neighbor with Roulette Wheel Selection Solutions (NNRW)					$\%Gap_{imp}$
			Dist	NV	CPU time	Dist	NV	CPU Time	Average Dist	SD	Best Dist	NV	CPU time	
n=25	R101	10	662.1	10	0.18	662.1	10	0.21	666.70	16.07	**643.4**	9	0.25	**-2.82**
		30	735.3	10	0.33	721.8	10	0.32	740.23	8.29	**721.8**	10	0.26	**0.00**
		50	693.1	11	0.21	678.8	10	0.19	689.46	20.75	**676.8**	10	0.29	**-0.29**
	R102	10	564.2	7	0.18	563.5	7	0.20	576.28	28.02	**563.5**	7	0.31	**0.00**
		30	629.6	10	0.57	628.1	9	0.90	630.77	2.63	**628.1**	9	0.45	**0.00**
		50	591.6	8	0.18	586.4	8	0.22	596.28	10.67	**584.4**	8	0.35	**-0.34**
	R103	10	507.1	6	0.20	488.8	6	0.22	508.13	9.90	**488.8**	6	0.27	**0.00**
		30	534.8	6	0.16	534.0	7	0.14	538.85	21.00	**514.8**	7	0.25	**-3.60**
		50	535.2	7	0.19	497.4	6	0.22	506.58	19.23	**490.6**	6	0.29	**-1.37**
	R104	10	486.2	5	0.22	465.5	5	0.23	471.91	10.20	**453.4**	5	0.27	**-2.60**
		30	517.4	6	0.16	513.3	6	0.14	504.77	16.39	**476.3**	6	0.22	**-7.21**
		50	506.5	5	0.17	500.5	5	0.14	487.69	20.28	**465.4**	5	0.19	**-7.01**
	R105	10	579.6	7	0.19	565.1	7	0.20	585.34	22.45	**565.1**	7	0.24	**0.00**
		30	633.4	8	0.21	632.9	8	0.20	642.08	7.34	**632.9**	8	0.26	**0.00**
		50	639.2	8	0.15	635.5	9	0.18	633.06	18.24	**591.1**	8	0.20	**-6.99**
n=50	R101	10	1175.5	16	0.19	1173.2	15	0.23	1156.30	18.27	**1134.0**	15	0.43	**-3.34**
		30	1223.2	16	0.22	1218.8	16	0.29	1233.47	16.85	**1215.0**	16	0.39	**-0.31**
		50	1203.1	16	0.25	1190.5	16	0.27	1199.93	8.52	**1183.9**	16	0.38	**-0.55**
	R102	10	994.3	12	0.34	987.8	12	0.43	1010.6	19.93	**977.0**	12	0.52	**-1.09**
		30	1091.1	14	0.22	1081.2	14	0.28	1079.18	23.00	**1054.7**	14	0.34	**-2.45**
		50	1100.9	14	0.18	1100.3	14	0.22	1086.25	20.86	**1060.9**	14	0.32	**-3.58**
	R103	10	877.8	10	0.19	874.9	10	0.37	860.93	20.69	**833.7**	10	0.49	**-4.71**
		30	955.3	12	0.23	951.7	12	0.22	938.85	21.69	**894.4**	11	0.38	**-6.02**
		50	947.6	11	0.3	939.1	11	0.34	931.99	26.38	**896.4**	10	0.44	**-4.55**
	R104	10	792.1	7	0.24	784.8	8	0.25	739.75	22.20	**704.3**	7	0.38	**-10.26**
		30	795.8	8	0.31	785.6	7	0.35	791.10	24.46	**745.9**	8	0.48	**-5.05**
		50	771.7	8	0.35	771.6	8	0.46	788.95	14.37	**767.0**	8	0.50	**-0.60**
	R105	10	1091.5	12	0.17	1091.5	13	0.24	1030.63	28.48	**983.3**	12	0.38	**-9.91**
		30	1084.3	13	0.21	1075.6	14	0.26	1077.62	23.40	**1053.2**	13	0.34	**-2.08**
		50	1078.4	12	0.26	1059.2	12	0.33	1065.59	22.75	**1026.3**	12	0.42	**-3.11**
n=100	R101	10	2072.7	28	0.58	1914.5	25	0.56	1859.44	28.33	**1811.6**	24	1.21	**-5.37**
		30	2091.2	26	0.95	1978.7	25	0.86	1937.07	26.95	**1898.8**	24	1.04	**-4.04**
		50	1992.0	26	2.15	1990.2	27	4.52	1973.86	26.72	**1944.1**	26	3.54	**-2.32**
	R102	10	1687.8	22	0.85	1671.8	21	1.00	1689.72	35.40	**1628.8**	21	1.14	**-2.57**
		30	1755.7	23	1.05	1733.7	22	1.20	1743.87	20.38	**1716.2**	23	1.45	**-1.01**
		50	2001.8	26	0.55	1891.2	25	0.51	1800.87	23.71	**1756.2**	22	0.97	**-7.14**
	R103	10	1457.4	19	0.54	1454.2	19	0.59	1424.71	15.74	**1399.8**	18	1.02	**-3.74**
		30	1478.8	18	1.03	1459.0	17	1.73	1467.99	23.11	**1439.2**	17	2.07	**-1.36**
		50	1563.5	20	1.11	1519.5	19	1.27	1535.40	12.92	**1514.2**	19	1.78	**-0.35**
	R104	10	1206.3	14	1.30	1152.3	13	1.44	1214.11	37.57	**1148.1**	13	2.31	**-0.36**
		30	1210.7	14	2.29	1201.7	14	5.62	1234.33	24.28	**1196.4**	14	4.21	**-0.44**
		50	1274.8	14	0.96	1274.7	15	1.17	1289.21	28.33	**1244.7**	14	2.22	**-2.35**
	R105	10	1632.1	19	0.48	1627.6	20	0.56	1609.73	33.76	**1557.2**	18	1.19	**-4.33**
		30	1626.3	20	0.75	1621.7	19	1.40	1684.87	36.55	**1612.3**	19	1.68	**-0.58**
		50	1724.2	21	0.99	1699.8	21	1.37	1720.85	44.86	**1683.4**	19	1.83	**-0.96**

In Table 1, the $\%Gap_{imp}$ column shows that the NNRW solutions are better or equal to INN solutions for all instances. The remarkable improvements (more than 10%) can be seen in R104 for 50 customers with 10% backhauls. Although the NNRW method used more execution time than INN algorithm for some instances, these results indicate that NNRW algorithm was more effective than INN heuristic in terms of solution quality.

To evaluate the efficiency of CS, we compared the CS solutions with the best known solutions collected from many papers in various instances as presented in Table 2. The other collected algorithms were Push-Forward Insertion Heuristic (PFIH) (Thangiah et al., 1996), Genetic Algorithm (GA) (Potvin et al., 1996), Hybrid Meta-heuristic Algorithm (HMA) (Küçükoğlu & Öztürk, 2015), and Unified Heuristic (UH) (Ropke & Pisinger, 2006). The numbers with bold face font in each row indicates the best known solution for that problem, and the $\%Gap_{best}$ in the last column is calculated by the following formula:

$$\%Gap_{best} = \frac{(\text{CS solution}) - (\text{the best known solution})}{\text{the best known solution}} \times 100.$$

where a positive value indicates that our solution is worse than the best known solution, zero value indicates that CS solution is equal to the best known solution, and a negative value indicates our proposed algorithm can find a new best known solution.

For small problems with 25 customers, the proposed algorithm obtained 12 solutions that were equal or better than the best known solutions out of 15 instances. The new best known solution was found in the R101 problem with 50% backhauls. From Table 2, CS performed better than PFIH, GA, and HMA in terms of number of best case solutions.

For medium problems with 50 customers, our algorithm obtained 2 matching best known solutions and 5 new best known solutions out of 15 problems, namely, the R101 problem with 10% backhauls, the R102 problem with 10%, the R104 problem with 50%, the R105 problem with 10% and 50% backhauls. According to Table 2, CS still outperformed PFIH, GA, and HMA in terms of number of best case solutions.

For large problems with 100 customers, the proposed method underperformed the other methods in terms of best known solutions except for two cases, namely, the R101 problem with 10% and 30% backhauls. Although CS underperformed GA and HMA, it performed better than PFIH while comparable with UH in terms of number of best case solutions.

Table 2 Comparison of the CS solutions with other algorithms for VRPBTW

Size	Prob	BH (%)	CS		PFIH		GA		HMA		UH		$\%GAP_{best}$
			Dist	NV	Dist	NV	Dist	NV	Dist	NV	Dist	NV	
n=25	R101	10	**643.4**	**9**	681.7	9	**643.4**	**9**	**643.4**	**9**	-	-	**0.00**
		30	721.8	10	**716.5**	**9**	721.8	10	721.8	10	-	-	0.74
		50	**676.8**	**10**	700.6	9	682.3	10	**676.8**	**10**	-	-	**0.00**
	R102	10	**563.5**	**7**	565.1	7	**563.5**	**7**	**563.5**	**7**	-	-	**0.00**
		30	**628.1**	**9**	629.3	9	622.3	9	**628.1**	**9**	-	-	**0.00**
		50	**584.4**	**8**	585.4	7	**584.4**	**8**	**584.4**	**8**	-	-	**0.00**
	R103	10	478.8	6	496.2	6	**476.6**	**6**	478.8	6	-	-	0.46
		30	**507.0**	**7**	520.4	6	**507.0**	**7**	**507.0**	**7**	-	-	**0.00**
		50	**483.0**	**6**	480.4	6	**483.0**	**6**	**483.0**	**6**	-	-	**0.00**
	R104	10	**452.8**	**5**	463.1	5	**452.8**	**5**	453.8	5	-	-	**0.00**
		30	473.1	6	470.1	6	**468.5**	**6**	**468.5**	**6**	-	-	0.98
		50	**446.8**	**5**	447.8	5	**446.8**	**5**	**446.8**	**5**	-	-	**0.00**
	R105	10	**565.1**	**7**	591.7	7	**565.1**	**7**	**565.1**	**7**	-	-	**0.00**
		30	**623.5**	**8**	630.6	7	630.2	8	**623.5**	**8**	-	-	**0.00**
		50	**591.1**	**8**	592.9	7	592.1	7	592.1	7	-	-	**-0.17**
n=50	R101	10	**1133.3**	**15**	1160.3	13	1138.1	14	1135.8	15	-	-	**-0.22**
		30	**1191.6**	**16**	1224.6	15	1192.7	16	**1191.6**	**16**	-	-	**0.00**
		50	1183.9	16	**1175.6**	**16**	1183.9	16	1183.9	16	-	-	0.71
	R102	10	**976.5**	**12**	978.8	12	976.8	12	976.8	12	-	-	**-0.03**
		30	1054.6	14	1034.9	14	**1029.2**	**13**	1046.0	14	-	-	2.47
		50	**1059.7**	**14**	1061.6	14	**1059.7**	**14**	1061.6	14	-	-	**0.00**
	R103	10	818.8	9	844.3	10	**813.3**	**9**	815.5	9	-	-	0.68
		30	894.4	11	917.8	11	**892.7**	**10**	889.3	11	-	-	0.19
		50	889.0	10	903.4	10	**885.5**	**10**	887.7	10	-	-	3.92
	R104	10	698.2	7	691.4	7	689.2	6	**687.7**	**7**	-	-	1.53
		30	742.3	8	743.8	8	751.5	7	**736.8**	**8**	-	-	0.75
		50	**734.5**	**8**	765.6	7	741.4	7	738.2	8	-	-	**-0.50**
	R105	10	**972.8**	**11**	996.2	11	1002.5	10	978.5	11	-	-	**-0.58**
		30	1027.1	13	1060.5	11	1047.8	11	**1026.7**	**12**	-	-	0.04
		50	**993.4**	**11**	1028.6	11	1018.0	11	996.2	11	-	-	**-0.28**
n=100	R101	10	**1805.7**	**24**	1842.3	24	1815.0	23	1811.6	23	1818.9	22	**-0.33**
		30	**1886.9**	**24**	1928.6	24	1896.6	23	1891.1	24	1959.6	23	**-0.22**
		50	1924.3	25	1937.6	25	**1905.9**	**24**	1911.2	25	1939.1	24	0.96
	R102	10	1624.1	20	1654.1	20	**1622.9**	**20**	1623.7	20	1653.2	19	0.07
		30	1705.6	22	1764.3	21	**1688.1**	**20**	1724.0	22	1750.7	22	1.04
		50	1757.8	22	1745.7	21	**1735.7**	**21**	1759.8	23	1775.8	22	1.27
	R103	10	1379.7	17	1371.6	15	**1343.7**	**16**	1346.9	16	1387.6	15	2.68
		30	1407.7	16	1477.6	16	**1381.6**	**15**	1385.9	16	1390.3	15	1.89
		50	1474.7	19	1543.2	17	1456.6	17	1465.0	18	**1456.5**	**17**	1.24
	R104	10	1145.2	13	1220.3	13	1117.7	12	1093.4	12	**1084.2**	**11**	5.63
		30	1167.8	14	1303.5	12	1169.1	12	**1136.6**	**12**	1154.8	11	2.74
		50	1197.3	14	1346.6	13	1203.7	13	**1189.6**	**13**	1191.4	11	0.65
	R105	10	1523.7	18	1553.4	17	1621.0	17	**1516.0**	**17**	1561.3	15	0.51
		30	1602.2	19	1643.0	18	1652.8	16	**1581.5**	**17**	1583.3	16	1.31
		50	1629.6	19	1657.4	18	1706.7	18	**1604.1**	**18**	1710.2	16	1.59

Result Discussion

When comparing the results in terms of the number of best case solutions, the CS algorithm is competitive with the other methods in literature for solving small and medium size VRPBTW problems. However, for some instances CS underperformed the existing algorithms especially GA (Potvin, Duhamel & Guertin, 1996) and HMA (Küçükoğlu & Öztürk, 2015). We speculated that there are two main reasons for this. First, the CS algorithm generates only initial 15 solutions for all instances while the GA (Potvin, Duhamel & Guertin, 1996) produces 100 initial solutions for small and medium size problem and 200 initial solutions for large size problem. Therefore, the GA can explore more in the solution space and get the better solutions than the CS algorithm. Second, the HMA (Küçükoğlu & Öztürk, 2015) is a hybrid meta-heuristic which is combined with tabu search, that prevents the search from cycling back to previously visited solutions, and simulated annealing algorithm, that prevents from trapping in the local optimum while the CS algorithm does not have those strategies. This is one of the advantages of hybrid algorithm.

Conclusions

In this paper, we present a cuckoo search (CS) algorithm to solve the VRPBTW problem. In the solution construction part, we use the nearest neighbor with roulette wheel selection method (NNRW) for generating a set of initial solutions. The solutions are iteratively improved within the CS framework by the neighborhood search algorithms, namely the 1-move intra-route exchange and the λ-interchange. The NNRW algorithm is compared with the general nearest neighbor algorithm (NN) and the improved nearest neighbor algorithm (INN) through the benchmark instances. The results show that NNRW is superior to NN and INN heuristic in terms of solution quality. In addition, CS algorithm was compared with other methods, namely Push-Forward Insertion Heuristic (PFIH), Genetic Algorithm (GA), Hybrid Meta-heuristic Algorithm (HMA), and Unified Heuristic (UH). The results showed that the proposed algorithm was able to give best known solutions or found the new best known solutions for some instances, especially problems with small and medium sizes. Hence, it is a competitive method for solving small and medium size VRPBTW problems. Further research can be done to enhance CS algorithm by combining with other heuristics. Hybrid heuristics can make the CS algorithm approach more effective for VRPBTW.

References

Berger, J. & Barkaoui, M. (2002). A memetic algorithm for the vehicle routing problem with time windows. In *The 7th International Command and Control Research and Technology Symposium*.

Berger, J. & Barkaoui, M. (2004). A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. *Computers & operations research, 31*(12), 2037-2053.

Brandao, J. (2006). A new tabu search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*, *173*(2), 540-555.

Bräysy, O. & Gendreau, M. (2002). Tabu search heuristics for the vehicle routing problem with time windows. *Top*, *10*(2), 211-237.

Casco, D., Golden, B. & Wasil, E. (1988). *Vehicle routing with backhauls: Models, algorithms and case studies. Vehicle Routing: Methods and Studies. Studies in management science and systems-Volume 16.* Publication of Dalctraf.

Chiang, W. C. & Russell, R. A. (1997). A reactive tabu search metaheuristic for the vehicle routing problem with time windows. *INFORMS Journal on computing*, *9*(4), 417-430.

Emeç, U., Çatay, B. & Bozkaya, B. (2016). An adaptive large neighborhood search for an e-grocery delivery routing problem. *Computers & Operations Research*, *69*, 109-125.

Gajpal, Y. & Abad, P. L. (2009). Multi-ant colony system (MACS) for a vehicle routing problem with backhauls. *European Journal of Operational Research*, *196*(1), 102-117.

Gelinas, S., Desrochers, M., Desrosiers, J. & Solomon, M. M. (1995). A new branching strategy for time constrained routing problems with application to backhauling. *Annals of Operations Research*, *61*(1), 91-109.

Holland, J. H. (1975). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. U Michigan Press.

Küçükoğlu, İ. & Öztürk, N. (2015). An advanced hybrid meta-heuristic algorithm for the vehicle routing problem with backhauls and time windows. *Computers & Industrial Engineering*, *86*, 60-68.

Osman, I. H. & Wassan, N. A. (2002). A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. *Journal of Scheduling*, *5*(4), 263-285.

Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of operations research*, *41*(4), 421-451.

Ouaarab, A., Ahiod, B. & Yang, X. S. (2014). Discrete cuckoo search algorithm for the travelling salesman problem. *Neural Computing and Applications*, *24*(7-8), 1659-1669.

Potvin, J. Y., Duhamel, C. & Guertin, F. (1996). A genetic algorithm for vehicle routing with backhauling. *Applied Intelligence*, *6(4)*, 345-355.

Pradhananga, R., Taniguchi, E., Yamada, T. & Qureshi, A. G. (2014). Environmental Analysis of Pareto Optimal Routes in Hazardous Material Transportation. *Procedia-Social and Behavioral Sciences*, *125*, 506-517.

Reimann, M., Doerner, K. & Hartl, R. F. (2002). Insertion based ants for vehicle routing problems with backhauls and time windows. In *International Workshop on Ant Algorithms* (pp. 135-148). Springer Berlin Heidelberg.

Ropke, S. & Pisinger, D. (2006). A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, *171(3)*, 750-775.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research*, *35(2)*, 254-265.

Tavakkoli-Moghaddam, R., Saremi, A. R. & Ziaee, M. S. (2006). A memetic algorithm for a vehicle routing problem with backhauls. *Applied Mathematics and Computation, 181*(2), 1049-1060.

Thangiah, S. R., Potvin, J. Y. & Sun, T. (1996). Heuristic approaches to vehicle routing with backhauls and time windows. *Computers & Operations Research*, *23(11)*, 1043-1057.

Wang, X., Wang, M., Ruan, J. & Zhan, H. (2016). The Multi-objective Optimization for Perishable Food Distribution Route Considering Temporal-spatial Distance. *Procedia Computer Science*, *96*, 1211-1220.

Yang, X. S. & Deb, S. (2009). Cuckoo search via Lévy flights. In *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on* (pp. 210-214). IEEE.

Yu, B., Yang, Z. Z. & Yao, B. Z. (2011). A hybrid algorithm for vehicle routing problem with time windows. *Expert Systems with Applications*, *38*(1), 435-441.

Zheng, H., Zhou, Y., Xie, J. & Luo, Q. (2013). A hybrid Cuckoo Search Algorithm-GRASP for Vehicle Routing Problem. *Journal of Convergence Information Technology*, *8(3)*, 821-828.

Zhong, Y. & Cole, M. H. (2005). A vehicle routing problem with backhauls and time windows: a guided local search solution. *Transportation Research Part E: Logistics and Transportation Review*, *41*(2), 131-144.

Name and Surname: Tanawat Worawattawechai
Highest Education: Master of Science Degree in Applied Mathematics and Computational Science, Chulalongkorn University
University or Agency: Chulalongkorn University
Field of Expertise: Optimization
Address: Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University 10330

Name and Surname: Boonyarit Intiyot
Highest Education: Ph.D. in Industrial and Systems Engineering, Virginia Polytechnic Institute and State University
University or Agency: Chulalongkorn University
Field of Expertise: Scheduling problems, Maintenance models, Optimization
Address: Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University 10330

Name and Surname: Chawalit Jeenanunta
Highest Education: Ph.D. in Industrial and Systems Engineering, Virginia Polytechnic Institute and State University
University or Agency: Thammasat University
Field of Expertise: Linear programming, Integer programming, Network optimization, Simulation, Supply chain management
Address: School of Management Technology, Sirindhorn International Institute of Technology (SIIT), Thammasat University 12120