International Journal of Innovative Computing, Information and Control
Volume **17**, Number **6**, December **2021**

ICIC International ©2021 ISSN 1349-4198
pp. 1921–1935

OPTIMAL SOLVING MULTI-VEHICLE ROUTING PROBLEMS VIA PARALLEL CUCKOO SEARCH

Supaporn Suwannarongsri

Department of Materials Handling and Logistics Engineering
Faculty of Engineering
King Mongkut's University of Technology North Bangkok
1518 Pracharaj-1 Road, Bangsue, Bangkok 10800, Thailand
supaporn.s@eng.kmutnb.ac.th

Received June 2021; revised September 2021

Abstract. *This paper presents the application of the parallel cuckoo search (PCS) to optimally solve the multi-vehicle routing problems (MVRP). As one of the latest enhanced versions of the original cuckoo search (CS) and one of the most efficient metaheuristic optimization search techniques, the PCS contains multiple CSs as the search units for running on a single-CPU platform. Based on the random process drawn from the Lévy distribution, the PCS possesses the partitioning strategy (PS) to divide an entire search space into many sub-search spaces for each CS, the sequencing strategy (SS) to arrange the search units to run one-by-one on each generation and the discarding strategy (DS) to terminate some unlikely to be successful CSs. In this paper, the PSC is applied to solving the MVRP based on the modern optimization approach. Ten real-world MVRP consisting of approximately 100-500 locations are selected and solved by the PCS. Results obtained by the PCS will be compared with those obtained by the original CS. From experimental results, it was found that the PCS can provide optimal solutions of all ten selected MVRP with shorter total distance than the original CS, significantly.*

Keywords: Multi-vehicle routing problems, Parallel cuckoo search, Modern optimization, Metaheuristic optimization technique

1. Introduction.

The multi-vehicle routing problem (MVRP) is one of the well-known logistic problems based on the original vehicle routing problem (VRP) formally introduced in 1959 by Dantzig and Ramser [1]. Based on the graph theory, the VRP is generally defined by a graph $G = (V, \varepsilon, C)$, where $V = (v_0, \ldots, v_n)$ is the set of vertices representing locations, $\varepsilon = \{(v_i, v_j)|(v_i, v_j) \in V^2, i \neq j\}$ is the arc set representing distances and $C = \{C_{ij}|(v_i, v_j) \in \varepsilon\}$ is the cost matrix defined over ε representing traveling times or traveling costs. Both VRP and MVRP can be considered as a class of combinatorial optimization problems and also the NP-hard problems [2-4]. The main objective of the VRP and MTSP is to minimize total distance in order to minimize the overall costs and to maximize the customers' demand. The MVRP, consisting of a fleet of vehicles, expands the VRP to include different service requirements (pickup and/or delivery of products) at each location, different capacities and time constraints of each vehicle in the fleet [5-7]. In the MVRP, all vehicles leave the depot, serve customers along the routes and return to the depot after completion of their routes. In addition, each location will be visited exactly once by any vehicle in a fleet. In case of a single depot, the MVRP consisting of 24 locations (including the depot) with 4 vehicles in the fleet can be visualized by Figure 1, where ◯ stands for the service locations and ■ stands for the depot.

DOI: 10.24507/ijicic.17.06.1921

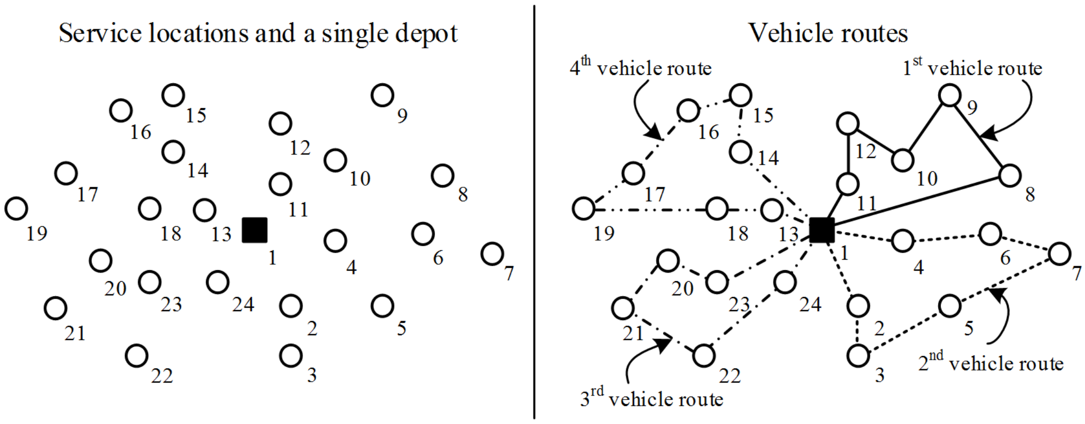


Figure 1. The MVRP with a single depot, 24 locations and 4 vehicles

Many variants of VRP and MVRP are studied and expanded to address the variety of conditions in real-world applications, for example, the capacitated VRP (CVRP), the VRP with time windows (VRPTW), the heterogeneous fleet VRP (HVRP), the VRP with pickup and delivery (VRPPD) and multiple depots VRP (MDVRP) [8,9]. Following the literature, the MVRP can be successfully solved by the efficient metaheuristic optimization search techniques, such as neural network (NN) [10], particle swarm optimization (PSO) [9,11-15], genetic algorithm (GA) [16,17], tabu search (TS) [18], ant colony optimization (ACO) [19,20], evolutionary game analysis (EGA) [21], cuckoo search (CS) [22-24], bee colony optimization (BCO) [25] and flower pollination algorithm (FPA) [26,27]. In addition, the CVRP can be optimally solved by the parallel hybrid CS, central force optimization (CFO), chemical reaction optimization (CRO) and 3-Opt local search called the HCSCROCFO-3Opt algorithm [28].

In 2021, the parallel cuckoo search (PCS) was firstly proposed as one of the most efficient metaheuristic optimization search techniques for numerical function optimization and continuous problems [29]. The PCS is one of the latest enhanced versions of the original cuckoo search (CS) proposed by Yang and Deb in 2009 [30,31] which is based on the random process drawn from the Lévy-flight distribution. The PCS algorithm was proposed for running on a single-CPU platform. It contains multiple CSs as the search units. In addition, the PCS possesses the partitioning strategy (PS) to divide an entire search space into many sub-search spaces for each CS, the sequencing strategy (SS) to arrange the search units to run one-by-one on each generation and the discarding strategy (DS) to terminate some unlikely to be successful CSs. The well performed search performance of the PCS was proven by tests against several benchmark optimization problems [29]. In this paper, the PCS is applied to optimally solving the MVRP. Ten selected real-world MVRP consisting of approximately 100-500 locations are selected and solved by the PCS. Results obtained by the PCS will be compared with those obtained by the original CS.

This paper consists of five sections. After an introduction presented in Section 1, the problem formulation including the MVRP model, objective function and constrained functions is illustrated in Section 2. Section 3 describes the PCS algorithm and the PCS-based MVRP optimization. In Section 4, experimental results and discussions are illustrated. Finally, Section 5 gives the conclusions and future research.

2. **Problem Formulation.** In this section, the MVRP model is described as the problem formulation. In general form, the objective of the MTSP is to minimize total distance

beyond all routes in such a way that the total routing cost is minimized, the number of vehicles is minimized and the fulfilled demand is maximized. The MVRP model can be formulated based on the graph theory [1-4,9,32]. In case of a single depot, there are n locations and m vehicles in a fleet. The distance between the ith and the jth locations is represented by d_{ij}. In the symmetric case, $d_{ij} = d_{ji}$, for all locations (i, j). They can be displayed by the distance matrix d: $n \times n \to \Re$ between the locations. All vehicles will start at the depot. They will take a route such that each location except the depot is visited by exactly one vehicle. Finally, all vehicles will return to the depot at the end of the tour. Let the decision variables $\delta_{i,j,k}$ be 1 if and only if the vehicle k travels from the ith location to the jth location, and 0 otherwise. Also, let $T_{i,j,k}$ be the traveling time of the vehicle k from the ith location to the jth location. $T_{i,j,k}$ can be calculated by the relation between the average vehicle's speed and the working time, and $T_{\max}$ is the maximum working time of each vehicle. The objective function $f(\cdot)$ of the MVRP is stated in (1) to minimize the total traveling distances. Generally, the MVRP optimization possesses the following constrained functions as stated in (2)-(8).

$$\textbf{Minimize} \quad f(\cdot) = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{m} \delta_{i,j,k} d(i,j) \tag{1}$$

$$\text{Subject to} \quad \sum_{j=2}^{n} \delta_{1,j,k} = 1, \quad \forall 1 \le k \le m \tag{2}$$

$$\sum_{i=2}^{n} \delta_{i,1,k} = 1, \quad \forall 1 \le k \le m \tag{3}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{m} \delta_{i,j,k} = 1, \quad \forall 2 \le i \le n \tag{4}$$

$$\sum_{i=1}^{n}\sum_{k=1}^{m} \delta_{i,j,k} = 1, \quad \forall 2 \le j \le n \tag{5}$$

$$\sum_{i=1}^{n} \delta_{i,r,k} = \sum_{j=1}^{n} \delta_{r,j,k}, \quad \forall 2 \le r \le n, \ \forall 1 \le k \le m \tag{6}$$

$$u_i - u_j + (n - m) \cdot \sum_{k=1}^{m} \delta_{i,r,k} \le n - m - 1, \quad \forall 2 \le i \ne j \le n \tag{7}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} T_{i,j,k} \le T_{\max}, \quad 1 \le k \le m \tag{8}$$

The constrained function in (2) ensures that all vehicles will leave the depot exactly once. The constrained function in (3) ensures that all vehicles will return to the depot exactly once. The constrained function in (4) ensures that all locations (except the depot) will be left by only one vehicle exactly once. The constrained function in (5) ensures that all locations (except the depot) will be arrived by only one vehicle exactly once. The constrained function in (6) ensures that the amount of time that all vehicles spend for visiting all locations equals the amount of time that all locations are left. The constrained function in (7) ensures that no sub-tours exist (degenerate routes that do not include the depot), by using $n-1$ as dummy variables of $u_2, \dots, u_n$. Finally, the constrained function in (8) ensures that each vehicle spends the working time within its defined maximum working time, where $T_{i,j,k}$ is the traveling time of the vehicle k from the ith location to the jth location and $T_{\max}$ is the maximum working time of each vehicle. The objective function of the MVRP in (1) will be minimized in order to satisfy the constrained functions in (2)-(8) by searching for the optimal routes for all vehicles in a fleet.

3. PCS Algorithm for MVRP Optimization.

The PCS algorithm is briefly described in this section. Then, the PCS-based MVRP optimization is elaborately illustrated.

3.1. PCS algorithm.

Based on the original cuckoo search (CS), the CS algorithm is represented by the pseudo code as shown in Figure 2 [30,31,33]. New solutions $\boldsymbol{x}^{(t+1)}$ of cuckoo i can be calculated by using the mathematical relations as stated in (9)-(13), where α is the step size, $\oplus$ is entrywise multiplication, Lévy(λ) is randomly drawn from Lévy-flight distribution, s is the step length, u and v are randomly drawn from normal distribution, σ_u and σ_v are the standard deviations of u and v, and Γ is the Gamma function.

$$\boldsymbol{x}_i^{(t+1)} = \boldsymbol{x}_i^{(t)} + \alpha \oplus \text{Lévy}(\lambda) \tag{9}$$

$$\text{Lévy} \approx u = t^{-\lambda}, \quad (1 < \lambda \leq 3) \tag{10}$$

$$s = \frac{u}{|v|^{1/\beta}} \tag{11}$$

$$u \approx N\left(0, \sigma_u^2\right), \quad v \approx N\left(0, \sigma_v^2\right) \tag{12}$$

$$\sigma_u = \sqrt[\beta]{\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma[(1+\beta)/2]\beta 2^{(\beta-1)/2}}}, \quad \sigma_v = 1 \tag{13}$$

```
Objective function f(x), x = (x_1,...,x_d)^T
Generate initial population of cuckoos n and solution x_i , i = 1, 2,..., n
Initialize Max_Gen, Gen = 1 and search spaces
for (Gen < Max_Gen)
    - Get n new nests randomly for n cuckoos by Lévy-flight distribution
    - Evaluate their quality via f(x_i)
    - Choose a nest among n (say j) randomly
   if {f(x_j)<f(x_i)},
       - Replace the solution x_i by the new solution x_j
   end if
   - A fraction (p_a) of worse nests are abandoned and new ones are built
   - Keep the best solutions (or nests with quality solutions)
   - Rank the solutions and find the current best
   - Update Gen = Gen + 1
end for
Report the best solution found
```

FIGURE 2. Pseudo code of CS algorithm

The PCS algorithm [29] consists of multiple CSs served as the search unit. Also, the PCS possesses the PS to divide an entire search space into many sub-search spaces for each CS, the SS to arrange the search units to run one-by-one on each generation and the DS to terminate some unlikely to be successful CSs. The algorithm of the PCS will identify entire search spaces, define number of search units, initialize N search-unit algorithms (CSs) and define the maximum generation (Max_Gen) as the termination criteria (TC) and a counter $Gen = 1$. Then, the algorithm will activate the PS, situate CSs to sub-search-spaces and remove partitions such that all CSs can search freely on the entire search space. After that, the algorithm will invoke the SS and execute $\text{CS}_1, \text{CS}_2, \ldots, \text{CS}_N$, for $N = N - K$, $K \leq N - 1$, $N_{\min} = 1$. If the TC is met ($Gen \geq Max\_Gen$), terminate the search process and report the best solution found. Otherwise, the algorithm will activate the DS to discard some unlikely to be successful CSs and update $Gen = Gen + 1$ for proceeding a next generation. The PCS algorithm can be represented by the flow diagram as shown in Figure 3, where $\text{CS}_1, \text{CS}_2, \ldots, \text{CS}_N$ are the CS algorithms as represented by the pseudo code in Figure 2. Once all CSs work independently without any modifications or intrusion, the convergent property of the CSs is always preserved. Thus, it makes the PCS have the convergent property for the global optimization [29].

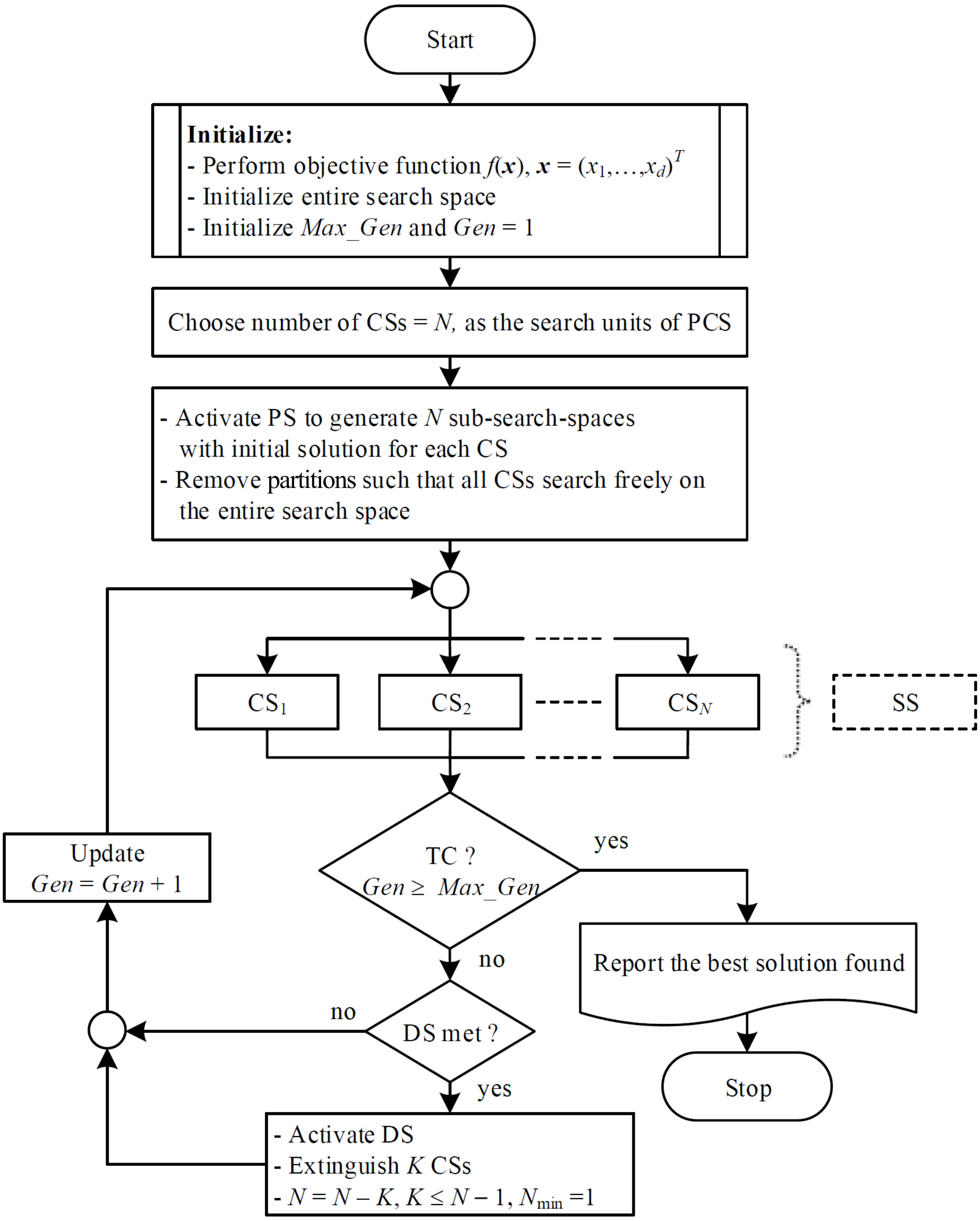


FIGURE 3. Flow diagram of PCS algorithm

For the complexity of algorithms, time and space (or size of problem) are two important resources needed by the algorithm for solving a problem. The time complexity of the algorithm is the number of steps required to solve a problem of size D. Generally, the complexity is defined in terms of the worst-case analysis and represented by Big-O notation [34]. The complexity of the original CS algorithm can be given by $O(n \cdot D \cdot t_{\max})$ [35,36], where n is the population number of cuckoos, D is the dimension size of a problem and $t_{\max}$ is the maximum number of generation. Compared to the original CS algorithm, the PCS algorithm proposed for running on a single-CPU platform has extra computation burden. The complexity of the PCS algorithm becomes $O(N \cdot n \cdot D \cdot t_{\max})$, where N is the number of CSs conducted by the PCS as the search units. This implies that the more the CSs conducted, the higher the complexity of the PCS algorithm. However, the complexity of the PCS algorithm can be improved to be equivalent to or almost the same as that of the original CS once the PCS algorithm is implemented on multicore or parallel-CPU platform.

3.2. **PCS-based MVRP optimization.** As mentioned earlier, the PCS algorithm was proposed for running on a single-CPU platform. The algorithm of the PCS contains multiple CSs as the search units. It consists of the PS to divide an entire search space into many sub-search spaces for each CS, SS to arrange the search units to run one-by-one on each generation and DS to terminate some unlikely to be successful CSs. In this work, the PCS algorithm is applied to optimally solving ten selected real-world MVRP. The PCS-based MVRP optimization can be described step-by-step as follows.

Step-1 For MVRP: initialize the objective function $f(\cdot)$ of the MVRP as stated in (1) and the constrained functions as stated in (2)-(8), initialize entire search space (number of locations and their corresponding distances) and number of vehicles m in the fleet, and define the depot among all locations.
For CS algorithm (search unit): initialize the population of cuckoo n, a fraction of worse nests p_a, and the step size α.
For PCS algorithm: initialize the number of CSs = N, *Max_Gen* and *Gen* = 1.

Step-2 Activate the PS to generate N sub-search-spaces with initial solutions $\boldsymbol{x}_i$ for each CS. Then, remove PS such that all N CSs can search freely on entire search space.

Step-3 Invoke the SS and execute $\text{CS}_1, \text{CS}_2, \ldots, \text{CS}_N$, for $N = N - K$, $K \leq N - 1$, $N_{\min} = 1$. From CS_1 to CS_N, n cuckoos find the new nests by randomly drawn from Lévy-flight distribution and lay their eggs (new solutions, $\boldsymbol{x}^*$) in the random nests. After that, the host bird will find cuckoo's egg with their ability fraction, p_a. If M ($M \leq n$) cuckoo's egg was found by the host bird, M cuckoo will find the new nests by randomly drawn from Lévy-flight distribution and lay their eggs (new solutions, $\boldsymbol{x}^*$) in the random nests again. Evaluate all cuckoo's eggs (new solutions, $\boldsymbol{x}^*$) via objective function $f(\cdot)$ of the MVRP in (1) and the constrained functions in (2)-(8). If $f(\boldsymbol{x}^*) < f(\boldsymbol{x})$, update $\boldsymbol{x} \leftarrow \boldsymbol{x}^*$.

Step-4 If the TC is met (*Gen* $\geq$ *Max_Gen*), terminate the search process. All best solutions of N CSs will be ranked. Among them, the best solution will be reported. Otherwise, go to Step-5.

Step-5 Activate the DS to discard some unlikely to be successful CSs. The convergent rates of all CSs will be compared. K CSs, $K \leq N - 1$, $N_{\min} = 1$, having lower convergent rate than $N - K$ CSs will be discarded. N CSs, $N = N - K$, $K \leq N - 1$, $N_{\min} = 1$, will be maintained to continue the search for the optimal solutions.

Step-6 Update *Gen* = *Gen* + 1 and go back to Step-3 for proceeding a next generation.

4. **Experimental Results and Discussions.** In this work, the MVRP with a single depot and the symmetric case are assumed. Ten real-world MVRP consisting of approximately 100-500 locations is selected from [37,38]. Details of ten selected MVRP are summarized in Table 1. For example, the 100 locations of MVRP#1 (KROA100) are plotted in Figure 4, where ◯ stands for the service locations and ■ stands for the depot. Once the symmetric case is assumed, $d_{ij} = d_{ji}$, the distance matrix of MVRP#1 (KROA100) is depicted in Figure 5.

In order to solve ten selected real-world MVRP in Table 1, the original CS and the PCS algorithms were coded by MATLAB version 2018b run on Intel(R) Core(TM) i5-3470 CPU@3.60GHz, 4.0GB-RAM. For each MVRP, 50 trial-runs are executed to search for their best solutions. Commonly, the depot will be defined by the (nearby) central location of each MVRP. The daily working time of any vehicle should not be longer than 8 hr. Therefore, $T_{\max} = 8$ hr. is set as the maximum working time in (8). The average speed of all vehicles is 80 km/hr., approximately. Thus, the overall distance of each vehicle

Table 1. Ten selected real-world MVRP problems

Problems	Names	Nunber of locations	Optimal tour for one vehicle (km.)	Comments
MVRP#1	KROA100	100	21,282	Nelson
MVRP#2	KROB100	100	22,141	Nelson
MVRP#3	CH150	150	6,528	Churritz
MVRP#4	BRG180	180	1,950	Rinaldi
MVRP#5	D198	198	15,780	Reinelt
MVRP#6	GIL262	262	2,378	Gillet
MVRP#7	LIN318	318	42,029	Kernighan
MVRP#8	PCB442	442	50,778	Juenger
MVRP#9	ATT532	532	92,794	Padberg
MVRP#10	PA561	561	2,763	Kleinschmidt

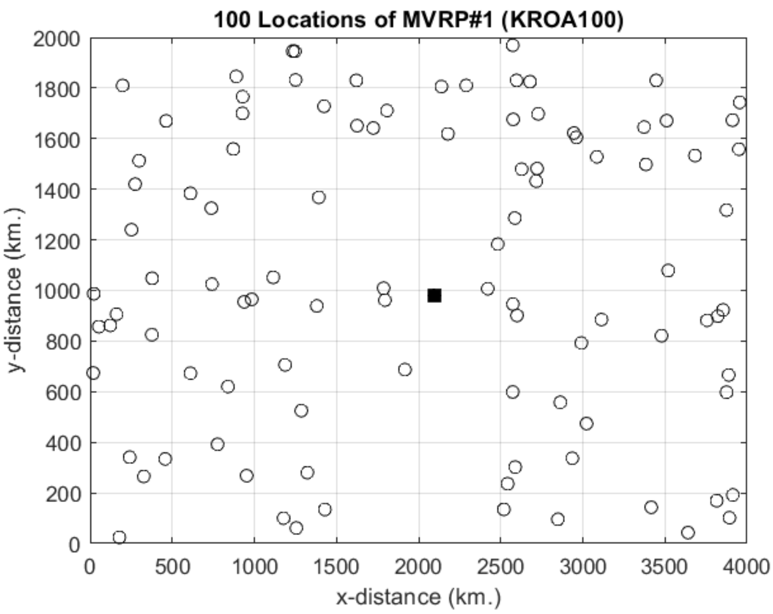


Figure 4. Locations of MVRP#1 (KROA100)

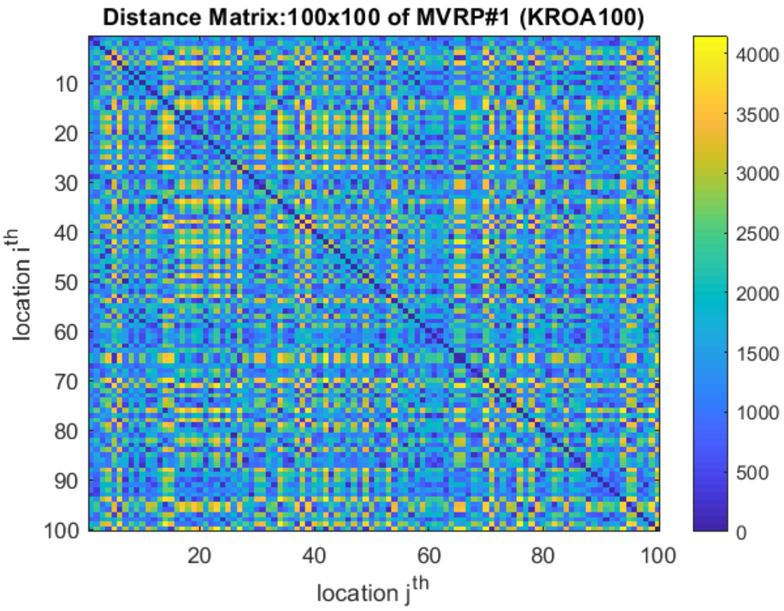


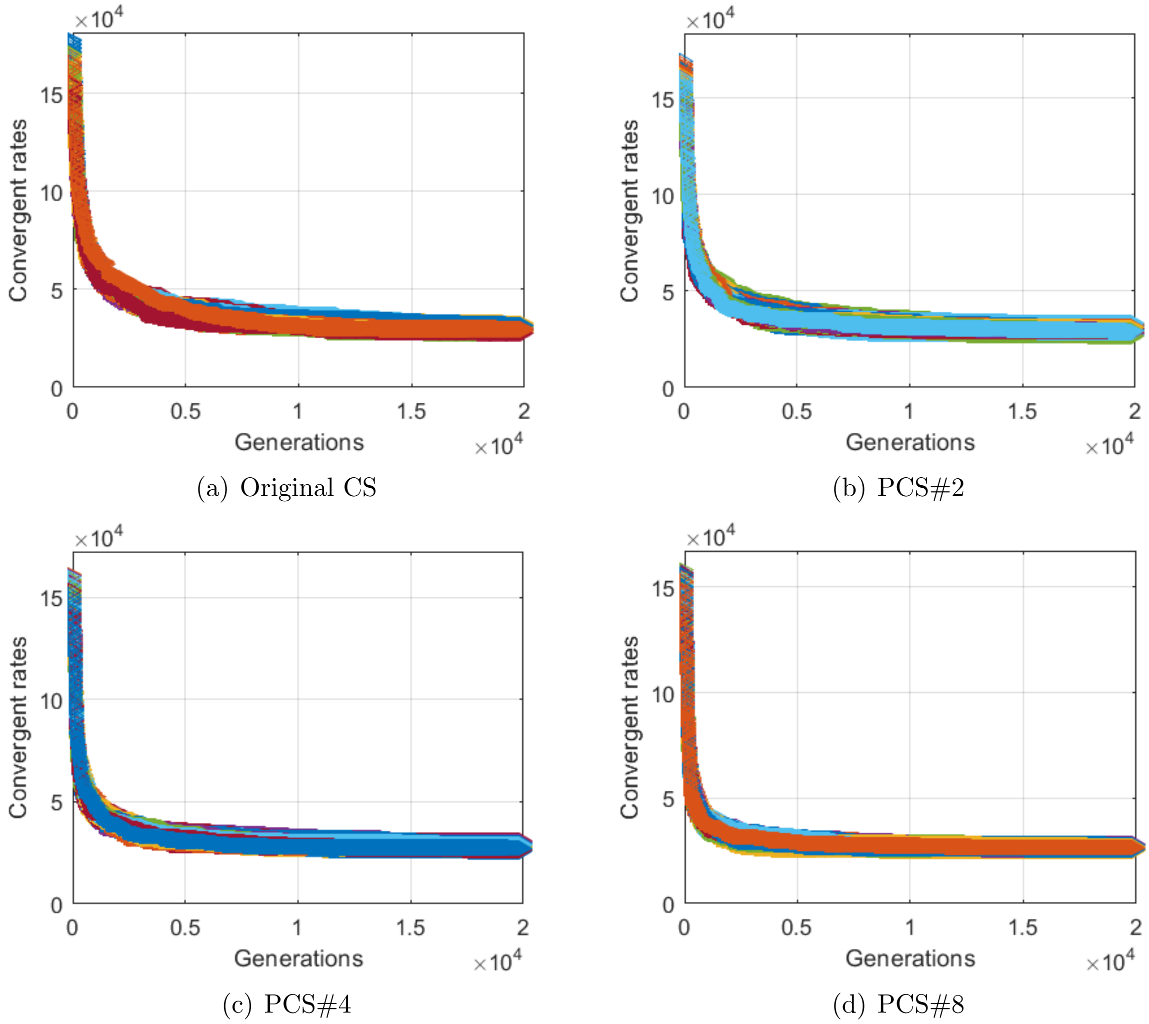
FIGURE 5. (color online) Distance matrix of MVRP#1 (KROA100)
(Color shades represent the distances between the ith and the jth locations)

TABLE 2. DS settings for PCS-based MVRP optimization

For all MVRP problems	The qth generation at which the DS is invoked and the number of discarded CSs					
	PCS#2	PCS#4		PCS#8		
	1st	1st	2nd	1st	2nd	3rd
qth generation	10,000	7,000	14,000	5,000	10,000	15,000
No. of discarded CSs	1	2	1	4	2	1

should not be longer than 640 km/day. These data are utilized to define the number of vehicles m in a fleet for each MVRP problem.

Performance comparisons of the MVRP optimization are made between the original CS and the PCS with 2, 4 and 8 CSs denoted as PCS#2, PCS#4 and PCS#8, respectively. The searching parameters of all CSs are set from the recommendations of Yang and Deb [30,31] and the preliminary studies with different ranges of parameters such as population number of cuckoos n, Lévy exponent β and discovery probability p_a. By varying n = 5, 10, 20, 30, 40 and 50, β = 0.5, 1.0, 1.5 and 2.0 and p_a = 0.1, 0.2, 0.3, 0.4 and 0.5, it was found that the best parameters for most cases are $n = 40$, $\beta = 1.5$ and $p_a = 0.2$. Thus, the searching parameters of all CSs are set as the same values, i.e., $n = 40$, $\beta = 1.5$ and $p_a = 0.2$ (20%). Max_Gen = 20,000 is set as the TC for both the original CS and PCS algorithms. For the PCS algorithm, the PS is set as the symmetrical boundaries of the sub-search-spaces for 2, 4 and 8 CSs for each function. The SS is set as a time-sharing technique for all CSs running one-by-one as sequential manner in each generation based on a single-CPU. The search process is repeated until one of the CSs hits the optimal solution or some CSs are discarded by the DS. For all MVRP problems, the DS is set as summarized in Table 2, where the qth generation is the generation at which the DS is activated. In each time of the DS activation, the number of CSs will be discarded by half. Eventually, there is only one CS left to continue the search for the global solutions.



(a) Original CS

(b) PCS#2

(c) PCS#4

(d) PCS#8

Figure 6. (color online) Convergent rates for MVRP#1 (KROA100) over 50 trial-runs

The convergent rates of the original CS, PCS#2, PCS#4 and PCS#8 for the MVRP#1 (KROA100) over 50 trial-runs are depicted in Figure 6 as the example. The convergent rates of other MVRP problems are omitted because they have a similar form to those in Figure 6. From Figure 6, it can be observed that the PCS#8 in Figure 6(d) performs higher convergent rates and more robust than the PCS#4 in Figure 6(c), the PCS#2 in Figure 6(b) and the original CS in Figure 6(a), respectively. Results of MVRP optimization obtained by the original CS, PCS#2, PCS#4 and PCS#8 are summarized in Table 3. For example, the optimal tours of 4 vehicles in a fleet of the MVRP#1 (KROA100) obtained by the original CS, PCS#2, PCS#4 and PCS#8 are displayed in Figures 7-10, where ◯ stands for the service locations, ■ stands for the depot, —— stands for the 1st vehicle route, – – – stands for the 2nd vehicle route, ······· stands for the 3rd vehicle route and -·-·- stands for the 4th vehicle route, respectively.

From the experimental results of all ten selected real-world MVRP problems in Table 3, the PCS#2 can yield shorter total distance than the original CS. The PCS#4 can give shorter total distance than the PCS#2, while the PCS#8 can provide shorter total distance than the PCS#4, PCS#2 and CS, respectively.

The average search times (AST) consumed by the original CS, PCS#2, PCS#4 and PCS#8 over 50 trial-runs are summarized in Table 4. It can be observed that the PCS#8

TABLE 3. Optimal tours of MVRP obtained by CS, PCS#2, PCS#4 and PCS#8

Problems	**No. of vehicles m**	**Optimal tour (km.)**			
		CS	**PCS#2**	**PCS#4**	**PCS#8**
MVRP#1	4	25,526.71	25,258.90	24,956.76	24,784.59
MVRP#2	5	26,802.34	26,430.72	25,144.82	25,013.46
MVRP#3	3	6,958.85	6,848.31	6,793.65	6,708.23
MVRP#4	4	2,310.68	2,270.18	2,247.30	2,103.56
MVRP#5	6	17,895.40	17,174.59	16,990.23	16,847.74
MVRP#6	4	2,867.52	2,803.46	2,705.68	2,640.18
MVRP#7	8	44,874.28	44,653.10	43,976.59	43,898.47
MVRP#8	9	53,892.06	53,669.74	52,561.42	52,308.60
MVRP#9	10	94,918.75	94,635.81	94,187.67	94,056.23
MVRP#10	5	3,304.50	3,286.62	3,012.78	2,926.56

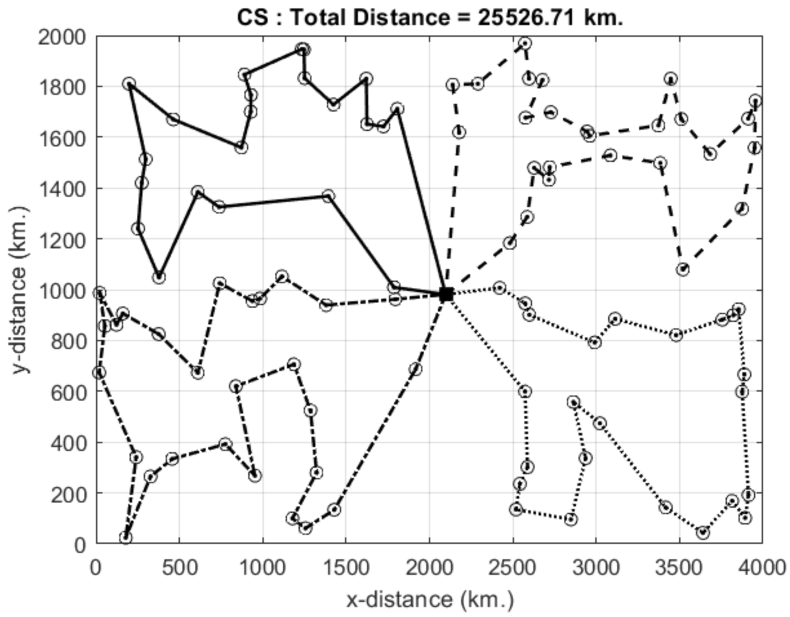


FIGURE 7. Optimal tour of MVRP#1 (KROA100) obtained by original CS

consumes the AST more than the PCS#4, PCS#2 and CS, respectively. This is because the PCS was proposed for running on a single-CPU platform. The more the search units (CSs) conducted in the PCS, the more the search time consumed. However, the AST of the PCS can be reduced by running the PCS algorithms on multicore or parallel processors. The numeric data in Table 4 can be converted to the equivalent AST (EAST) with-respect-to the original CS by using the formulation as stated in (14), where N is the number of CSs used in the PCS#N. The percent decrease of AST (PDAST) of the equivalent PCS with-respect-to the original CS can be calculated by using the formulation expressed in (15) for comparison as declared in Table 5. It can be noticed that the equivalent PCS#2, PCS#4 and PCS#8 averagely consume the AST less than the original CS by 28.81%, 52.14% and 68.53%, respectively.

$$\mathrm{EAST}_{\mathrm{PCS}\#N} = \frac{\mathrm{AST}_{\mathrm{PCS}\#N}}{N} \tag{14}$$

$$\mathrm{PDAST}_{\mathrm{PCS}} = 100 \times \left(\frac{\mathrm{AST}_{\mathrm{CS}} - \mathrm{EAST}_{\mathrm{PCS}}}{\mathrm{AST}_{\mathrm{CS}}} \right) \tag{15}$$

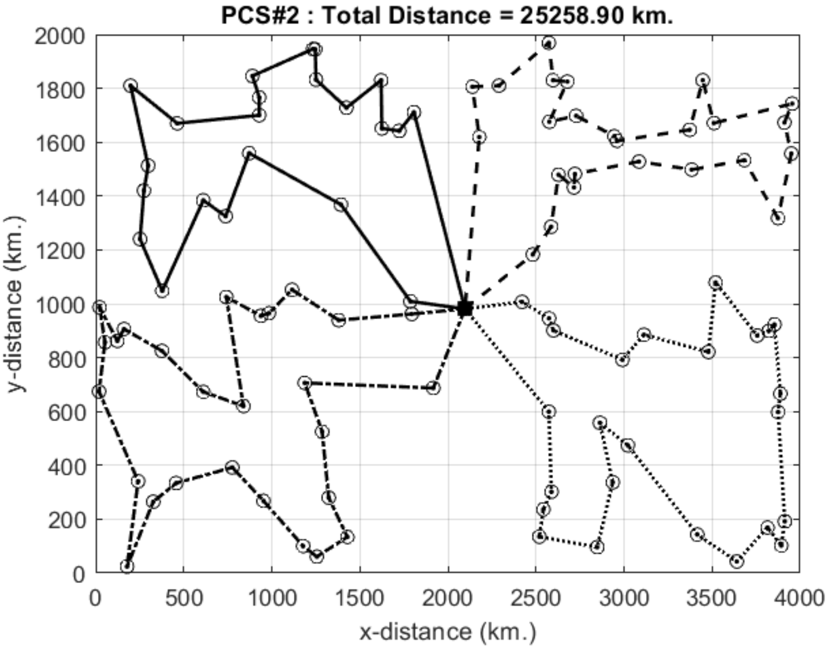


FIGURE 8. Optimal tour of MVRP#1 (KROA100) obtained by PCS#2

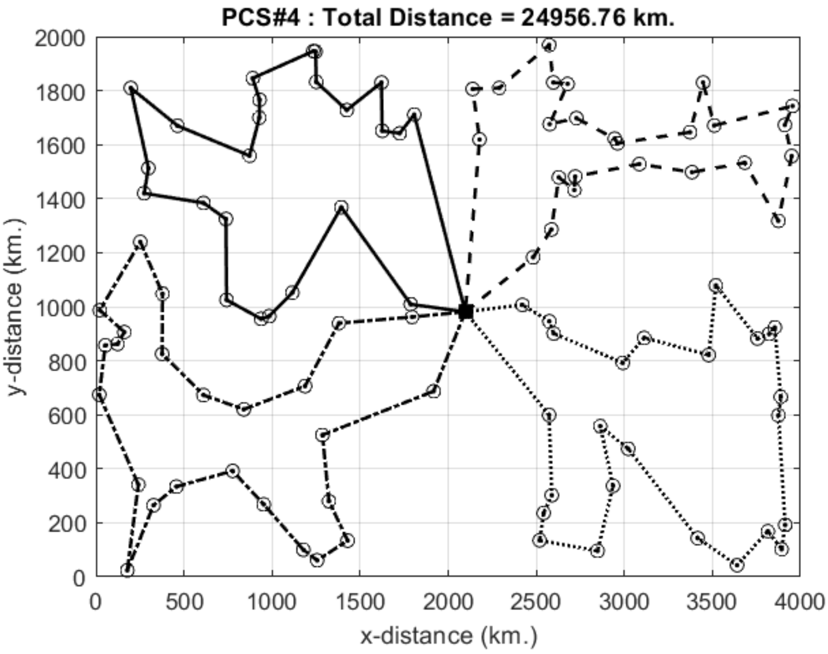


FIGURE 9. Optimal tour of MVRP#1 (KROA100) obtained by PCS#4

5. **Conclusions.** The application of the parallel cuckoo search (PCS) to optimally solving the multi-vehicle routing problems (MVRP) has been presented in this paper. The PCS is one of the latest modified versions of the original CS utilizing the random process drawn from the Lévy distribution. The PCS algorithm contains multiple CSs as the search units in order for running on a single-CPU platform. In addition, it possesses the partitioning strategy (PS) to divide sub-search spaces for each CS, the sequencing strategy (SS) to arrange the search units and the discarding strategy (DS) to terminate some unsuccessful CSs. For the MVRP optimization in this work, the PCS has been

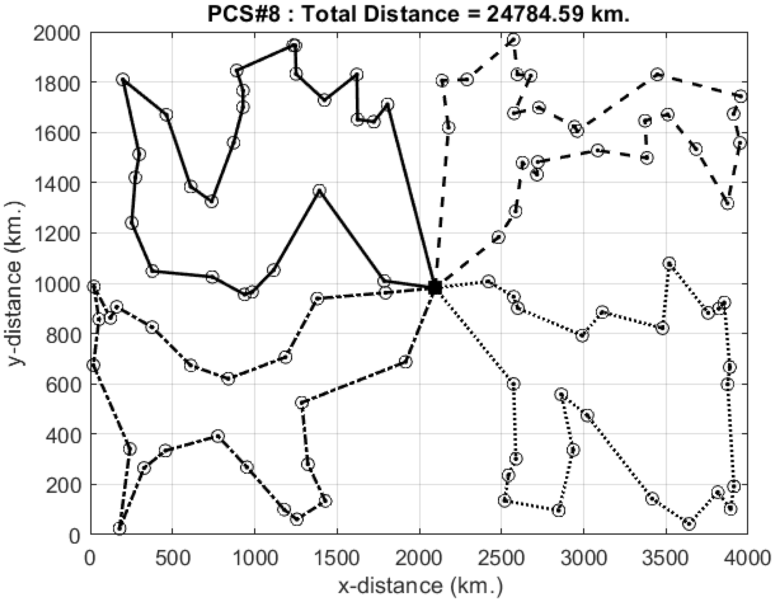


FIGURE 10. Optimal tour of MVRP#1 (KROA100) obtained by PCS#8

TABLE 4. Average search times (AST) of original CS, PCS#2, PCS#4 and PCS#8 for MVRP optimization

Problems	Average search time: AST (sec.)			
	CS	PCS#2	PCS#4	PCS#8
MVRP#1	34.0584	50.1270	67.8511	89.4706
MVRP#2	35.6406	52.4762	69.0234	91.6755
MVRP#3	8.6312	13.8304	17.5930	25.0882
MVRP#4	3.7883	5.9083	8.0216	10.1914
MVRP#5	28.6164	38.8031	51.2503	70.3850
MVRP#6	3.9488	6.1255	8.4152	11.6813
MVRP#7	56.4309	74.4682	102.7805	138.2991
MVRP#8	68.5267	86.7304	114.2289	147.6042
MVRP#9	134.2802	202.2307	275.3518	354.6407
MVRP#10	4.2648	7.7872	9.4686	13.2154
Averages	37.8186	53.8487	72.3984	95.2251

applied to optimally solving ten selected real-world MVRP consisting of approximately 100-500 locations based on the modern optimization approach. Results obtained by the PCS containing 2, 4 and 8 CSs (PCS#2, PCS#4 and PCS#8) were compared with those obtained by the original CS. From experimental results, it was found that the PCS#8 can provide the optimal tours for all selected MVRP with shorter total distance than the PC-S#4, PCS#2 and CS, respectively. This can be concluded that the PCS can give optimal solutions of all ten selected MVRP with shorter total distance than the CS, significantly. However, by running on a single-CPU platform, the search time consumed by the PCS is more than that consumed by the original CS. Such the search time can be reduced by running the PCS algorithms on multicore or parallel processors. For the future research, the more practical MVRP problems will be studied and expanded to address the variety of conditions in real-world applications. Then, the PCS will be applied to solving the

Table 5. Equivalent average search time (EAST) of PCS#2, PCS#4 and PCS#8 with-respect-to original CS for MVRP optimization

Problems	Equivalent AST: EAST (sec.) with-respect-to the CS			
	CS	PCS#2	PCS#4	PCS#8
MVRP#1	34.0584	25.0635	16.9628	11.1838
MVRP#2	35.6406	26.2381	17.2559	11.4594
MVRP#3	8.6312	6.9152	4.3983	3.1360
MVRP#4	3.7883	2.9542	2.0054	1.2739
MVRP#5	28.6164	19.4016	12.8126	8.7981
MVRP#6	3.9488	3.0628	2.1038	1.4602
MVRP#7	56.4309	37.2341	25.6951	17.2874
MVRP#8	68.5267	43.3652	28.5572	18.4505
MVRP#9	134.2802	101.1154	68.8380	44.3301
MVRP#10	4.2648	3.8936	2.3672	1.6519
Averages	37.8186	26.9244	18.0996	11.9031
PDAST (%)	0%	28.8065%	52.1410%	68.5258%

multiple-depots multiple-vehicle routing problems (MDMVRP) for minimizing total distance beyond all routes and balancing the number of vehicles, number of service locations, traveling times, vehicles' capacity and the distance of each vehicle in the fleet.

Acknowledgment. This paper was funded by King Mongkut's University of Technology North Bangkok with contract No. KMUTNB-64-DRIVE-25.

REFERENCES

[1] G. B. Dantzig and J. H. Ramser, The truck dispatching problem, *Management Science*, vol.6, no.1, pp.80-91, 1959.
[2] B. L. Golden and A. A. Assad, *Vehicle Routing: Methods and Studies*, Elsevier Science, Amsterdam, 1988.
[3] S. Kikuchi and P. Chakroborty, Place of possibility theory in transportation analysis, *Transportation Research, Part B*, vol.40, pp.595-615, 2006.
[4] T. Bektas, The multiple traveling salesman problem: An overview of formulations and solution procedures, *Omega*, vol.34, no.3, pp.209-219, 2006.
[5] A. Király and J. Abonyi, Optimization of multiple traveling salesmen problem by a novel representation based genetic algorithm, *Proc. of the 10th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics*, pp.315-326, 2011.
[6] H. Larki and M. Yousefikhoshbakht, Solving the multiple traveling salesman problem by a novel metaheuristic algorithm, *Journal of Optimization in Industrial Engineering*, vol.16, pp.55-63, 2014.
[7] H. Fleishner, Traversing graphs: The Eulerian and Hamiltonian theme, in *Arc Routing, Theory, Solutions, and Applications*, M. Dror (ed.), Kluwer Acad. Publ., 2000.
[8] G. Nagy and S. Salhi, Heuristics algorithm for single and multiple depot vehicle routing problem with pickups and deliveries, *European Journal of Operational Research*, vol.162, pp.126-141, 2005.
[9] P. Sombunthama and V. Kachitvichyanukul, Multi-depot vehicle routing problem with pickup and delivery requests, *IAENG Trans. Engineering Technologies*, vol.5, pp.71-85, 2010.
[10] A. Modares, S. Somhom and T. Enkawa, A self-organizing neural network approach for multiple traveling salesman and vehicle routing problems, *International Transactions in Operational Research*, vol.6, no.6, pp.591-606, 1999.
[11] X. C. Shi, Y. C. Liang, H. P. Lee, C. Lu and Q. X. Wang, Particle swarm optimization-based algorithms for TSP and generalized TSP, *Information Processing Letters*, vol.103, pp.169-176, 2007.
[12] P. Sombuntham and V. Kachitvichayanukul, A particle swarm optimization algorithm for multi-depot vehicle routing problem with pickup and delivery requests, *Proc. of the International Multi-Conference of Engineers and Computer Scientists*, vol.3, 2010.

[13] M. M. Tavakoli and A. Sami, Particle swarm optimization in solving capacitated vehicle routing problem, *Bulletin of Electrical Engineering and Informatics*, vol.2, no.4, pp.252-257, 2013.

[14] Y.-J. Gong, J. Zhang, O. Liu, R.-Z. Huang, H. S.-H. Chung and Y.-H. Shi, Optimizing the vehicle routing problem with time windows: A discrete particle swarm optimization approach, *IEEE Trans. Systems, Man, and Cybernetics – Part C: Applications and Reviews*, vol.42, no.2, pp.254-267, 2012.

[15] S. Zou, J. Li and X. Li, A hybrid particle swarm optimization algorithm for multi-objective pickup and delivery problem with time windows, *Journal of Computers*, vol.8, no.10, pp.2583-2589, 2013.

[16] C. Evelyn, C. T. Brown, A. Ragsdale and A. E. Carter, A grouping genetic algorithm for the multiple traveling salesperson problem, *International Journal of Information Technology & Decision Making*, vol.6, no.2, pp.333-347, 2007.

[17] W. Zhou and Y. Li, Improved genetic algorithm for multiple traveling salesman problem, *Proc. of the 2nd International Asia Conference on Informatics in Control, Automation and Robotics*, vol.1, pp.493-495, 2010.

[18] J. W. Escobar, R. Linfati, P. Toth and M. Baldoquin, A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem, *Journal of Heuristics*, vol.20, no.5, pp.1-27, 2014.

[19] Y. Kao, M.-H. Chen and Y.-T. Huang, A hybrid algorithm based on ACO and PSO for capacitated vehicle routing problems, *Mathematical Problems in Engineering*, pp.1-17, 2012.

[20] N. E. Toklu, L. M. Gambardella and R. Montemanni, A multiple ant colony system for a vehicle routing problem with time windows and uncertain travel times, *Journal of Traffic and Logistics Engineering*, vol.2, no.1, pp.52-58, 2014.

[21] Y. Yang, J. Sun and J. Cui, Evolutionary game analysis on the choice of vehicle sharing travel modes, *ICIC Express Letters, Part B: Applications*, vol.11, no.3, pp.289-295, 2020.

[22] H. Zheng, Y.-Q. Zhou and Q. Luo, A hybrid cuckoo search algorithm-GRASP for vehicle routing problem, *Journal of Convergence Information Technology*, vol.8, no.3, pp.821-828, 2013.

[23] M. Alssager and Z. A. Othman, Cuckoo search algorithm for capacitated vehicle routing problem, *Journal of Theoretical and Applied Information Technology*, vol.88, no.1, pp.11-19, 2016.

[24] A. Goli, A. Aazami and A. Jabbarzadeh, Accelerated cuckoo optimization algorithm for capacitated vehicle routing problem in competitive conditions, *International Journal of Artificial Intelligence*, vol.16, no.1, pp.88-112, 2018.

[25] S. George and S. Binu, Vehicle route optimisation using artificial bees colony algorithm and cuckoo search algorithm – A comparative study, *International Journal of Applied Engineering Research*, vol.13, no.2, pp.953-959, 2018.

[26] S. Suwannarongsri, Solving multiple vehicle routing problems with time constraints by flower pollination algorithm optimization, *WSEAS Trans. Systems*, vol.19, pp.178-187, 2020.

[27] S. Suwannarongsri, Application of modified flower pollination algorithm to multiple vehicle routing problems with time constraints, *WSEAS Trans. Systems and Control*, vol.15, pp.459-467, 2020.

[28] T. Saeheaw, Parameter tuning of the HCSCROCFO-3Opt algorithm for solving the capacitated vehicle routing problem, *International Journal of Industrial Engineering Computations*, vol.11, no.4, pp.481-490, 2020.

[29] S. Suwannarongsri, Parallel cuckoo search for global optimization, *International Journal of Innovative Computing, Information and Control*, vol.17, no.3, pp.887-903, 2021.

[30] X. S. Yang and S. Deb, Cuckoo search via Lévy flights, *Proc. of the World Congress on Nature & Biologically Inspired Computing (NaBIC2009)*, pp.210-214, 2009.

[31] X. S. Yang and S. Deb, Multiobjective cuckoo search for design optimization, *Computers & Operations Research*, vol.40, no.6, pp.1616-1624, 2013.

[32] D. Chen and Z. Yang, Multiple depots vehicle routing problem in the context of total urban traffic equilibrium, *Journal of Advanced Transportation*, vol.2017, pp.1-14, 2017.

[33] T. Jitwang and D. Puangdownreong, Application of cuckoo search to robust PIDA controller design for liquid-level system, *International Journal of Innovative Computing, Information and Control*, vol.16, no.1, pp.189-205, 2020.

[34] E. Talbi, *Metaheuristics from Design to Implementation*, John Wiley & Sons, New Jersey, 2009.

[35] X. S. He, F. Wang, Y. Wang and X. S. Yang, Global convergence analysis of cuckoo search using Markov theory, in *Nature-Inspired Algorithms and Applied Optimization. Studies in Computational Intelligence*, X. S. Yang (ed.), Cham, Springer, 2017.

[36] R. Salgotra, U. Singh and S. Saha, New cuckoo search algorithms with enhanced exploration and exploitation properties, *Expert Systems with Applications*, vol.95, pp.384-420, 2018.

[37] TSPLIB95, *Symmetric Traveling Salesman Problem*, http://comopt.ifi.uniheidelberg.de/software/TSPLIB95/, 2011.

[38] M. Held and R. Karp, A dynamic programming approach to sequencing problems, *SIAM Journal*, vol.1, pp.196-210, 1962.

Author Biography

Supaporn Suwannarongsri received the B.Eng. degree in industrial engineering from Thonburi University (TRU), Bangkok, Thailand, 2004; the M.Eng. degree in industrial engineering from King Mongkut's Institute of Technology Ladkrabang (KMITL), Bangkok, Thailand, 2008; the Ph.D. degree in sustainable energy and environment technology and management from Rajamangala University of Technology Rattanakosin (RMUTR), Nakhon Pathom, Thailand, 2014.

Assoc. Prof. Supaporn Suwannarongsri is currently an associate professor at the Department of Materials Handling and Logistics Engineering, Faculty of Engineering, King Mongkut's University of Technology North Bangkok (KMUTNB), Bangkok, Thailand. Her research interests include logistics and materials handling, operation research, production planning and design, and applications of metaheuristic algorithms to various real-world industrial engineering problems. She has published over 80 papers in journals and conferences, nationally and internationally.