sciendo

PRODUCTION ENGINEERING ARCHIVES 2024, 30(3), 303-313

PRODUCTION ENGINEERING ARCHIVES

ISSN 2353-5156 (print)
ISSN 2353-7779 (online)

Exist since 4th quarter 2013
Available online at https://pea-journal.eu

Application of Metaheuristics for Multi-Trip Capacitated Vehicle Routing Problem with Time Window

Kantapong Niyomphon[1], Warisa Nakkiew[1]*

[1] Department of Industrial Engineering, Faculty of Engineering, Chiang Mai University, Chiang Mai, Thailand
*Correspondence: warisa@eng.cmu.ac.th

Article history
Received 08.03.2024
Accepted 25.06.2024
Available online 09.09.2024

Keywords
Vehicle Routing Problem,
Multi-trip,
Time window,
Particle Swarm Optimization,
Differential Evolution.

Abstract
This study focuses on the delivery routing problem faced by a transport company located in Phuket, Thailand. The goal of this study is to find a daily optimum route in order to minimize the total transportation cost, which comprises fixed costs associated with vehicle rental and variable costs calculated based on factors of travel distance, fuel prices, and fuel consumption. The complexity of this problem is compounded by the fact that customer demand often exceeds a vehicle capacity, in terms of weight and volume. In addition, delivery must be made within specific time windows. To tackle this issue, the delivery routing problem is classified as a multi-trip capacitated vehicle routing problem with time window (MTCVRPTW). Since the problem is NP-hard, an application of metaheuristic is more practical to determine the delivery routing of the company within a reasonable computing time. In this study, Particle Swarm Optimization (PSO) and Differential Evolution (DE) algorithm are applied to solve MTCVRPTW. The numerical results show that DE provides better solution quality compared to those obtained from PSO and company current practices.

DOI: 10.30657/pea.2024.30.30

1. Introduction

Transportation costs are the key concern of logistics companies across the world. The fact shows that transportation costs contribute almost half of the total logistic cost, making it one of the key factors in identifying a performance-driven logistics industry (Banomyong et al., 2021). A considerable proportion of transportation costs includes expenses related to vehicles, fuel prices, driver wages, and so on. Therefore, it is necessary for the logistics industry to find an efficient way to improve delivery schedules or delivery routing. One common approach to enhance a company's competitive advantages is cost minimization. In the field of optimization, the management of vehicle routing is a well-established technique for reducing these costs.

The Vehicle Routing Problem (VRP) is a mathematical optimization problem that aims to identify the most efficient routes for a fleet of vehicles to deliver goods to a specific set of customers. In the literature, efficient delivery routing in VRP aims to achieve numerous objectives, such as minimizing transportation time, distance, and costs, maximizing vehicle capacity usage, meeting customer demands, reducing environmental impact, enhancing operational productivity, and effectively adapting to dynamic conditions. Through route optimization, companies can achieve significant cost savings, improve resource allocation, enhance customer satisfaction, reduce environmental footprint, boost operational efficiency, and maintain flexibility in response to changing circumstances. The VRP is a fundamental problem in logistics and transportation that has been extensively studied in literature. Typically, this problem is formulated as a single depot problem, where vehicles originate from a central depot and need to visit multiple customer locations (Clarke and Wright, 1964). VRP is well-known as a combinatorial optimization problem, similar to the Traveling Salesman Problem (TSP), with the only difference being the constraint on vehicle capacity (Tirkolaee et al., 2018). According to Eraslan and Derya (2010), a variant of the VRP, known as the Vehicle Routing Problem with Time Window (VRPTW) introduces the constraint of time windows for customer visits. The problem is particularly relevant in real-world scenarios as it considers the constraint that customers may only be available to receive deliveries within specific time windows. Moreover, to tackle the challenge of high customer demand exceeding vehicle capacity, some researchers presented the Multi-trip Vehicle Routing Problem with Time Window (MTVRPTW) model (Maffioli,

2003; Cattaruzza et al., 2016; Neira et al., 2020: Zhen et al., 2020). This model considers the necessity for multiple trips to fulfill customer demands and considers the requirement to deliver goods at specified times. The MTVRPTW model is an extension of the VRPTW and, therefore, offers a more realistic representation of practical distribution problems.

The VRP problem is easy to explain but challenging to solve. Typically, it takes exponential time to find the optimal solution, making it a nondeterministic polynomial time (NP-hard) problem (Utamima et al. 2015; Moghaddam et al., 2012). While various solution techniques have been proposed to address VRP, achieving a global minimum solution remains computationally complex. Solving VRP often requires the application of specific algorithms capable of determining the optimal solution. One traditionally approach is the used of exact algorithm, which systematically explores the entire solution space using techniques such as Branch and Bound and Branch-and-Cut (Du and He, 2012; Hernandez et al., 2014; Tang et al., 2015; Theurich et al., 2021). These methods are proficient at identifying the optimal solution among all possible solutions. However, it is important to note that the MTVRPTW is an NP-hard problem (Geetha et al., 2012), and the exact algorithm is restricted to small-scale problems due to the prohibitively long computational time required for solving larger instances. An alternative method, known as the approximate approach, provides a practical solution for efficiently addressing large-scale problems. When seeking an appropriate solution to the VRP within a reasonable timeframe, it is suitable to employ metaheuristics, which are effective for finding optimal solutions in such scenarios. Metaheuristic methods offer flexible and efficient problem-solving strategies that explore solution spaces to identify optimal or near-optimal solutions. Unlike exact methods, which exhaustively search all possibilities, metaheuristics utilize iterative improvement or exploration strategies inspired by natural phenomena or computational paradigms. The family of metaheuristic algorithms includes various options such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Memetic Algorithm (MA), Simulated Annealing (SA), Tabu Search (TS), and Large Neighborhood Search (LNS) (Tirkolaee et al., 2019; Lyu and He, 2021; Yağmur and Kesen, 2021; Yeh and Tan, 2021; Dumez et al., 2021; Kumari et al.,2023). These algorithms balance exploration and exploitation to efficiently navigate complex solution spaces. Metaheuristics are flexible, applicable to various optimization problems, and capable of finding high-quality solutions within reasonable timeframes. Their adaptability and effectiveness in handling diverse problem types make them helpful tools for effectively addressing real-world optimization challenges. Metaheuristic algorithms, particularly population-oriented ones like Particle Swarm Optimization (PSO), have gained significant power in addressing VRP, as evidenced by the studies of MirHassani and Abolghasemi (2011) and Xu et al. (2015). PSO stands out for its superior solution quality, fast computation speed, and user-friendly programming, as highlighted in research by Marinakis et al. (2013). Its implementation often yields better results in VRP compared to GA. Additionally, the Differential Evolution algorithm (DE) has emerged as a powerful tool for VRP optimization, frequently surpassing PSO in terms of solution quality (Hu and Wu, 2010; Wisittipanich et al., 2021; Wisittipanit et al., 2021). The algorithm exhibits its own distinct performance characteristics, as demonstrated in Kachitvichyanukul (2012) comparative analysis of GA, PSO, and DE. GA's requirement for ranking solutions adds complexity compared to PSO and DE, which do not necessitate such ranking. Population size significantly affects solution time in GA, whereas the impact is linear for PSO and DE, highlighting GA's sensitivity to population changes. PSO emphasizes exploiting the best solution, while DE ensures stable average fitness, resulting in more consistent convergence and less premature convergence compared to GA and PSO. Both DE and PSO demonstrate higher continuity and superior global search capabilities, enabling them to find quality solutions without local search. While GA and PSO benefit from homogeneous subgrouping for convergence, DE does not rely on this feature.

This research was inspired by delivery routing problem of a transport company located in Phuket province, Thailand. Despite the importance of efficient delivery routing, the company has not prioritized finding solutions to the VRP. Instead, they often rely on the experience of their delivery drivers. Consequently, their current method may not be optimal due to the complexity of routing problems and the varying skills and experience levels of individual employees.

Therefore, the purpose of this research is to utilize two metaheuristic algorithms, PSO and DE, to solve the delivery routing in a case study. The goal is to find an optimal delivery schedule with respect to minimization of total transportation costs. The proposed approach can be adapted for use in other businesses encountering similar challenges.

The remainder of this paper is organized as follows. The problem description is provided in Section 2. Section 3 presents the mathematical model of the problem. Section 4 presents the proposed PSO and DE algorithms and their applications to the problem. The experiment and result analysis conducted to compare performance between two metaheuristics are reported in Section 5. Finally, conclusion and recommendations for further research are given in Section 6.

2. Problem statement

As mentioned previously, this paper is inspired by delivery routing problem of a transport company located in Phuket province, Thailand. The company is responsible for the delivery process which offers a range of parcel delivery services, including miscellaneous goods, construction materials, equipment, wheels, gas tanks, water pumps, and others. The company's data reveals that the number of customers and the proportion of delivered parcels vary daily, and the company’s current practice utilizes an employee to arrange delivery routing on a daily basis. However, the selection of vehicles and the sequence of deliveries are based on employee’s intuition and experience, which may not result in minimizing overall cost.

The distribution area is divided into six lines: Choeng Thale and Kamala (Line 1), Kathu and Patong (Line 2), Haryak (Line 3), Thalang (Line 4), near depot (Line 5), and City (Line

6). The company has a total of eleven vehicles, which can be classified into two vehicle categories: 6-wheeled and 10-wheeled vehicles. This delivery routing problem is considered as a heterogeneous vehicle routing problem, and thus the selection of proper vehicle affects delivery cost. The difficulty of this problem is compounded by the fact that customer demand often exceeds the capacity of a single vehicle, in terms of weight and volume, leading to the deployment of multiple vehicles and multiple delivery trips to ensure that all parcels are successfully delivered. In addition, deliveries must be made within the customer's specific time windows. Therefore, the problem in this study is classified as the Multi-trip Capacitated Vehicle Routing Problem with Time Window (MTCVRPTW).

The conceptual model of the distribution system used in this study is illustrated in Fig. 1.

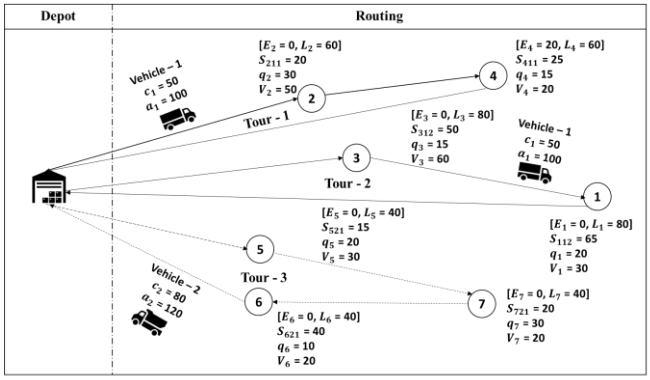

Fig. 1. The conceptual model of distribution system

To enhance comprehension of the problem, an example of a specific instance with seven customers and two heterogeneous capacitated vehicles is explained. According to Fig. 1, the variable s denotes the arrival time of the vehicle at each node, and $[E_i, L_i]$ corresponds to the time window allocated for each customer. The variables q and v stand for weight and volume size of a customer's parcel, respectively. In this case, weight (c) and volume (a) capacities are set as 50 and 100 for vehicle 1, and 80 and 120 for vehicle 2. One of the feasible solutions in Fig. 1 illustrates that vehicle 1 travels for two delivery trips. First, vehicle 1 departs from the depot to nodes 2, 4, and comes back to the depot (Tour 1). Then, it loads parcels for the next trip and travels from the depot to nodes 3, 1, and finally returns to the depot (Tour 2). Vehicle 2 completes its delivery in a single trip, moving from the depot to nodes 5, 7, 6, and coming back to the depot (Tour 3). In terms of time windows, this example ensures that all customers are served within their respective constraints.

3. Mathematical Model

The mathematical model of MTCVRPTW presented in this section is referred to the model proposed by Niyomphon and Wisittipanich (2022). The problem was formulated as Mixed Integer Linear Programming (MILP). The objective of the model is to minimize overall transportation costs, considering both fixed and variable costs. The fixed cost is associated with vehicle which depends on the type of vehicle assigned for delivery. On the other hand, the variable cost is calculated based on delivery sequence solutions, incorporating various factors which are travel distance, fuel price, and fuel consumption rate. The model operates under certain key assumptions, which are enumerated as follows:

- All vehicles must begin and end at the same depot for each trip.
- Each customer can be visited only once and exclusively by a single vehicle.
- Total customer demand during any given trip must not exceed the vehicle capacity.
- Vehicles are heterogeneous and may undertake multiple trips.
- Time windows are specified for each node, and each vehicle must visit within these time frames.
- The fuel price of all vehicles is assumed to be the same.
- The matrix of traveling time and distance is asymmetric, and a relationship exists between traveling time and distance.
- There is a single depot in the distribution system.

The notations including sets, indices, parameters, and decision variables are listed as follows.

Indices

i, j : Depot (i, j = 1) and customer (i, j = 2, 3,…, N)
k : Vehicle (k = 1, 2,…, K)
r : Vehicle trip (r = 1, 2,…, R)

Parameters

d_{ij} : Distance from i to j (km)
t_{ij} : Traveling time from i to j (mins)
FC_k : Fixed cost of vehicle k (Bath/day)
F_k : Fuel price of vehicle k (Bath/liter)
f_k : Fuel consumption rate of vehicle k (km/liter)
q_j : Parcel weight (kg)
v_j : Parcel volume (cm^3)
c_k : Weight capacity of vehicle k (kg)
a_k : Volume capacity of vehicle k (cm^3)
$[E_i, L_i]$: Time window of customer i (mins)
U_i, U_j : Constant value used to avoid a subtour
h_i : Operating time at customer i (mins)
TR : Loading time at depot (mins)
M : Large number
N : Number of customers
K : Number of vehicles
R : Number of trips

Decision variable

x_{ijkr} : 1 if trip r of vehicle k travels from i to j or 0 otherwise
y_{ikr} : 1 if trip r of vehicle k visits customer i or 0 otherwise
s_{ikr} : Visiting time at customer i vehicle k in trip r
u_k : 1 if vehicle k is used or 0 otherwise

Objective function

$$min \sum_{k=1}^{K} FC_k u_k + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K}\sum_{r=1}^{R} \frac{F_k d_{ij} x_{ijkr}}{f_k} \quad (1)$$

Constraints

$$\sum_{k=1}^{K}\sum_{r=1}^{R} y_{ikr} = 1 \qquad \forall_i; i = 2,3,\dots,N \quad (2)$$

$$\sum_{j=1}^{N} x_{ijkr} = y_{ikr} \qquad \forall_i \forall_k \forall_r \quad (3)$$

$$\sum_{j=1}^{N} x_{jikr} = y_{ikr} \qquad \forall_i \forall_k \forall_r \quad (4)$$

$$\sum_{i=2}^{N} q_i y_{ikr} \le c_k \qquad \forall_k \forall_r \quad (5)$$

$$\sum_{i=2}^{N} v_i y_{ikr} \le a_k \qquad \forall_k \forall_r \quad (6)$$

$$\sum_{j=2}^{N} x_{1jkr} \le u_k \qquad \forall_k \forall_r \quad (7)$$

$$\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{r=1}^{R} x_{ijkr} \le M u_k \qquad \forall_k \quad (8)$$

$$s_{ikr} + h_i + t_{ij} + M(x_{ijkr} - 1) \le s_{jkr} \qquad \forall_i \forall_j \forall_k \forall_r;\ i \ne j \quad (9)$$

$$s_{1kr} + TR + t_{1j} + M(x_{1jkr} - 1) \le s_{jkr} \qquad \forall_j \forall_k \forall_r; r = 2,3,\dots,R \quad (10)$$

$$s_{ikr} + h_i + t_{ij} + M(x_{i1kr} - 1) \le s_{1kr+1} \qquad \forall_i \forall_k \forall_r; r = 1,2,\dots,R-1 \quad (11)$$

$$E_i y_{ikr} \le s_{ikr} \le L_i y_{ikr} \qquad \forall_i \forall_k \forall_r \quad (12)$$

$$U_i - U_j + N\sum_{k=1}^{K} x_{ijkr} \le N - 1 \qquad \forall_j \forall_r; i,j = 2,3,\dots,N \text{ and } i \ne j \quad (13)$$

$$x_{ijkr} \in \{0,1\} \qquad \forall_i \forall_j \forall_k \forall_r \quad (14)$$

$$y_{ikr} \in \{0,1\} \qquad \forall_i \forall_k \forall_r \quad (15)$$

$$u_k \in \{0,1\} \qquad \forall_k \quad (16)$$

Equation (1) represents the objective function of the model, total cost minimization. Equation (2) ensures that each customer is visited exactly once and exclusively by a single vehicle. Equations (3) and (4) guarantee that each trip begins and terminates at the depot. Equations (5) and (6) ensure the load capacity of the vehicles, weight, and volume respectively, are not exceeded. Equation (7) is a constraint that tracks the number of vehicles utilized when leaving the depot. Equation (8) ensures that each vehicle can only be utilized if its usage cost is paid. Equation (9) ensures that visiting time of a vehicle from one point to another does not exceed specified time window. Equation (10) is a constraint that accounts for the loading time occurring from the second trip onwards. Equation (11) guarantees the continuity of multiple trips. Equation (12) ensures that visiting times of vehicles adhere to time windows specified for customer. Equations (13) is used to eliminate sub-tours. Equations (14), (15), and (16) represent that decision variables are binary variables.

4. Application of PSO and DE for MTCVRPTW

This section presents applications of PSO and DE algorithm to solve MTCVRPTW.

4.1. Particle Swarm Optimization (PSO)

PSO was introduced by Kennedy and Eberhart in 1995. PSO draws its inspiration from the social behaviors exhibited by animals, exemplified by the foraging behavior of birds or fishes. It involves initializing a set of candidate solutions to search for an optimal solution within a given search space. PSO operates on the premise that each particle can be analogously likened to a particle. Each particle represents a candidate solution and is scattered throughout the search space, thus combining into a collective known as a swarm. The pseudocode of PSO algorithms applied in this study is provided in Fig. 2.

Particle Swarm Optimization (PSO)	
[1]	Set the following parameters: population size, number of iterations, cognitive learning factor (*Cp*), social learning factor (*Cg*), and inertia weight (*W*)
[2]	Initialize a swarm of particles with random *position* and zero *velocity*
[3]	**For** *t* = 1 to the maximum iteration
[4]	**For** *i* = 1 to the swarm size
[5]	Decode to obtain the customer and the vehicle sequence
[6]	Consider the time window and capacity constraints
[7]	Evaluate objective value or fitness value
[8]	**If** particle's fitness value is better than its *Pbest* in history:
[9]	Update *Pbest* to current position
[10]	**End if**
[11]	**If** particle's fitness value is better than *Gbest*:
[12]	Update Gbest to *current position*
[13]	**End if**
[14]	**For** *d* = 1 to the total dimension
[15]	Generate a random number *R1* between 0 and 1
[16]	Generate a random number *R2* between 0 and 1
[17]	Update particle's velocity using the formula: *Velocity* [*d*] = *W* Velocity* [*d*] + *Cp * R1 **(*Pbest – Current position*[*d*]) + *Cg * R2 ** (*Gbest – Current position*[*d*])
[18]	Update particle's position using the formula: *Position*[*d*] = *Position*[*d*] + *Velocity*[*d*]
[19]	**End for**
[20]	**End for**
[21]	**End for**
[22]	**Return** *Gbest* as the optimal solution

Fig. 2. The pseudocode of the Basic PSO

According to Fig.2, each particle is responsible for locating itself within the search space. Once locations of particles are identified, they will initiate a process of intercommunication to enable the exchange of information along the entirety of their trajectory. Within the PSO algorithm, the process of movement is influenced by an individual cognitive (personal best or $Pbest$) desire to search and the group's collective action (global best or $Gbest$). There exists a categorization of data aggregation into two distinct types: position and velocity. The updated velocity of each particle depends on several critical parameters: the inertia weight (W), the coefficient for the personal best (Cp), and the coefficient for the global best (Cg). This updated velocity consequently leads to the derivation of a new position of a particle, which is acquired through the best position attributed to each particle and that of the entire swarm. This methodology resides within the framework of evolutionary algorithms.

4.2. Differential Evolution (DE)

DE was introduced by Storn and Price in 1995. It is a population-based optimization algorithm and constitutes a member of the evolutionary algorithm's family. DE is frequently employed for addressing optimization and search challenges. The pseudocode of DE algorithms applied in this study is provided in Fig. 3.

Differential Evolution Algorithm (DE)

```
[1]  Set the following parameters: population size, number of iterations,
     Scale factor (F), Crossover rate (Cr), and Crossover type.
[2]  Initialize a population of vector with randomly
[3]  For t = 1 to the maximum iteration
[4]      For i = 1 to the population size:
[5]      Generate three distinct random parents (a, b, c) from the current population
         to generate a trail vector (u) by blending the solutions a, b, and c
[6]      For d = 1 to the total dimension
[7]          If a random number r from [0, 1] is less than CR or equal to d is randomly selected:
[8]              u[d] = a[d] + F * (b[d] - c[d])
[9]          Else:
[10]             u[d] = Target vector[d]
[11]         End if
[12]     End for
[13]     Decode to obtain the customer and the vehicle sequence
[14]     Consider the time window and capacity constraints
[15]     Evaluate objective value or fitness value
[16]     If the trial vector is better than the current candidate solution
[17]         Vector = u
[18]     Else:
[19]         Vector = Target vector
[20]     End if
[21]   End for
       Update global best vector.
[22]
[23] End for
[24] Return the best vector as the optimal solution
```

Fig. 3. The pseudocode of the Basic DE

According to Fig. 3, DE operates on the fundamental principle of a population comprising candidate solutions, called vectors. Each vector represents the algorithm's derived solution. Over successive generations, DE refine these candidate solutions by the use of a two-fold strategy which are mutation and crossover operations to construct novel trial solutions. Subsequently, these trial solutions undergo evaluation in alignment with the objective function's values. Should a trial solution prove superior, it supersedes the corresponding solution within the population; conversely, if it fails to outperform, it is disregarded. Customization and adaptation of DE to diverse problem domains are achieved through parameter adjustments such as population size, mutation strategies, and crossover methods. This study employs the fundamental DE variant known as DE/rand/1, wherein vectors are generated by introducing a weighted difference between randomly selected vectors and a third randomly chosen vector.

Not only mechanism of finding optimal solution, but solution representation is also required in order to apply metaheuristics to solve real-world problems. Solution representation involves encoding and decoding, essential stages that explain how problem solutions are transformed into a format understandable by the algorithm and how the algorithm's output is translated back into meaningful solutions. The description of the solution representation of MTCVRPTW is provided in the following section.

4.3. Solution Representation

In order to implement PSO and DE for solving delivery routing problem, the solution representation is required to transform real-number values of each dimension into a practical solution of the problem. This study designs a solution representation with encoding and decoding procedure, as shown in Fig.4.

The encoding process constitutes a critical step in the solution representation of vehicle routing problem. It begins with generating a string of dimensions, which consists of two parts: customer dimensions and vehicle dimensions. In, the number of dimensions is set as equal to the total number of customers and vehicle. Let's consider a scenario of a delivery routing problem with five customers and two vehicles, as shown in Fig. 4(a). In this context, the number of dimensions is calculated as 5 + 2 = 7, where the first five dimensions represent customers, and the last two dimensions represent two vehicles. Then, each value in a dimension is initially assigned a random number in range of [0, 1].

In this study, the decoding procedure is divided into three stages. First, the delivery sequence of customers is determined. Second, the sequence of assigned vehicles is considered. Finally, handling constraints, vehicle capacity and delivery time window, are thoroughly checked to guarantee feasible delivery routes.

Dimension	1	2	3	4	5	6	7
Random	0.17	0.14	0.87	0.19	0.97	0.73	0.17

Customers Vehicles

(a) Encoding process for 5 customers and 2 vehicles

Dimension	1	2	3	4	5	6	7
Random	0.17	0.14	0.87	0.19	0.97	0.73	0.17
Sequence	C2	C1	C4	C3	C5	V2	V1

Customers Vehicles

(b) Decoding process to determine sequence of customers and vehicles

Fig. 4. The procedure of encoding and decoding to obtain a solution

Consider the example of the routing problem with five customers and two vehicles; the first and the second stages of decoding procedure are depicted in Fig. 4(b). In the first stage, random values of five dimensions are sorted in ascending order to determine the delivery sequence of customers. According to the row of "Sequence" in Fig. 4(b), customer C1 is assigned to dimension 2 since the smallest dimension value, 0.14, is in dimension 2. Then, customer C2 assigned dimension 1 as the next smallest dimension value, 0.17, is in dimension 1. This process continues until all customers are assigned with sequence. Similarly, in the second stage, random values of two dimensions are sorted in ascending order to determine the sequence of assigned vehicles. As seen in Fig. 4(b), vehicle V1 is assigned to dimension 7 as the smallest value, 0.17, is in dimension 7, and vehicle V2 is assigned to dimension 6 as the next smallest value is in dimension 6. Consequently, sequence of customers is obtained as C2-C1-C4-C3-C5, and sequence of assigned vehicles is V2-V1 which means that vehicle V2 is the first choice to use for delivery and vehicle V1 is the next choice.

Even though the sequences of customers and assigned vehicles are attained, delivery constraints regarding to time window and vehicle capacity have not been considered in these two stages. In order to obtain a feasible solution, the third stage of decoding procedure, called constraint handling, is required. This stage involves checking conditions related to vehicle capacity and customer time window in order to allocate vehicles responsible for transportation and determine delivery trips. The description of constraint handling is provided in the following section.

Constraint-Handling

In this MTCVRPTW, two primary constraints are concerned: time window constraint and vehicles capacity constraint. Constraint handling procedure considers the specified constraints in order to allocate vehicle and determine vehicle trips for transportation.

The solution obtained from previous stages is input into constraint-handling process. The pseudocode of the constraint-handling algorithm is shown in Fig. 5. The main idea of constraint-handling is to sequentially consider a customer in a delivery order until all customers are delivered. While the algorithm iterates through each customer, it takes constraints of customer time window and vehicle capacity into account. The procedure of constraint-handling consists of three parts which are lower-bound constraints, upper-bound constraints, and capacity constraints.

- Lower-bound constraint

The lower bound constraint restricts that a vehicle must arrive at any location when it is opened. In cases where visiting time of a vehicle is less than the opening time, a special equation or penalty cost function will be applied, see equation (17). Where ξ is a real parameter within the interval [0, 1] allowing for flexibility regarding the acceptable arrival time of vehicles before $\boldsymbol{E_i}$ without incurring a constraint violation, and $\boldsymbol{P}$ represents an acceptable waiting time.

```
Initialize index of customer i , index of the trips r, index of vehicle k, weight load of vehicle
k, volume load of vehicle k, visiting time of vehicle k, distance of vehicle k.
While index of vehicle k <= total number of vehicles:
    While index of trip r <= total number of trips:
        While index of customer i <= total number of customer and index of vehicle k
              <= total number of vehicles:
            distance of vehicle k += distance from customer i to j
            visiting time of vehicle k += traveling time from customer i to j
            weight load of vehicle k += weight demand of customer i
            volume load of vehicle k += volume demand of customer i
            # Lowerbound constraint ;
            if visiting time of vehicle k < opening time of customer i:
                if opening time of customer i – visiting time of vehicle k
                > Acceptable waiting time:
                    Waiting time = 1
                Else:
                    Waiting time = 0
                Penalty cost += max (0, Waiting time*(opening time of customer i –
                                visiting time of vehicle k) * Waiting cost
                End if
            End if
            # Upperbound constraint ;
            if visiting time of vehicle k > opening time of customer i :
                Break while (3) for considering next vehicle k + 1 .
            End if
            # Capacity constraint ;
            if weight load of vehicle k > weight capacity of vehicle k or volume load
               of vehicle k > volume capacity of vehicle k:
                    distance of vehicle k  −= distance from customer i to j
                    visiting time of vehicle k −= traveling time from customer i to j
                    weight load of vehicle k  −= weight demand of customer i
                    volume load of vehicle k −= volume demand of customer i
                    vehicle k go back to depot and consider trip r + 1.
            End if
        End while (3)
        if index of customer i > total number of customers:
            Break while (2)
        End if
    End while (2)
    Cost = fixed cost of vehicle k + (distance of vehicle k * oil price of vehicle
           k/consumptionoil rate of vehicle k) + Penalty cost
    Total cost += Cost
    if index of customer i > total number of customers or index of vehicle k >
      Total number of total number of vehicles:
        Break while (1)
    End if
    End while (1)
```

Fig. 5. The pseudocode of constraint handling

$$\xi = \begin{cases} 0, & if\ E_i - s_{ikr} \le P \\ 1, & if\ E_i - s_{ikr} > P \end{cases}, \quad for\ all\ s_{ikr} < E_i \qquad (17)$$

The value of ξ depends on an acceptable waiting time (the time a vehicle can wait until the customer's opening time without violating constraints). The value of $\xi = 0$ when a vehicle arrives before the customer's opening time, and the waiting time until that location opens remains within an acceptable period. In this case, the constraint is not violated, resulting in a zero penalty. In contrast, $\xi = 1$ when a vehicle early arrives at customer location and waiting time exceeds an acceptable period. This violation results in a penalty cost being added to the objective function. It is noted that the penalty cost is obtained from equation (18).

$$Penalty\ cost = \max\{\ 0,\ \xi * \left(E_{i+1} - \left(s_{ikr} + h_i + t_{ij}\right)\right) * Waiting\ cost\} \qquad (18)$$

- Upper-bound constraint

The upper-bound constraint guarantees that a vehicle must visit any customer before the closing time. This constraint is used as a criterion for selecting the other vehicle for delivery. When the visiting time of a current vehicle exceeds the closing

time, the next vehicle in the sequence is considered for delivery.

- Capacity constraint

The capacity constraint means that no vehicle can load goods exceeding its capacity on any trip. In this study, a vehicle capacity includes both weight and volume capacity. If loads of any vehicle exceed its capacity on any trip, it requires the vehicle to return to the depot, load new goods, and perform a new trip for distribution. This process allows using the same vehicle for multiple trips until other vehicles need to be considered.

5. Results and discussion

This section presents the numerical experiment including dataset used in this study, parameters setting for PSO and DE algorithm, and experimental results. The performance of two metaheuristics is evaluated and compared with those obtained from an exact method and current practice. A computational experiment was run on a platform AMD Ryzen 7 5700U CPU processor. This processor operates at a clock speed of 1.80 GHz and is supported by 16 GB of RAM.

5.1. Datasets

The datasets used in this study are categorized into two types: generated data and current practice data. Instances 1 to 9 represent small and medium size problems using generated data. The purpose of using generated data is to validate and compare performance of metaheuristics against an exact method. However, real-world delivery problems are often in large scale. Thus, instances 10 to 30 employ current practice data in order to evaluate the performance of metaheuristics for real-world MTCVRPTW. The current practice data are obtained from historical delivery data of 21 operating days.

5.2. Parameters Setting for Metaheuristics

Not only searching mechanisms and solution mapping procedures that influence solution quality obtained from metaheuristics, but how the parameters are set also affects how well solutions converge. In this study, the number of function evaluations are set to 100,000 with the population size of 100 and iterations of 1,000. According to preliminary experiments, PSO and DE parameters used in this study are illustrated in Table 1 and Table 2, respectively.

Table 1. The configuration of PSO parameters

Parameter	Parameter Value
Swarm Size	100
Iterations	1000
Personal Best Coefficient (Cp)	0.8
Global Best Coefficient (Cg)	0.2
Inertia Weight (Linearly decrease)	0.9 – 0.5

According to PSO parameters in Table 1, when the value of Cp is greater than the value of Cg, it implies that the algorithm places more emphasis on each particle's personal best information when updating its position. In simpler terms, each particle is more influenced by its own historical best solution than by the best solution found by other particles in the swarm. This configuration tends to encourage particles to thoroughly explore their individual solution spaces before being guided by the global best solution. Furthermore, concerning the inertia weight, a linearly decreasing strategy was applied. This means that the algorithm initially allows broad exploration and then gradually shifts its focus towards exploiting promising regions, ultimately aiding in the convergence to high-quality solutions.

Table 2. The configuration of DE parameters

Parameter	Parameter Value
Swarm Size	100
Iterations	1000
Scale Factor (F, linearly decrease)	0.9 – 0.5
Crossover rate (Cr, linearly decrease)	0.5 – 0.1
Crossover Type	Binomial Crossover

According to DE parameters in Table 2, values of F and Cr were set to be linearly decrease. This implies that at the initial stages of the algorithm, when these parameters are relatively high, there is a strong emphasis on exploration. Thus, solutions are greatly influenced by the differences among existing solutions, and there is a probability of selecting traits from various individuals. This encourages a wide-ranging exploration of the solution space with the aim of discovering diverse regions. These settings promote this exploration during the mutation process by encouraging vectors to search broadly and providing a 50 percent chance of acceptance between the mutant vector and target vector. As the optimization process progresses, both parameters undergo linear decreases. This implies that the impact of differential variation and the probability of crossover become weaker over time. For example, when the crossover rate equals 0.1, it means that the algorithm accepts the target vector with a 90 percent probability and the mutant vector with a 10 percent probability. Consequently, the algorithm shifts its focus from exploration towards exploitation as the optimization process continues.

5.3. Computational Results

The metaheuristics have been implemented utilizing the Python programming language within Jupiter Lab version 3.2.9. The performance of metaheuristics is evaluated using 30 instances. As mentioned earlier, instances 1 to 9 represent small and medium size problems using generated data. Instances 10 to 30 are large-size problems with current practice data of 21 operating days. The problem is characterized by the number of customers and vehicles. The performance comparison between two metaheuristics; PSO and DE, are performed under the same conditions which are population size, numbers of iterations, and solution representation scheme.

Table 3 shows numerical results of the transportation costs obtained from the exact method, the current practice, and metaheuristics. For metaheuristics, the best, average, and standard deviation (SD) values from 10 replicated runs for each instance are reported.

Table 3. Comparison results of total transportation costs obtained from PSO and DE

Instance	Customers	Vehicles	Total Transportation Cost (Baht)										
			Current Practice	Optimal Solution	Time (Mins)	PSO			Time (Mins)	DE			Time (Mins)
						Best	Avg.	SD		Best	Avg.	SD	
1	8	2	-	1811	0.11	1811	1811	0.00	0.21	1811	1811	0.00	0.46
2	9	2	-	1816	0.06	1816	1816	0.00	0.19	1816	1816	0.00	0.47
3	10	2	-	1832	1.05	1832	1839	4.84	0.24	1832	1832	0.00	0.54
4	11	2	-	1838	1.55	1838	1912	111.43	0.20	1838	1838	0.00	0.52
5	12	2	-	1840	2.96	1840	1840	0.06	0.20	1840	1840	0.00	0.54
6	13	2	-	1843	50.79	1843	1846	5.23	0.28	1843	1843	0.00	0.68
7	14	2	-	1860*	> 600	1860	1863	2.98	0.63	1860	1860	0.00	1.51
8	15	2	-	1833*	> 600	1879	1881	2.31	0.23	1879	1879	0.00	0.62
9	16	2	-	1880*	> 600	1880	1889	7.25	0.22	1880	1880	0.09	0.47
10	22	2	2060	2039*	> 600	2068	2091	18.08	0.24	2040	2048	5.56	0.52
11	23	2	2120	2032*	> 600	2077	2132	39.64	0.24	2037	2048	14.02	1.20
12	24	2	2396	2142*	> 600	2293	2405	74.11	0.43	2182	2208	18.09	0.69
13	25	2	2196	2108*	> 600	2204	2254	45.66	0.26	2074	2103	19.73	0.56
14	27	2	2101	2064*	> 600	2173	2242	59.77	0.27	2106	2130	12.88	0.59
15	27	2	2287	2142*	> 600	2270	2408	114.06	0.34	2158	2178	17.51	0.56
16	27	2	2176	2071*	> 600	2162	2247	82.58	0.26	2087	2103	21.25	1.38
17	27	2	2151	2062*	> 600	2159	2283	68.74	0.27	2094	2115	17.46	1.46
18	29	2	2277	2093*	> 600	2231	2306	60.40	0.38	2083	2146	28.88	1.43
19	31	2	2141	2079*	> 600	2242	2314	39.03	0.30	2119	2139	15.73	0.60
20	31	2	2316	2179*	> 600	2338	2459	70.51	0.32	2220	2256	20.66	0.60
21	31	2	2316	2189*	> 600	2338	2472	105.45	0.36	2211	2252	27.67	1.39
22	31	2	2176	2045*	> 600	2122	2234	58.92	0.38	2062	2100	14.40	0.60
23	32	2	2318	2239*	> 600	2440	2842	673.97	0.33	2272	2312	20.97	0.65
24	32	2	2355	2258*	> 600	2344	2506	90.61	0.39	2284	2308	13.70	0.61
25	32	2	2346	2078*	> 600	2220	2319	59.60	0.42	2089	2134	23.07	0.61
26	35	2	2287	2176*	> 600	2320	2498	117.73	0.40	2240	2287	27.92	0.67
27	36	2	2544	2215*	> 600	2251	2415	75.19	0.42	2207	2235	11.88	0.69
28	36	2	2229	2124*	> 600	2350	2467	59.44	0.44	2179	2239	31.42	0.67
29	36	2	2469	2302*	> 600	2502	2702	128.48	0.42	2397	2436	29.63	0.66
30	39	2	2278	2139*	> 600	2355	2469	88.61	0.37	2227	2262	28.78	0.72

*A solution obtained from an exact method within a span of 10 hours

Based on the results in Table 3, it is evident that, for instance 1 to 6 when the number of customers is up to 13, the exact method easily found the global solution with fast computing time. Similarly, both PSO and DE were able to find the optimal solutions for instances 1 to 6, but their potential differed. DE consistently generated a solution with an SD equal to zero for almost all instances of this size. On the other hand, PSO provides higher SD values to DE for all instances. This indicates that DE outperforms PSO in terms of stability and reliability. It is also noteworthy to mention that, even in instances where the exact method faced difficulties, PSO and DE exhibited robust performance in finding the globally optimal solution.

For the larger instances, when the number of customers is greater than 13, the exact method struggles to find the optimal solutions within a 10-hour time frame. This limitation of the exact method emphasizes the crucial role played by metaheuristics. As shown in Table 3, both PSO and DE demonstrated outstanding performance in generating solutions with lower transportation costs compared to current practices in most of instances. Particularly, the results indicated that DE generally outperforms PSO since the SD values obtained from DE are than those from PSO. This implies that DE can generate solutions that are more consistent and tightly clustered around the average value. In terms of computational time, both algorithms exhibit rapid processing capabilities for finding solutions, with PSO demonstrating a litter faster search capability than DE

To further evaluate the performance of metaheuristics against an exact method and current practice method, this study utilized the Percentage Deviation equation (PD), as shown in equation (19) and (20), respectively. It is noted that $Sol_{optimal}$ represents a solution obtained from an exact method, $Sol_{practice}$ represents a solution obtained through practical methods, and $Sol_{metaheuristic}$ represents a solution derived from the metaheuristic approach.

According to equations (19) and (20), a positive value indicates that the metaheuristic generates a solution worse than the one under comparison. Conversely, a negative value denotes

that the metaheuristic provides a superior solution compared to the one used for comparison. The comparison results of PD value are shown in Table. 4.

$$PD = \left(\frac{(Sol_{metaheuristic} - Sol_{optimal})}{Sol_{optimal}}\right) \times 100 \quad (19)$$

$$PD = \left(\frac{(Sol_{metaheuristic} - Sol_{practice})}{Sol_{practice}}\right) \times 100 \quad (20)$$

According to Table 4, the superior performance of DE over PSO is further emphasized. Even though both PSO and DE demonstrated their ability to find optimal solutions for instances 1 to 6, as the problem complexity increases, DE outperformed PSO. In particular, when comparing results between metaheuristics and the exact method, PSO yielded PD values ranged from 10.64% to 0.00%, with an average of 4.26%. While DE shows lower PD values ranged from 4.13% to -1.61%, with an average of 1.00%. Similarly, when comparing results between metaheuristics and the current practice, PSO provides PD values ranged from 5.41% to -11.52%, with an average of -0.07%, whereas DE demonstrated lower PD values ranged from 0.23% to -13.24%, with an average of -4.45%. These results underline superior performance of DE

Table 4. Comparison results of PD value

Total Transportation Cost (Baht)										
Instance	Customer	Vehicles	Optimal Solution	Current Practice	PSO			DE		
					Best	PD		Best	PD	
						Optimal Solution	Current Practice		Optimal Solution	Current Practice
1	8	2	1811	-	1811	0.00	-	1811	0.00	-
2	9	2	1816	-	1816	0.00	-	1816	0.00	-
3	10	2	1832	-	1832	0.00	-	1832	0.00	-
4	11	2	1838	-	1838	0.00	-	1838	0.00	-
5	12	2	1840	-	1840	0.00	-	1840	0.00	-
6	13	2	1843	-	1843	0.00	-	1843	0.00	-
7	14	2	1860*	-	1860	0.00	-	1860	0.00	-
8	15	2	1833*	-	1879	2.51	-	1879	2.51	-
9	16	2	1880*	-	1880	0.00	-	1880	0.00	-
10	22	2	2039*	2060	2068	1.46	0.42	2040	0.06	-0.97
11	23	2	2032*	2120	2077	2.17	-2.07	2037	0.20	-3.96
12	24	2	2142*	2396	2293	7.05	-4.29	2182	1.85	-8.94
13	25	2	2108*	2196	2204	4.56	0.37	2074	-1.61	-5.56
14	27	2	2064*	2101	2173	5.27	3.44	2106	2.00	0.23
15	27	2	2142*	2287	2270	5.99	-0.72	2158	0.75	-5.63
16	27	2	2071*	2176	2162	4.36	-0.64	2087	0.78	-4.06
17	27	2	2062*	2151	2159	4.70	0.35	2094	1.57	-2.64
18	29	2	2093*	2277	2231	6.57	-2.03	2083	-0.48	-8.51
19	31	2	2079*	2141	2242	7.87	4.72	2119	1.95	-1.02
20	31	2	2179*	2316	2338	7.32	0.94	2220	1.90	-4.16
21	31	2	2189*	2316	2338	6.82	0.97	2211	1.03	-4.51
22	31	2	2045*	2176	2122	3.77	-2.49	2062	0.85	-5.23
23	32	2	2239*	2318	2440	9.00	5.29	2272	1.47	-1.99
24	32	2	2258*	2355	2344	3.82	-0.44	2284	1.16	-2.99
25	32	2	2078*	2346	2220	6.83	-5.37	2089	0.53	-10.94
26	35	2	2176*	2287	2320	6.60	1.43	2240	2.93	-2.06
27	36	2	2215*	2544	2251	1.60	-11.52	2207	-0.37	-13.24
28	36	2	2124*	2229	2350	10.64	5.41	2179	2.62	-2.23
29	36	2	2302*	2469	2502	8.68	1.35	2397	4.13	-2.90
30	39	2	2139*	2278	2355	10.10	3.41	2227	4.11	-2.22
Average						4.26	-0.07		1.00	-4.45

*A solution obtained from an exact method within a span of 10 hours

over PSO, particularly in scenarios of high problem complexity.

Furthermore, the comparison of convergence behavior between PSO and DE is visually demonstrated using four largest instances in this study, as shown in Fig. 6.

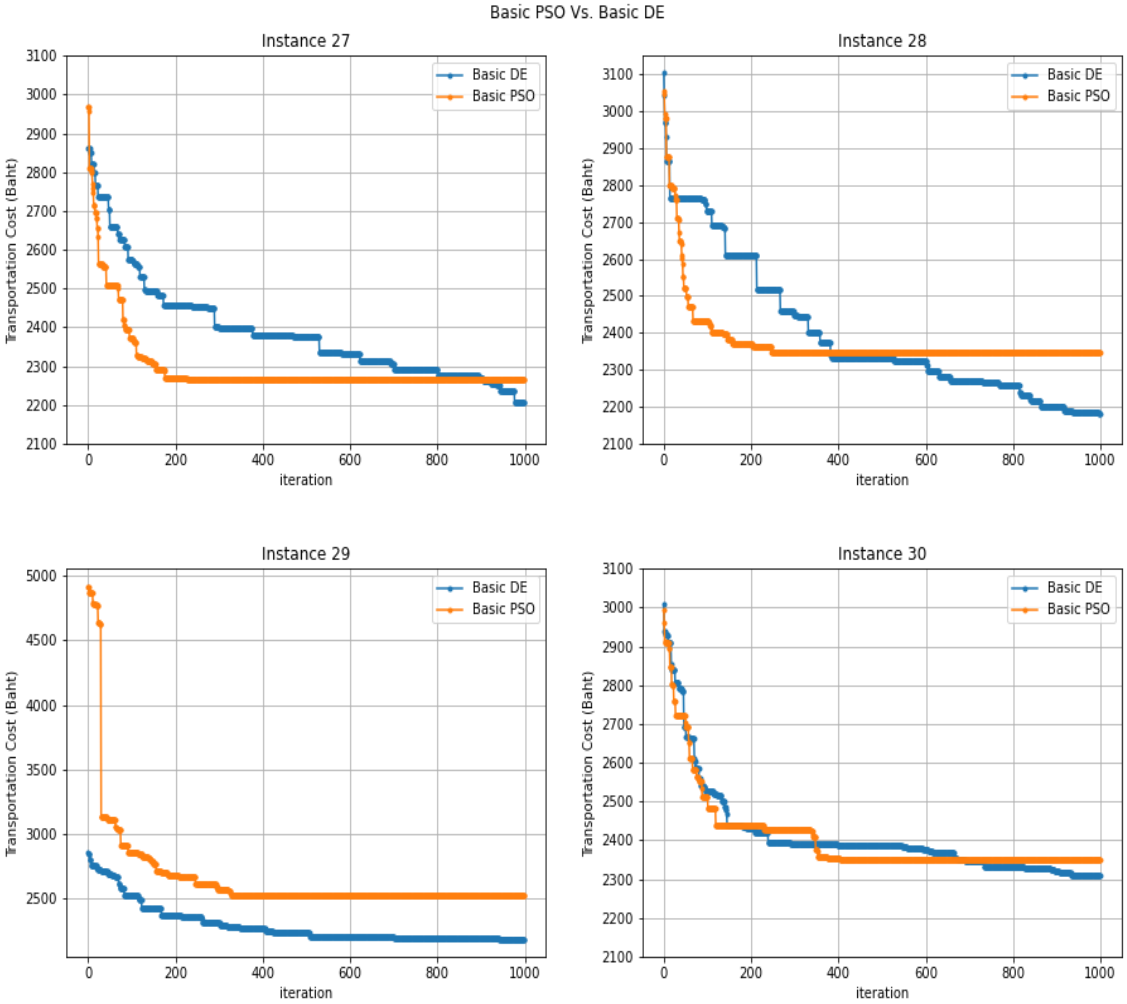

Fig. 6. Comparing of convergence behaviour between PSO and DE algorithm

As shown in Fig. 6., PSO quickly searched for good solutions during the initial iterations. However, as the search progressed, PSO often encountered challenges in escaping local optimal solutions. On the other hand, using the same number of function evaluations, DE showed its gradual improvement in generating better solutions compared to PSO.

6. Conclusion

This study presents an empirical study for concept of solving a delivery routing problem using a Mixed Integer Programming model for the multi-trip capacitated vehicle routing problem with time window (MTCVRPTW) to minimize total transportation cost. However, since MTCVRPTW is classified as an NP-hard problem, the exact method is considered inappropriate as it can only find optimal solutions for small-scale problems. In addition, it encounters difficulties in locating optimal solutions for real-world problems which often involves larger-scale problems. For this reason, a metaheuristic is more suitable for solving large-scale MTCVRPTW in real-world practices.

This study thus presents application of two metaheuristic algorithms, namely, Particle Swarm Optimization (PSO), and Differential Evolution (DE) for solving the MTCVRPTW. Using a real case of transport company in Thailand, the experimental results demonstrate that DE is more effective in providing cost-effective solutions compared to PSO. The comparison results also reveal that DE can significantly reduce transportation costs of company's current practice. Even though PSO slightly consumes less computing time than DE when using the same number of population size and iterations, DE can finally achieve better solution quality. Based on convergence behavior, DE also illustrate its superior ability to search for better solution. These results can be attributed to the fact that the search algorithm in PSO is mostly relied on a global best solution. Consequently, solution can be quickly

trapped in local optimum and escaping from that local optimum can be very difficult. In contrast, DE algorithm allows. each vector to continuously search for solution more independently. Thus, the chance of DE trapping in local optimum is lessened.

For further study, some other constraints in real-world scenarios may be considered. For examples, the sequence of loading and unloading parcels can be included in the problem. The performance evaluation among other metaheuristic algorithms, such as Ant Colony Algorithms, Large Neighborhood Search, or Firefly Algorithms are also noteworthy to investigate.

Acknowledgement

This research was partially funded by Chiang Mai University, Thailand. This research is a part of the project "A Strategic Roadmap Toward the Next Level of Intelligent, Sustainable, and Human-Centered SME: SME 5.0" from the European Union's Horizon 2021 research and innovation program under the Marie Skłodowska-Curie Grant agreement No. 101086487.

Reference

Banomyong, R., Grant, D.B., Varadejsatitwong, P. and Julagasigorn, P., 2022. Developing and validating a national logistics cost in Thailand. Transport Policy, 124, 5-19. DOI:10.1016/j.tranpol.2021.04.026

Cattaruzza, D., Absi, N., Feillet, D., 2016. Vehicle routing problems with multiple trips. 4or, 14, 223-259. DOI:10.1007/s10288-016-0306-2

Clarke, G., Wright, J. W.,1964. Scheduling of vehicles from a central depot to a number of delivery points. Operations research, 12(4), 568-581. DOi:10.1287/opre.12.4.568

Du, L., He, R., 2012. Combining nearest neighbor search with tabu search for large-scale vehicle routing problem. Physics Procedia, 25, 1536-1546. DOI:10.1016/j.phpro.2012.03.273

Dumez, D., Lehuédé, F. and Péton, O., 2021. A large neighborhood search approach to the vehicle routing problem with delivery options. Transportation Research Part B: Methodological, 144, 103-132, DOI: 10.1016/j.trb.2020.11.012

Eraslan, E., Derya, T., 2010. Daily newspaper distribution planning with integer programming: an application in Turkey. Transportation Planning and Technology, 33(5), 423-433. DOi: 10.1080/03081060.2010.502374

Geetha, S., Vanathi, P. T., Poonthalir, G., 2012. Metaheuristic approach for the multi-depot vehicle routing problem. Applied Artificial Intelligence, 26(9), 878-901. DOI: 10.1080/08839514.2012.727344

Hernandez, F., Feillet, D., Giroudeau, R., Naud, O., 2014. A new exact algorithm to solve the multi-trip vehicle routing problem with time windows and limited duration. 4or, 12, 235-259. DOI: 10.1007/s10288-013-0238-z

Hu, F., Wu, F., 2010. Diploid hybrid particle swarm optimization with differential evolution for open vehicle routing problem. 2010 8th World Congress on Intelligent Control and Automation, Jinan, 2692-2697.

Kumari, M., De, P. K., Chaudhuri, K., Narang, P., 2023. Utilizing a hybrid metaheuristic algorithm to solve capacitated vehicle routing problem. Results in Control and Optimization, 13, 100292. DOI: 10.1016/j.rico.2023.100292

Kachitvichyanukul, V., 2012. Comparison of three evolutionary algorithms: GA, PSO, and DE. Industrial Engineering and Management Systems, 11(3), 215-223. DOI: 10.7232/iems.2012.11.3.215

Lyu, J., He, Y., 2021. A two-stage hybrid metaheuristic for a low-carbon vehicle routing problem in hazardous chemicals road transportation. Applied Sciences, 11(11), 4864. DOI: 10.3390/app11114864

Maffioli, F., 2003. The vehicle routing problem: A book review. Quarterly Journal of the Belgian, French and Italian Operations Research Societies, 1(2), 149-153. DOI: 10.1007/s10288-003-0013-7

Neira, D. A., Aguayo, M. M., De la Fuente, R., Klapp, M. A., 2020. New compact integer programming formulations for the multi-trip vehicle routing problem with time windows. Computers & Industrial Engineering, 144, 106399. DOI: 10.1016/j.cie.2020.106399

Marinakis, Y., Iordanidou, G. R., Marinaki, M., 2013. Particle swarm optimization for the vehicle routing problem with stochastic demands. Applied Soft Computing, 13(4), 1693-1704. DOI: 10.1016/j.asoc.2013.01.007

MirHassani, S. A., Abolghasemi, N., 2011. A particle swarm optimization algorithm for open vehicle routing problem. Expert Systems with Applications, 38(9), 11547-11551. DOI: 10.1016/j.eswa.2011.03.032

Moghaddam, B. F., Ruiz, R., Sadjadi, S. J., 2012. Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm. Computers & Industrial Engineering, 62(1), 306-317. DOI: 10.1016/j.cie.2011.10.001

Niyomphon, K., Wisittipanich, W., 2022. A Mathematical Model for Multi-Trip Vehicle Routing Problem with Time Window in Transportation Business. ICLS2022 16th International Congress on Logistics and SCM System, Khon Kaen, Thailand, 94

Tang, J., Yu, Y., Li, J., 2015. An exact algorithm for the multi-trip vehicle routing and scheduling problem of pickup and delivery of customers to the airport. Transportation Research Part E: Logistics and Transportation Review, 73, 114-132. DOI: 10.1016/j.tre.2014.11.001

Tirkolaee, E. B., Alinaghian, M., Hosseinabadi, A. A. R., Sasi, M. B., Sangaiah, A. K., 2019. An improved ant colony optimization for the multi-trip Capacitated Arc Routing Problem. Computers & Electrical Engineering, 77, 457-470. DOI: 10.1016/j.compeleceng.2018.01.040

Tirkolaee, E. B., Hosseinabadi, A. A. R., Soltani, M., Sangaiah, A. K., Wang, J., 2018. A hybrid genetic algorithm for multi-trip green capacitated arc routing problem in the scope of urban services. Sustainability, 10(5), 1366. DOI: 10.3390/su10051366

Theurich, F., Fischer, A., Scheithauer, G., 2021. A branch-and-bound approach for a Vehicle Routing Problem with Customer Costs. EURO Journal on Computational Optimization, 9, 100003. DOi: 10.1016/j.ejco.2020.100003

Utamima, A., Pradina, K. R., Dini, N. S., Studiawan, H., 2015. Distribution route optimization of gallon water using genetic algorithm and tabu search. Procedia Computer Science, 72, 503-510. DOI: 10.1016/j.procs.2015.12.132

Wisittipanich, W., Phoungthong, K., Srisuwannapa, C., Baisukhan, A., Wisittipanit, N., 2021. Performance comparison between particle swarm optimization and differential evolution algorithms for postman delivery routing problem. Applied Sciences, 11(6), 2703. DOI: 10.3390/app11062703

Wisittipanit, N., Baisukhan, A., Srisuwannapa, C., 2021. Comparisons of VRP Optimization Algorithmic Methods for the Optimal Routing of Multiple Delivery Vehicles with Time Constraint. International Journal of Engineering Sciences, 13(4), 131-140. DOI: 10.36224/ijes.130401

Xu, S. H., Liu, J. P., Zhang, F. H., Wang, L., Sun, L. J., 2015. A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows. Sensors, 15(9), 21033-21053. DOI: 10.3390/s150921033

Yağmur, E., Kesen, S. E., 2021. Multi-trip heterogeneous vehicle routing problem coordinated with production scheduling: Memetic algorithm and simulated annealing approaches. Computers & Industrial Engineering, 161, 107649. DOI: 10.1016/j.cie.2021.107649

Yeh, W. C., Tan, S. Y., 2021. Simplified swarm optimization for the heterogeneous fleet vehicle routing problem with time-varying continuous speed function. Electronics, 10(15), 1775. DOI: 10.3390/electronics10151775

Zhen, L., Ma, C., Wang, K., Xiao, L., Zhang, W., 2020. Multi-depot multi-trip vehicle routing problem with time windows and release dates. Transportation Research Part E: Logistics and Transportation Review, 135, 101866. DOI: 10.1016/j.tre.2020.101866