



University of
Pittsburgh

PITT

SWANSON
ENGINEERING

INDUSTRIAL ENGINEERING
SUMMER 2021


TRANSPORTATION PLANNING/ IE 2045
BY
PROF. NATASA VIDIC

Research Paper review

SUBMITTED BY

SABELLA PRASANNA
4450873
prs98@pitt.edu

Metaheuristic algorithm for solving the multi-objective vehicle routing problem with time window and drones

Yun-qi Han¹, Jun-qing Li^{1,2} , Zhengmin Liu³, Chuang Liu² and Jie Tian¹

1.1 Background

- A special case of a Vehicle routing problem is to transport needed goods using drones because of the inaccessible landform.
- This is a Vehicle routing problem with multiple objectives and time window and the vehicles also include the drones for vertical transportation. (MO-VRPTW-D)
- Most helpful in natural disaster scenarios such as landslides and flood relief.
- The authors considered different types of vehicles based on their carrying capacity and marked demands for every customers.
- Energy consumption of these vehicles including the drones is also different.
- Each truck can only carry one drone.
- Customer's location is first addressed as a 2D point on a plane and then the perpendicular height is taken into account.

Problem statement is to find out the best possible combinations of these trucks and drones such that the total energy consumption (\propto travel distance) is minimum without violating the time window margins.

1.2 Variables and terminology

- Customer set $C = \{1, 2, \dots, n\}$
- Horizontal distance between customer i and j , $A_h = \{i, j \mid i, j \in C\}$
- Perpendicular distance at a customer's location, $A_p = \{i_b, i_t\}$
- Vehicles set $V = \{1, 2, \dots, v\}$
- Drones set $D = \{d_1, d_2, \dots, d_v\}$
- $x_{ijk} = 1$ if vehicle k travels from i to j , else 0
- $y_i = 1$ if vertex i is served by vehicle k
- yd_k = Speed of the drone
- s_i = Serving duration
- wt_k = Energy consumption coefficient of truck k
- wd_k = Energy consumption coefficient of drone k
- t_{ij} = Travelled distance from i to j (0 if $x_{ijk} = 0$)
- tp_i = Travelled height by drone at i
- z_i = Arrival time of truck k at customer i
- $[e_i, l_i]$ = Time limit window at vertex i
 $e_0 = 0$ and $l_0 = 0$
- $[e_i, l_i]$ = Time limit window at vertex i
- w_i = wait time if $z_i < e_i$

2.1 Objective function and constraints

weighted sum of the total energy consumption of the trucks

weighted sum of the total energy consumption of the drones

Total number of the trucks/drones

$$\min \alpha \sum_{i=0}^n \sum_{j=0}^n \sum_{k=1}^v x_{ijk} wt_k t_{ij} + \beta \sum_{i=1}^n \sum_{k=1}^v x_{0jk} 2tp_i \frac{1}{yd_k} wd_k + \gamma \sum_{k=1}^v x_{0jk}, i \neq j, i, j \in N, k \in V$$

$$\sum_{i=1}^n dm_i y_{ik} \leq bm_k, \forall k \in V$$

Customer's main demands do not exceed truck's main capacity

$$\sum_{i=1}^n ds_i y_{ik} \leq bs_k, \forall k \in V$$

Customer's supplementary demands do not exceed truck's supplementary capacity

$$\sum_{k=1}^v y_{ik} = 1, \forall i \in C$$

Every customer sees only one truck

$$\sum_{i=0}^n x_{ijk} = y_{jk}, \forall k \in V, \forall j \in C$$

No truck visits the same customer twice

$$\sum_{j=0}^n x_{ijk} = y_{jk}, \forall k \in V, \forall i \in C$$

A truck visits no customer twice

$$\sum_{j=1}^n x_{0jk} - \sum_{i=1}^n x_{i0k} = 0, \forall k \in V$$

Every truck starts and ends at the depot

$$s_i = 2tp_i \frac{1}{yd_k}, \forall i \in C, \forall k \in V$$

Time = (Distance/Speed); Serving duration

$$\sum_{i=0}^n \sum_{j=0}^n x_{ijk} (t_{ij} + s_i + w_i) \leq r_k, \forall k \in V$$

Maximum trip duration does not exceed the limit

$$w_0 = s_0 = 0$$

No wait time or service time is expected at the depot

$$\sum_{k=1}^v \sum_{i=0}^n x_{ijk} (z_i + w_i + s_i + t_{ij}) = z_j, \forall j \in C$$

Departure time at i should be arrival time at j (also includes travel time)

$$e_i \leq z_i + w_i \leq l_i, \forall i \in C$$

Arrival time and wait time should be within the time window

$$w_i = \max\{0, e_i - z_i\}, \forall i \in C$$

Wait time is equal to expected arrival time – actual arrival time

$$x_{ijk} \in \{0, 1\}, \forall i, j \in C, \forall k \in V$$

x_{ijk} can only take values of 0 and 1

$$y_{ik} \in \{0, 1\}, \forall i \in C, \forall k \in V$$

y_{ik} can only take values of 0 and 1

$$z_i \geq 0, \forall i \in C$$

Arrival times cannot be negative

2.2 Methodology

The **artificial bee colony (ABC) algorithm** was first proposed by Author Karaboga to solve VRP. Author of our paper has modified this ABC algorithm by inducing an **Enhanced employed bee strategy and scout bee strategy**. Authors also induced repair strategy and Initialization strategy. This new modified strategy is named as **Improved Artificial Bee Colony (IABC) algorithm**.

2.3 The IABC algorithm

Input: the datas of customers, trucks and drones

Output: the best solution X^*

1 generate P_s initial solutions X by the strategy in sub-section 3.4

2 **employed bee phase**

for each solution x do

3 generate a new solution x^* by the strategy in sub-section 3.5 for solution x

4 if x^* is better than x

5 replace x with x^*

6 update the best solution that has been found so far to x^*

7 update the local best to x

8 end

9 end

10 **onlooker bee phase**

for each solution x do

11 randomly select a solution x_r

12 implement the Employed bee phase on the best solution between x and x_r

13 update the best solution X^*

14 end

15 **scout bee phase**

implement the Scout bee strategy for the best solution X^*

16 update the iterative time T of the IABC algorithm

17 update iteration times without improvement I_x for each solution x

18 if I_x exceeds the maximum no- improvement time L_n

19 implement the Scout bee strategy for the solution x'

20 if T not exceeds the T_{\max}

21 go to step 2

The **Repair Strategy** replaces infeasible solution with feasible ones.

- The traditional employed bee strategy chooses customers randomly of certain route, then re-arrange these customers to other routes.
- Easy to accomplish but hard to reach to an improved solution.
- Thus, use **Enhanced employed bee strategy** which chooses customers not randomly but based on **Push Forward Insertion Heuristic (PFIH)**

- The **Scout bee strategy** makes large changes to the solution.
- Executes a local search to insert some customers of solution
- Replace the truck of this solution with another new one.

3. Evaluation

The author evaluates the effectiveness of changes made in the employed bee phase and scout bee phase by using 55 benchmarked Solomon instances which denotes the deviation of the objective value from the best value for every method or algorithm. He evaluates using hypothetical ANOVA testing that the best performing algorithm is the one that holds the most 'best objective function values' and which on an average deviates less from the best value.

Employed bee	Before	After	Scout bee phase	Before	After
Total best	20	35	Total best	18	37
Mean deviation	2.3	0.87	Mean deviation	2.41	1.09

4. Results and conclusion

The author compares the IABC with TS, GA & VNS algorithms again by using the 55 benchmarked Solomon instances of the vehicle routing problem. IABC holds the greatest number of best solutions (=38) (70%). With this the author concludes that IABC performs better off when compared to other three algorithms. TS & VNS are close. GA is the worst performer.

	Objective value					Deviation			
Instances	Best	GA	TS	VNS	IABC	GA	TS	VNS	IABC
Inst1	1038.78	1571.34	1306.84	1351.27	1038.78	532.56	268.06	312.49	0
Inst2	1001.63	1275.01	1163.9	1125.35	1001.63	273.38	162.27	123.72	0
Inst3	950.11	1134.35	950.11	980.92	966.26	184.24	0	30.81	16.15
Inst4	956.58	1217.11	981.48	1011.2	956.58	260.53	24.9	54.62	0
Inst5	1010.41	1313.63	1108.59	1112.75	1010.41	303.22	98.18	102.34	0
.
.
.
Inst51	544.7	627.43	564.02	569.84	544.7	82.73	19.32	25.14	0
Inst52	479.69	511.9	485.82	479.69	483.92	32.21	6.13	0	4.23
Inst53	826.79	912.19	844.77	826.79	842.6	85.4	17.98	0	15.81
Inst54	626.18	658.18	627.59	626.18	631.6	32	1.41	0	5.42
Inst55	565.48	707.02	589.38	688.58	565.48	141.54	23.9	123.1	0
Mean		931.25	841.23	852.65	791.37				
Total best		0	09	09	38				

5. Critique

Though the energy consumption coefficient of the trucks (wt_k) and the drones (wd_k) mostly depend on their engine technology and its delivered speed, we are only limited to optimize by choosing different vehicles/drones and by selecting different routes. One drone on one truck constraint is missing. No where in the paper did the author mention what trucks are chosen with what drones. Bias of the output result due to unknown weights: The weights (α , β & Γ) in the objective function are very critical to the solution.

The author has compared an improved version of ABC with raw versions of TS, GA and VNS which is not a fair comparison and I doubt if IABC algorithm would be better than improved versions of these algorithms. As mentioned previously that the energy coefficients mostly depends on the engine speed/performance the author admits that constraints such as traffic intensity, accident situations, road/terrain and weather conditions are necessary. However, modifications to algorithm contributed well for the betterment of the solution.

An Efficient Genetic Algorithm for Large Scale Vehicle Routing Problem Subject to Precedence Constraints

Noraini Mohd Razali^{a*}

6.1 Background

- Vehicle routing problem with only one vehicle and precedence constraints is Travelling Salesman Problem (TSP) with precedence constraints.
- The special case of this problem is that the vehicle need not return to the depot.
- Real life vehicle routing problems are usually so large that exact methods such as TS, GA cannot be used to solve them as many heuristic problems do not seek for a global minima but a local one.
- Hence use a metaheuristic approach.
- The authors considers one vehicle available at the depot and a set of n customers $C = \{0, 1 \dots n\}$ and the vehicle need not report back to the depot.
- This is also known as Open Vehicle Routing Problem with Precedence Constraints. (VRP-PC)
- **Problem/Goal statement:** To find the most optimal route of delivery by the vehicle without violating the precedence constraints

6.2 Variables and terminology

- n = Number of nodes
- $n-1$ = Number of nodes excluding the depot
- C_{ij} = Distance to cover in transporting from i to j
- x_{ij}^p = Quantity of commodity p from i to j
- x_{ij}^q = Quantity of commodity q from i to j
- $x_{ij} = 1$, if j is visited immediately after i , else 0
- t_{ij} = cost associated on travelling/delivering from route i to j

7.1 Objective function and constraints

The objective function is to minimize the total distance travelled upon total number of nodes.

$$\text{Minimise } \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j}}^n \frac{1}{(n-1)} c_{ij} (x_{ij}^p + x_{ij}^q)$$

$$\sum_{j=1}^n x_{ij}^p - \sum_{j=1}^n x_{ji}^p = \begin{cases} n-1 & \text{for } i = s, \\ -1 & \text{otherwise,} \end{cases} \quad \left. \vphantom{\sum_{j=1}^n} \right\} \text{Flow of commodity p is feasible (No backflow in the route)}$$

$$\sum_{j=1}^n x_{ij}^q - \sum_{j=1}^n x_{ji}^q = \begin{cases} -(n-1) & \text{for } i = s, \\ +1 & \text{otherwise,} \end{cases} \quad \left. \vphantom{\sum_{j=1}^n} \right\} \text{Flow of commodity q is feasible (No backflow in the route)}$$

$$\sum_{j=1}^n (x_{ij}^p + x_{ij}^q) = n-1 \quad \forall i, \quad \text{Makes sure no node is left unvisited}$$

$$x_{ij}^p + x_{ij}^q = (n-1)x_{ij} \quad \forall i \text{ and } j, \quad \text{If } x_{ij} = 1 \text{ then sum of commodities between } i \text{ and } j \text{ is } (n-1)$$

$$\sum_{j=1}^n x_{uj}^p - \sum_{j=1}^n x_{vj}^p \geq 1 \quad \text{for } (v_u \rightarrow v_v)(v_v \neq s) \quad \text{For precedence relationships between vertices}$$

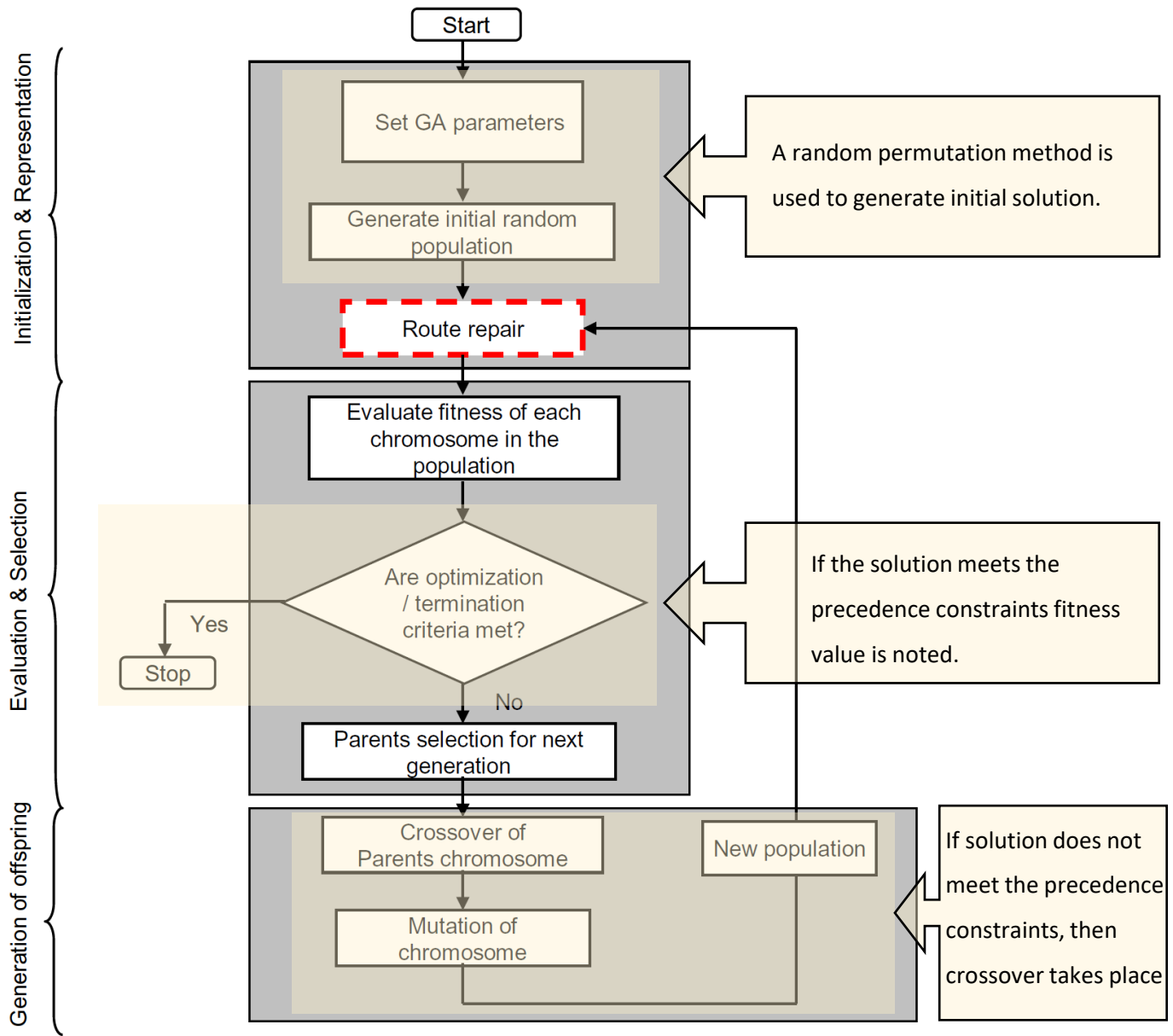
$$\left. \begin{array}{l} x_{ij}^p \geq 0 \quad \forall i \text{ and } j, \\ x_{ij}^q \geq 0 \quad \forall i \text{ and } j, \end{array} \right\} \text{Quantity of commodities cannot be negative}$$

$$x_{ij} \in \{0,1\} \quad \forall i \text{ and } j. \quad x_{ij} \text{ can only take binary values}$$

7.2 Methodology

- Genetic Algorithms are inspired by the theory of natural selection by Charles Darwin.
- A population of **individuals** are developed by means of a **chromosome** crossover (**reproduction**) and population with a lower **fitness value** are removed in every iteration.
- The author compares individuals to **nodes**, chromosomes to **solutions**, reproduction to the **repair strategy** and fitness value to the **objective function value**.
- The author names this new algorithm **GAnew** as she introduces a new repair strategy.
- Repair strategy in GAnew algorithm was inspired by a similar type of strategy used by Author Moon (**GAold algorithm**).
- ‘earliest position’ based topological sorting has been used in GAold’s repair strategy & ‘priority’ based topological sorting has been used in GAnew’s repair strategy.

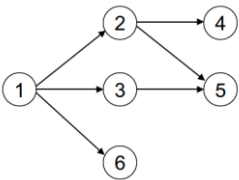
7.3 The GAnew algorithm



The Repair strategy of GAnew

Let us consider a randomly generated solution route as { 4 1 3 6 5 2 }

Violation: 4 has to be preceded by 2 and 5 has to be preceded by 2 & 3.



available set	updated sequence
1	[1]
2, 3, 6	[1 3]
2, 6	[1 3 6]
2	[1 3 6 2]
4, 5	[1 3 6 2 4]
5	[1 3 6 2 4 5]

Choosing the node that appears first in the initial random solution and without violating the precedence rule.

{1 3 6 2 4 5} is a feasible solution and fitness value is recorded.

8. Evaluation

The author compares GAnew and GAold algorithms based on two setups, 51 locations (customers), 71 precedence constraints and 100 locations (customers), 141 precedence constraints. Number of solution possibilities with 51 locations is $51! = 1.5511188e+66$. Author considers popsize (population size) = 500,1000 possibilities, Probability of crossover (P_c) = 0.6,0.9 and Probability of mutation (p_m) = 0.001, 0.2. This boils down to 8 experiments that must be performed and the best of these 8 are presented in the table below.

9. Results and conclusion

Test Problem	GAnew			GAold		
	Gen#	Best	Convergence time (sec)	Gen#	Best	Convergence time (sec)
51 tasks & 71 PC	193	184	650	163	256	1109
100 tasks & 141 PC	174	441	928	381	559	1240

Summary of results for all tests using GAnew and GAold algorithm

Based on the above results the author concludes that GAnew is performing better than GAold. Its not just the objective function value (fitness value) that was better in GAnew but also the convergence time.

10. Critique

Author did not mention how popsize = 500,1000 possibilities was chosen. Probability of crossover (P_c) and mutation (p_m) are very critical in deciding the best objective function value and in the analysis, these values are just fixed to Probability of crossover (P_c) = 0.6,0.9 and Probability of mutation (p_m) = 0.001, 0.2.

The only difference between the GAold and GAnew algorithm's repair strategy was that in GAnew's repair strategy the node that appears first in the randomly generated initial solution is chosen to deliver while the GAold's repair strategy chooses nodes based on delivery priority and the author claims that this small difference in node choosing strategy has made quite a significant difference in objective function. The mechanism behind how exactly this subtle change in node choosing caused such a big difference in the fitness value is not explained by the author but however it is still skeptic if GAnew would be better than GAold with a different parameter setting.

Metaheuristic algorithm for solving the multi-objective vehicle routing problem with time window and drones

Yun-qi Han¹, Jun-qing Li^{1,2} , Zhengmin Liu³, Chuang Liu² and Jie Tian¹

Abstract

In some special rescue scenarios, the needed goods should be transported by drones because of the landform. Therefore, in this study, we investigate a multi-objective vehicle routing problem with time window and drone transportation constraints. The vehicles are used to transport the goods and drones to customer locations, while the drones are used to transport goods vertically and timely to the customer. Three types of objectives are considered simultaneously, including minimization of the total energy consumption of the trucks, total energy consumption of the drones, and the total number of trucks. An improved artificial bee colony algorithm is designed to solve the problem. In the proposed algorithm, each solution is represented by a two-dimensional vector, and the initialization method based on the Push-Forward Insertion Heuristic is embedded. To enhance the exploitation abilities, an improved employed heuristic is developed to perform detailed local search. Meanwhile, a novel scout bee strategy is presented to improve the global search abilities of the proposed algorithm. Several instances extended from the Solomon instances are used to test the performance of the proposed improved artificial bee colony algorithm. Experimental comparisons with the other efficient algorithms in the literature verify the competitive performance of the proposed algorithm.

Keywords

Vehicle routing problem, artificial bee colony, time window, drone transportation, multi-objective

Date received: 9 October 2019; accepted: 23 February 2020

Topic: Robot Manipulation and Control

Topic Editor: Henry Leung

Associate Editor: Bin He

Introduction

The utilization of smart manufacturing and intelligent devices has greatly promoted the transforming of traditional industries.¹ Thus far, many emerging computer technologies^{2,3} and sophisticated equipment such as the robotics⁴ and drones⁵ have extremely improved the production efficiency and reduced the cost of the enterprise. Recently, numerous excellent enterprises have invested various drones to deliver their commodities^{6,7} while many researchers have also concentrated on the vehicle routing problem with drones (VRPD),^{8–12} which can be observed as an extended version of the vehicle routing problem (VRP).¹³ Many exhaustive overviews of the VRP could be found in the literature,^{14–16} where a number of customers need to be visited successively by a fleet of vehicles. In

VRPD, the customers should be served by both vehicles and drones.⁸ The applications of drones in the VRP systems have attracted many unexpected benefits.¹⁰ Wang et al. implemented a set of thoroughly mathematical experiments to testify that combining the drones with trucks or vehicles can

¹ School of Information and Engineering, Shandong Normal University, Jinan, China

² School of Computer Science, Liaocheng University, Liaocheng, China

³ School of Management Science and Engineering, Shandong University of Finance and Economics, Jinan, China

Corresponding author:

Jun-qing Li, School of Information and Engineering, Shandong Normal University, Jinan 250014, Shandong, China..

Email: lijunqing@lcn-cs.com



decrease nearly 75% costs.¹⁰ Wang and Sheu generalized that avoiding crowdedness, quick delivery, and saving manpower and material resources are the primary characteristics of the VRPD.¹⁷ Besides, Yurek and Ozmutlu suggested the drones' own unique advantages when delivering goods on the rough or difficult terrains.¹⁸

In the last decades, many algorithms have been applied to the vehicle routing problem with time window (VRPTW), such as the genetic algorithm (GA),¹⁹ the tabu search (TS) algorithm,²⁰ the variable neighborhood search (VNS) algorithm,²¹ the gradient evolution algorithm,²² the particle swarm optimization,²³ the ant colony optimization,²⁴ the adaptive large neighborhood search heuristic,²⁵ and the simulated annealing algorithm.²⁶ However, for some special requirements, the drones can complete transportation tasks. Chang and Lee established a model for one truck equipped a set of drones.²⁷ Wang et al. studied a group of trucks with several drones.¹⁰ Murray and Chu designed a dispatching manner in which one drone cooperates with one truck to serve a fleet of customers.⁸ Furthermore, another valuable research direction involving the VRPD is the vehicle routing problem with time window and drones (VRPTWD).^{28,29} Guerriero et al. proposed a VRPTWD model considering the soft time window and customer satisfactions.²⁸ Phan and Suzuki studied a multi-objective optimization problem of VRPTWD with the constraints of simultaneous receiving and dispatching.³⁰ In a nutshell, the VRP with drones should be taken more and more research focuses, especially in special scenarios, such as the disaster relief.^{15,31,32}

The artificial bee colony (ABC) algorithm was firstly proposed by Karaboga.^{33,34} Since then, the ABC algorithm has been applied in many types of optimization problem, such as the graph selection problem,³⁵ the flexible job shop scheduling problem,^{36,37} the path planning problem,³⁸ the distributed flow shop scheduling problem,^{39,40} the block flowshop,⁴¹ the prefabricated optimization problem,⁴² the VRPTW problem,⁴³ the fuzzy scheduling problem,⁴⁴ and the task scheduling problem.⁴⁵ The ABC algorithm has been shown as a competitive optimization method compared with other efficient optimization methods, such as the Jaya algorithm,⁴⁶ the invasive weed optimization algorithm,⁴⁷ the differential evolution algorithm,⁴⁸ the harmony search algorithm,⁴⁹ the shuffled frog-leaping algorithm,⁵⁰ the bioinspired metaheuristic algorithm,⁵¹ the krill herd algorithm,⁵² the TS algorithm,⁵³ the particle swarm algorithm,⁵⁴ and other multi-objective optimization algorithms.^{55,56}

Considering the special requirements of drones in the VRP problem, and the efficiency of the ABC algorithm, we design an improved ABC (IABC) to solve the multi-objective VRPTWD (MO-VRPTW-D) problem. The main contributions of this study are as follows: (1) to the best of our knowledge, this study is the first one to dispose the VRPTW with drone transportation by using the ABC algorithm; (2) to enhance the exploitation abilities, an improved employed heuristic is developed to perform detailed local search; and

(3) a novel scout bee strategy is presented to improve the global search abilities of the proposed algorithm.

The remainder of this report is organized as follows. The second section describes the MO-VRPTW-D model and exhibits an example; the third section clarifies the framework of the proposed algorithm; the fourth section introduces the simulation results and the fifth section draws the conclusion and future work.

Problem statement and MO-VRPTW-D model

Problem description

The proposed MO-VRPTW-D can be generalized from three aspects: customers, trucks and drones, objectives and constraints.

For the customers, the complex transportation scenario, such as the disaster relief and rescue or the transport of delicate goods, should be considered. Besides, each customer has multiple pairs of productions or demands. Except from the special of height and demand, the customers also have the synchronized visits and time windows constraints.

For the trucks and drones, each truck equips a drone and the multiple pairs of productions are delivered to the horizontal location of a customer by the truck and raised to the customer by the drone. Moreover, each truck has two types of capacities due to the multiple pairs of productions for customers and the capacity of each truck is different. The truck that has a greater capacity will consume more fuels and generate more energy consumption than the smaller one. Furthermore, each drone possesses different flight speed and different energy consumption, that is, the faster drone will create greater energy consumption.

Additionally, the objectives of MO-VRPTW-D are to minimize the total energy consumptions of trucks, the entire energy consumptions of drones, and the maximal number of the employed trucks. And the total energy consumptions are respected to the energy consumption index and the travel distances for each truck or drone. For the constraints, first, each truck has the limit of maximum capacity and maximal travel distance, and the maximum number of the trucks is fixed; secondly, each customer has a time windows and the penalty will emerge when the arrival time of a truck at a customer is overstepped the time window. Besides, all the truck must depart and return to the depot and each customer must be visited only once by exactly one truck.

MO-VRPTW-D model

The MO-VRPTW-D is modeled as a directed complete graph $G = (N, A)$: N is the vertex set including the customers set $C = \{1, 2, \dots, n\}$ and the depot node 0; here 0_s and 0_e are respecting the start and end depot, respectively, and we set the $0 = 0_s = 0_e$ for clear expression; while A is the arc set comprising the set of horizontal distances between

customers $A_h = \{(i, j) | i, j \in C, i \neq j\}$ and a set of perpendicular distances $A_p = \{(i_b, i_t) | i \in C\}$; here i_b and i_t are symbolizing the bottom and top location of the perpendicular arc for customer i , respectively.

$V = \{1, 2, \dots, v\}$ is the trucks set and $D = \{d_1, d_2, \dots, d_v\}$ is a set of drones, where d_k respects the drone of the truck k . Each truck k has two types of capacities bm_k and bs_k while the former is the capacity of the main materials and the latter is the supplementations. Each customer $i \in C$ also has two kinds of demands dm_i and ds_i . There is a certain amount of proportional relationship between dm_i and ds_i such as one tent must be equipped four or more tent frames when building the temporary accommodations for the victims of a natural calamity. Notice that the amount of all these demands carried by truck k must be less than or equal to the corresponding maximal capacities of truck k . There is a maximal travel distance r_k of each truck k which starts at depot 0_s and ends at depot 0_e after serving some customers.

Moreover, each customer must be visited only once by exactly one truck and each customer i has a time window $[e_i, l_i]$. $[e_0, l_0]$ is the time window of depot, e_0 is equal to zero while l_0 is equal to r_k . The wait time w_i will be produced if the arrival time z_i of truck k at customer i is early than e_i and when z_i is later than l_i , the penalty will be attached to this route.

Specially, there is an energy consumption coefficient wt_k for each truck k while each drone d_k has an energy consumption coefficient wd_k and a flight speed yd_k . In this article, each customer i has a vertical trip distance tp_i and the serving duration s_i of customer i can calculate by dividing the double trip distance $2tp_i$ of by the yd_k . Here yd_k is the speed of drone equipped at truck k which visiting the customer i . Besides, α , β , and γ are the weight coefficients of the total energy consumption of the trucks, total energy consumption of the drones, and the total number of trucks, respectively.

Finally, all of the travel distance t_{ij} or tp_i is numerically equal to the travel or trip time in the MO-VRPTW-D model. The mathematical formulations are described below.

Decision variables

$$x_{ijk} = \begin{cases} 1, & \text{if truck } k \text{ travel from vertex } i \text{ to vertex } j \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ik} = \begin{cases} 1, & \text{if vertex } i \text{ is served by vehicle } k \\ 0, & \text{otherwise} \end{cases}$$

Objective

$$\begin{aligned} \min & \alpha \sum_{i=0}^n \sum_{j=0}^n \sum_{k=1}^v x_{ijk} wt_k t_{ij} + \beta \sum_{i=1}^n \sum_{k=1}^v x_{0jk} 2tp_i \frac{1}{yd_k} wd_k \\ & + \gamma \sum_{k=1}^v x_{0jk}, i \neq j, i, j \in N, k \in V \end{aligned} \quad (1)$$

Constraints

$$\sum_{i=1}^n dm_i y_{ik} \leq bm_k, \forall k \in V \quad (2)$$

$$\sum_{i=1}^n ds_i y_{ik} \leq bs_k, \forall k \in V \quad (3)$$

$$\sum_{k=1}^v y_{ik} = 1, \forall i \in C \quad (4)$$

$$\sum_{i=0}^n x_{ijk} = y_{jk}, \forall k \in V, \forall j \in C \quad (5)$$

$$\sum_{j=0}^n x_{ijk} = y_{jk}, \forall k \in V, \forall i \in C \quad (6)$$

$$\sum_{j=1}^n x_{0jk} - \sum_{i=1}^n x_{i0k} = 0, \forall k \in V \quad (7)$$

$$s_i = 2tp_i \frac{1}{yd_k}, \forall i \in C, \forall k \in V \quad (8)$$

$$\sum_{i=0}^n \sum_{j=0}^n x_{ijk} (t_{ij} + s_i + w_i) \leq r_k, \forall k \in V \quad (9)$$

$$w_0 = s_0 = 0 \quad (10)$$

$$\sum_{k=1}^v \sum_{i=0}^n x_{ijk} (z_i + w_i + s_i + t_{ij}) = z_j, \forall j \in C \quad (11)$$

$$e_i \leq z_i + w_i \leq l_i, \forall i \in C \quad (12)$$

$$w_i = \max\{0, e_i - z_i\}, \forall i \in C \quad (13)$$

$$x_{ijk} \in \{0, 1\}, \forall i, j \in C, \forall k \in V \quad (14)$$

$$y_{ik} \in \{0, 1\}, \forall i \in C, \forall k \in V \quad (15)$$

$$z_i \geq 0, \forall i \in C \quad (16)$$

The objective (1) aims to minimize the weighted sum of the total energy consumption of the trucks, total energy consumption of the drones, and the total number of trucks. Constraints (2) and (3) ensure the two kinds of truck capacities are not exceeded; constraints (4) to (6) guarantee that each customer is visited by only one truck and served only once; constraint (7) safeguards that each truck starts and ends at the depot; constraint (8) sets the serving duration; constraint (9) guarantees the maximal trip distance is not exceeded; constraint (10) indicates the waiting and starting time of depot; constraint (11) gives the connection between the arrival time of a vertex and the departure time of its predecessor; constraints (12) and (13) illuminate that the arrival time should be within the time window and the wait time.

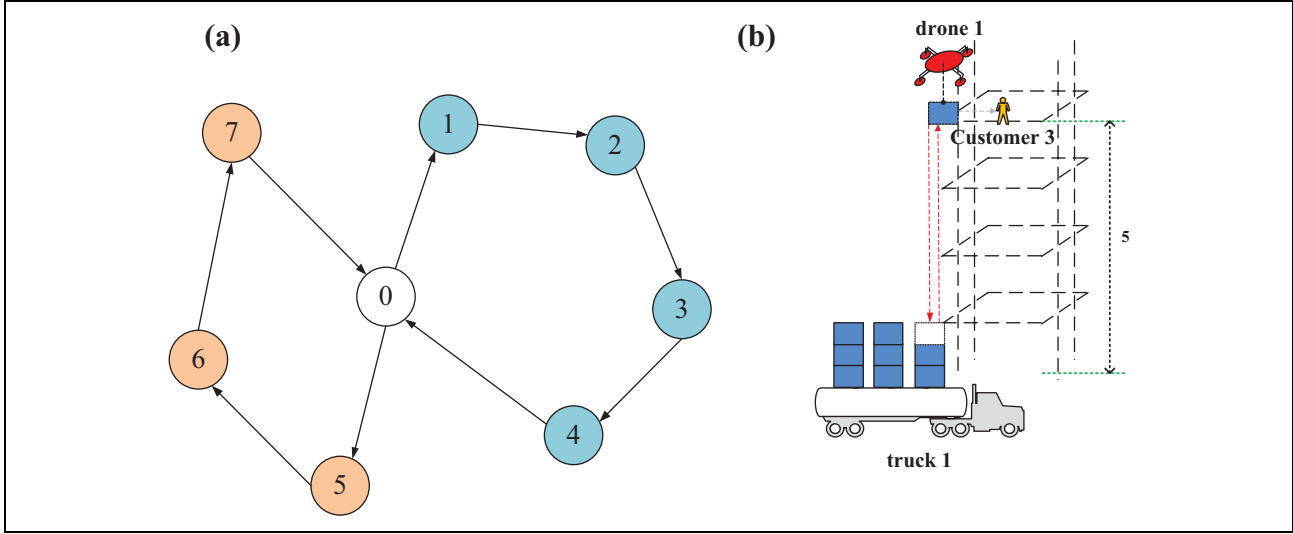


Figure 1. The example of MO-VRPTW-D. MO-VRPTW-D: multi-objective vehicle routing problem with time window and drones.

Example of MO-VRPTW-D

Figure 1 displays an example of the proposed problem, which includes seven customers, two trucks and drones, which is conducive to comprehending the dispatching process. Figure 1(a) is the whole delivery routes for all customers, while Figure 1(b) is the situation of dispatching goods to the customer 3. For convenience, we only discuss the delivery process of customer 3 and other conditions are as follows: (1) the route $\{0, 1, 2, 3, 4, 0\}$ and $\{0, 5, 6, 7, 0\}$ are executed by the truck 1 and truck 2, respectively; (2) drone 1 is equipped on the truck 1 and the speed of drone 1 is 0.5. It can be acquired from Figure 1 that, when the truck 1 arrival at customer 3, the drone 1 starts to dispatch goods. According to the height of customer 3 is 5 and the dispatching route is a double journey, thus, the total distance of drone 1 is 10. Moreover, on the basis of the speed of drone 1 is 0.5, thus, the trip time of the drone is 10 divided by 0.5 is 20. Furthermore, the serving duration of customer 3 is 20. And Figure 2 explains the more detailed distribution scheme, where each route is built by a truck with a drone and each customer has two types of demands. The bottom of Figure 2 shows a part of the route, in which the main materials and the supplementations of customer 2 are 1 and 3; of customer 4 are 1 and 2; of customer 4 are 1 and 4, respectively.

Proposed algorithm

The framework of the proposed IABC algorithm is shown in Algorithm 1.

Encoding and decoding methods

For the problem encoding, this study states each solution as a 2-D array where each array representing the sequence of a set of vertexes, and Figure 3 gives an example of the encoding. In Figure 3, $\{0, 1, 2, 3, 4, 0\}$ represent a visiting sequence of a truck, where this truck starts at depot then

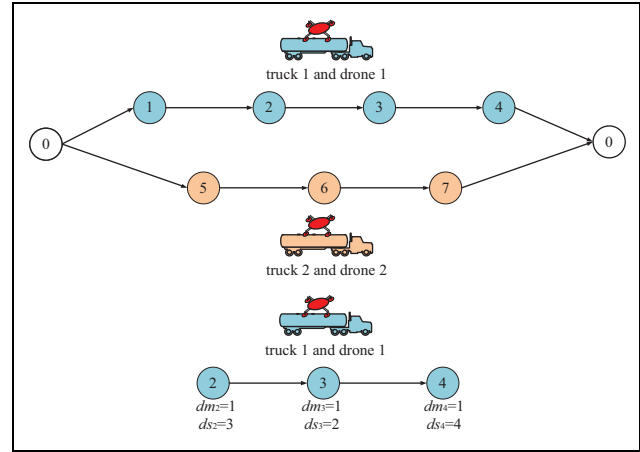


Figure 2. The illustration of two types of demands and the combination of truck and drone.

visits customer 1, 2, and so on until it returns to the depot. And $\{0, 5, 6, 7, 0\}$ is symbolizing another route. Notice that each customer in the constructed route must be selected under the constraints of the MO-VRPTW-D model.

Repair strategy

For each infeasible solution, Algorithm 2 gives the proposed repair strategy and the time complexity is $O(n^2m)$.

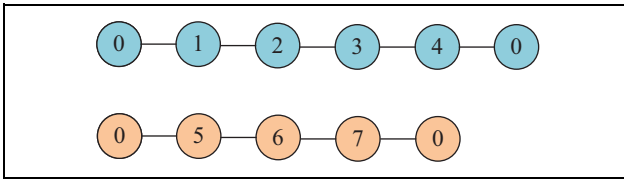
Initialization strategy

Solomon proposed the Push-Forward Insertion Heuristic (PFIH) for the VRPTW in 1987 and the general process of PFIH is as follow: first, select one of the unscheduled customers who must satisfy all the constraints of VRPTW model if insert it into the current route; then, calculate all the costs of inserting it into each location of the current route; thirdly, insert this customer into current route at the

Algorithm 1. The framework of the IABC.

Input: the datas of customers, trucks and drones
Output: the best solution X^*

- 1 generate P_s initial solutions X by the strategy in sub-section 3.4
- 2 **employed bee phase**
for each solution x do
 - 3 generate a new solution x^* by the strategy in sub-section 3.5 for solution x
 - 4 **if** x^* is better than x
 - 5 replace x with x^*
 - 6 update the best solution that has been found so far to x^*
 - 7 update the local best to x
 - 8 **end**
- 9 **end**
- 10 **onlooker bee phase**
for each solution x do
 - 11 randomly select a solution x_r
 - 12 implement the Employed bee phase on the best solution between x and x_r
 - 13 update the best solution X^*
- 14 **end**
- 15 **scout bee phase**
implement the Scout bee strategy for the best solution X^*
- 16 update the iterative time T of the IABC algorithm
- 17 update iteration times without improvement I_x for each solution x
- 18 **if** I_{x^*} exceeds the maximum no- improvement time L_n
- 19 implement the Scout bee strategy for the solution x'
- 20 **if** T not exceeds the T_{\max}
- 21 go to step 2

**Figure 3.** Example of the problem encoding.

position of minimum costs; fourthly, repeat the above three steps until all customers are arranged.²⁹ In this study, we also embed the PFIH method in the proposed algorithm.

Enhanced employed bee strategy

Algorithm 3 gives an enhanced employed bee strategy of this study. In the traditional employed bee strategy, we

select some customers randomly of certain route, then rearrange these customers to other routes. When some customers cannot be inserting into the current route, we could deploy a new truck if there was an empty one. This method is easy to accomplish but hard to receive a better solution. Thus, we propose a novel and effective strategy to obtain better routing schemes, where we expect rearranging a fleet of relevant customers with similar location.⁵⁷ And the time complexity of our strategy is $O(n^2m)$.

Scout bee strategy

When the IABC algorithm comes to the scout phase, we expect to make large changes to a solution in order to enhance the fitness and reduce the energy consumptions of these routes. Thus, the scout bee strategy described

Algorithm 2. Repair strategy.

```

Input: an infeasible solution
Output: an feasible solution
1  for each customer  $i$  on each route of an infeasible solution  $X$  do
2      if  $i$  violates its time window
3          keep  $i$  in the set  $S$  and delete it from its route
4  end
5  for each customer  $j$  in the set  $S$  do
6      try to insert  $j$  to each route of the current  $X$  according the PFIH method in sub-section 3.4
7      if customer  $j$  still cannot be arranged and there has at least one available empty truck
8           $k$ 
9          put the customer  $j$  into the truck  $k$ 
10         end
11        else
12            reject this solution  $X$ 
13        end
14    end

```

Algorithm 3. Enhanced employed bee strategy.

```

Input: an solution
Output: an improved solution
1  set the number  $n$  of selected customers set  $D$  is 0, and put all the customers into set  $D_n$ 
2  While  $n < Psize/10$ 
3      random select a customer  $r$  of  $D_n$  and sent  $r$  to the set  $D$ 
4      delete the customers in set  $D$  from set  $D_n$ 
5      random select a customer  $i$  of set  $D$ 
6      for each customer  $j$  in  $D_n$  do
7          calculate all the distance arc  $a(i, j)$ 
8      end
9      sort the set  $D_n$  from smallest to biggest accord all the arc  $a(i, j)$ 
10     generate a number  $d$  from  $[1, Psize/10]$ 
11     store the top  $d$  customers of  $D_n$  into  $D$  and delete these customers from  $D_n$ 
12     update the customers number  $n$  of set  $D$ 
13 end
14 delete set  $D$  from current solution
15 for each customer  $l$  of set  $D$  do
16     insert  $l$  to current solution according to the PFIH and delete  $l$  from set  $D$ 
17 end
18 if the set  $D$  is not empty
19     dispatch an available truck and reinsert these unscheduled customers
20 end

```

Algorithm 4. Scout bee strategy.

Input: the best solution X^*
Output: an improved solution x^*

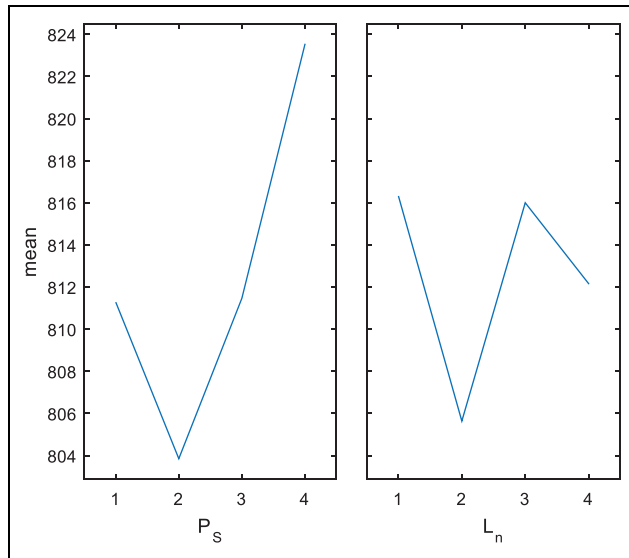
```

1  random select  $Psize/10$  customers from the best solution  $X^*$ 
2  put the  $Psize/10$  customers into the set  $D_{sc}$  and delete them from the best solution  $X^*$ 
3  for each customer  $i$  of  $D_{sc}$  do
4      | insert  $i$  into the current best solution  $X^*$  by the PFIH strategy
5  end and clean the set  $D_{sc}$ 
6  random select a truck  $k$  from the current best solution  $X^*$ 
7  store the customers of truck  $k$  in the set  $D_{sc}$  and delete these customers from truck  $k$ 
8  transform the truck  $k$  of  $X^*$  into another available empty truck
9  insert the customers of  $D$  into current  $X^*$  by the PFIH strategy
10 update the best solution  $X^*$ 
11 end

```

Table 1. The levels of the two parameters.

Parameter	Level 1	Level 2	Level 3	Level 4
P_s	50	100	150	200
L_n	5	10	15	20

**Figure 4.** The result of the DOE experiments for P_s and L_n . DOE: design of experiments.

in Algorithm 4 has two parts. First, we execute a kind of local search method to reinsert some customers of a solution; then, replacing a truck of this solution with another empty one for increasing the probability to reduce the energy consumption of current routes. The steps 6–11 of

Table 2. The comparison results of IABC-NE and IABC.

Instance	Deviation	
	IABC-NE	IABC
Inst1	0.00	3.65
Inst2	6.25	0.00
Inst3	0.00	4.57
Inst4	0.00	2.75
Inst5	0.00	1.95
Inst6	1.23	0.00
Inst7	2.44	0.00
Inst8	0.00	3.32
Inst9	8.12	0.00
Inst10	0.00	1.46
Inst11	7.31	0.00
Inst12	0.88	0.00
Inst13	11.35	0.00
Inst14	0.00	0.10
Inst15	0.00	2.34
Inst16	5.08	0.00
Inst17	8.51	0.00
Inst18	3.35	0.00
Inst19	2.14	0.00
Inst20	1.05	0.00
Inst21	5.45	0.00
Inst22	0.00	0.92
Inst23	3.45	0.00
Inst24	6.36	0.00
Inst25	0.00	1.28
Inst26	1.52	0.00
Inst27	7.48	0.00
Inst28	0.00	0.15
Inst29	0.18	0.00
Inst30	7.08	0.00
Inst31	0.00	8.35

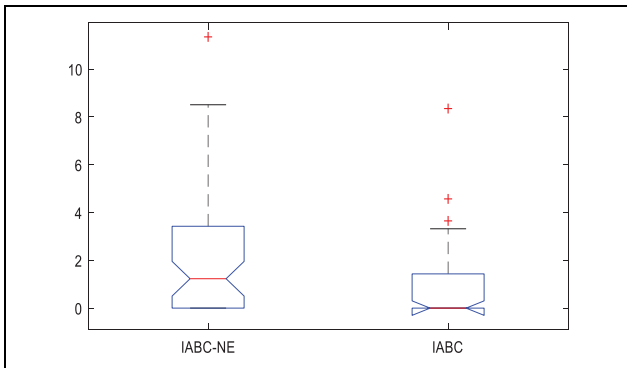
(continued)

Table 2. (continued)

Instance	Deviation	
	IABC-NE	IABC
Inst32	1.04	0.00
Inst33	3.48	0.00
Inst34	2.55	0.00
Inst35	1.33	0.00
Inst36	1.14	0.00
Inst37	0.00	3.03
Inst38	1.21	0.00
Inst39	0.00	1.35
Inst40	0.00	3.23
Inst41	0.44	0.00
Inst42	0.00	2.47
Inst43	0.00	2.88
Inst44	3.29	0.00
Inst45	0.00	1.75
Inst46	0.00	0.63
Inst47	1.99	0.00
Inst48	4.06	0.00
Inst49	3.18	0.00
Inst50	0.00	1.66
Inst51	2.20	0.00
Inst52	5.25	0.00
Inst53	1.36	0.00
Inst54	2.52	0.00
Inst55	2.31	0.00
Mean	2.30	0.87

IABC: improved artificial bee colony.

IABC-NE: improved artificial bee colony with a random employed bee strategy without the enhanced employed bee strategy.

**Figure 5.** The ANOVA comparisons of IABC-NE and IABC.

ANOVA: analysis of variance; IABC: improved artificial bee colony.

IABC-NE: improved artificial bee colony with a random employed bee strategy without the enhanced employed bee strategy.

Algorithm 4 is called the truck replacement strategy and the time complexity of the scout bee phase is $O(n^2m)$.

Experimental comparisons

In this study, in order to testify the validity of IABC, three relevant and effective algorithms, that is, GA,¹⁹ TS,²⁰ and

Table 3. The results of the IABC-NS and IABC.

Instance	Deviation	
	IABC-NS	IABC
Inst1	0.29	0.00
Inst2	10.22	0.00
Inst3	0.53	0.00
Inst4	0.00	2.32
Inst5	2.20	0.00
Inst6	0.25	0.00
Inst7	0.00	2.42
Inst8	2.16	0.00
Inst9	1.54	0.00
Inst10	0.00	4.34
Inst11	0.00	5.79
Inst12	6.98	0.00
Inst13	17.64	0.00
Inst14	0.00	5.00
Inst15	0.00	1.02
Inst16	0.00	2.99
Inst17	14.80	0.00
Inst18	0.68	0.00
Inst19	0.56	0.00
Inst20	0.10	0.00
Inst21	5.46	0.00
Inst22	6.08	0.00
Inst23	3.87	0.00
Inst24	0.00	2.25
Inst25	0.06	0.00
Inst26	1.21	0.00
Inst27	2.84	0.00
Inst28	0.00	0.78
Inst29	1.07	0.00
Inst30	0.68	0.00
Inst31	0.00	0.85
Inst32	1.69	0.00
Inst33	4.78	0.00
Inst34	0.00	0.19
Inst35	0.90	0.00
Inst36	0.00	10.45
Inst37	0.00	4.10
Inst38	3.46	0.00
Inst39	0.00	3.79
Inst40	5.51	0.00
Inst41	0.00	1.32
Inst42	1.78	0.00
Inst43	3.16	0.00
Inst44	2.95	0.00
Inst45	2.64	0.00
Inst46	1.59	0.00
Inst47	2.33	0.00
Inst48	3.04	0.00
Inst49	0.77	0.00
Inst50	2.17	0.00
Inst51	0.00	2.20
Inst52	0.00	7.47
Inst53	6.72	0.00
Inst54	9.65	0.00
Inst55	0.00	2.62
Mean	2.41	1.09

IABC: improved artificial bee colony.

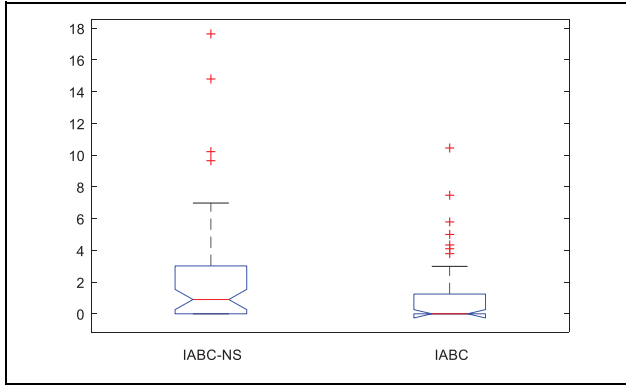


Figure 6. The ANOVA of the IABC-NE and IABC. ANOVA: analysis of variance; IABC: improved artificial bee colony. IABC-NE: improved artificial bee colony with a random employed bee strategy without the enhanced employed bee strategy.

VNS²⁵ are selected as the comparing algorithm. The “Experimental parameters and instances” subsection describes the parameters and instances setting of the MO-VRPTW-D and other experimental results are expressed in the rest of the fourth section.

Experimental parameters and instances

The two most significant parameters of IABC, that is, are the size of population P_s and the iteration times without improvement L_n . We set four levels for the each parameter in Table 1 and perform a mass of adjustable parameter experiments according to these levels. Accord to the result of design of experiments (DOE) in Figure 4, we set $P_s = 100$ and $L_n = 10$, respectively.

For the instances of this study, we design our instances based on the 55 famous Solomon Benchmark examples of

Table 4. The results of comparing with GA, TS, and VNS.

Instance	Best	Algorithm				Deviation			
		GA	TS	VNS	IABC	GA	TS	VNS	IABC
Inst1	1038.78	1571.34	1306.84	1351.27	1038.78	51.27	25.81	30.08	0.00
Inst2	1001.63	1275.01	1163.90	1125.35	1001.63	27.29	16.20	12.35	0.00
Inst3	950.11	1134.35	950.11	980.92	966.26	19.39	0.00	3.24	1.70
Inst4	956.58	1217.11	981.48	1011.20	956.58	27.24	2.60	5.71	0.00
Inst5	1010.41	1313.63	1108.59	1112.75	1010.41	30.01	9.72	10.13	0.00
Inst6	1018.00	1443.83	1175.09	1220.65	1018.00	41.83	15.43	19.91	0.00
Inst7	993.27	1246.36	1116.44	1151.58	993.27	25.48	12.40	15.94	0.00
Inst8	898.22	1244.89	908.67	968.04	898.22	38.60	1.16	7.77	0.00
Inst9	891.91	1152.61	975.02	946.11	891.91	29.23	9.32	6.08	0.00
Inst10	401.30	927.18	783.94	747.19	401.30	131.05	95.35	86.19	0.00
Inst11	519.29	802.33	733.91	801.66	519.29	54.51	41.33	54.38	0.00
Inst12	525.56	779.52	657.00	661.34	525.56	48.32	25.01	25.83	0.00
Inst13	518.87	751.77	591.87	600.71	518.87	44.89	14.07	15.77	0.00
Inst14	503.33	932.10	811.10	791.39	503.33	85.19	61.15	57.23	0.00
Inst15	464.63	796.87	680.49	707.22	464.63	71.51	46.46	52.21	0.00
Inst16	558.53	839.41	664.32	714.66	558.53	50.29	18.94	27.95	0.00
Inst17	499.76	680.18	499.76	513.54	509.22	36.10	0.00	2.76	1.89
Inst18	1578.94	1643.01	1585.90	1578.94	1599.14	4.06	0.44	0.00	1.28
Inst19	1355.03	1461.51	1355.03	1390.12	1355.08	7.86	0.00	2.59	0.00
Inst20	1063.10	1151.01	1064.14	1082.83	1063.10	8.27	0.10	1.86	0.00
Inst21	797.18	895.03	813.34	818.38	797.18	12.27	2.03	2.66	0.00
Inst22	1117.63	1185.05	1142.73	1117.63	1135.03	6.03	2.25	0.00	1.56
Inst23	981.92	1062.77	981.92	1002.99	999.78	8.23	0.00	2.15	1.82
Inst24	832.25	887.26	870.08	892.30	832.25	6.61	4.55	7.22	0.00
Inst25	687.64	769.06	687.64	730.08	694.18	11.84	0.00	6.17	0.95
Inst26	955.97	1038.45	965.79	971.37	955.97	8.63	1.03	1.61	0.00
Inst27	824.02	881.13	860.44	858.06	824.02	6.93	4.42	4.13	0.00
Inst28	807.97	909.46	844.24	830.10	807.97	12.56	4.49	2.74	0.00
Inst29	692.69	746.96	696.30	698.55	692.69	7.84	0.52	0.85	0.00
Inst30	844.25	899.89	887.98	911.36	844.25	6.59	5.18	7.95	0.00
Inst31	746.70	780.15	753.14	746.70	748.66	4.48	0.86	0.00	0.26
Inst32	634.32	699.21	647.16	639.32	634.32	10.23	2.02	0.79	0.00
Inst33	432.21	449.42	447.22	446.69	432.21	3.98	3.47	3.35	0.00
Inst34	561.85	575.88	561.85	567.96	565.48	2.50	0.00	1.09	0.65
Inst35	460.63	501.04	460.63	476.30	465.17	8.77	0.00	3.40	0.98

(continued)

Table 4. (continued)

Instance	Best	Algorithm				Deviation			
		GA	TS	VNS	IABC	GA	TS	VNS	IABC
Inst36	435.31	485.19	463.62	450.79	435.31	11.46	6.50	3.56	0.00
Inst37	351.92	358.75	351.92	355.68	352.98	1.94	0.00	1.07	0.30
Inst38	527.66	544.00	542.80	527.66	536.13	3.10	2.87	0.00	1.61
Inst39	538.09	615.62	538.09	553.50	541.62	14.41	0.00	2.86	0.65
Inst40	428.06	479.07	462.60	462.27	428.06	11.92	8.07	7.99	0.00
Inst41	1324.72	1401.42	1324.97	1349.98	1324.72	5.79	0.02	1.91	0.00
Inst42	1087.30	1231.29	1158.80	1169.08	1087.30	13.24	6.58	7.52	0.00
Inst43	936.53	1000.98	937.17	974.72	936.53	6.88	0.07	4.08	0.00
Inst44	878.78	933.55	887.90	878.78	903.21	6.23	1.04	0.00	2.78
Inst45	1269.41	1417.76	1290.85	1303.18	1269.41	11.69	1.69	2.66	0.00
Inst46	987.55	1040.40	998.61	987.55	1016.75	5.35	1.12	0.00	2.96
Inst47	939.38	1000.64	952.59	952.25	939.38	6.52	1.41	1.37	0.00
Inst48	876.64	937.55	891.03	919.59	876.64	6.95	1.64	4.90	0.00
Inst49	855.66	899.20	864.75	873.83	855.66	5.09	1.06	2.12	0.00
Inst50	731.28	811.90	756.29	780.86	731.28	11.02	3.42	6.78	0.00
Inst51	544.70	627.43	564.02	569.84	544.70	15.19	3.55	4.62	0.00
Inst52	479.69	511.90	485.82	479.69	483.92	6.72	1.28	0.00	0.88
Inst53	826.79	912.19	844.77	826.79	842.60	10.33	2.17	0.00	1.91
Inst54	626.18	658.18	627.59	626.18	631.60	5.11	0.23	0.00	0.87
Inst55	565.48	707.02	589.38	688.58	565.48	25.03	4.23	21.77	0.00
Mean	787.92	931.25	841.23	852.65	791.37	20.61	8.60	10.10	0.42

GA: genetic algorithm; TS: tabu search; VNS: variable neighborhood search; IABC: improved artificial bee colony.

the VRPTW.²⁹ Each Solomon example has 100 customers and the distributions of customers' location have three levels: random, cluster, and semi-cluster. Each customer in the Solomon examples has a time window, a demand, a severing duration, and a pair of coordinate points. The designed instances have the same customer numbers, locations, and time windows with the Solomon example and the name of our instances are corresponding to the order of Solomon instances. Due to the two kinds of demands constraints, we randomly set the dm_i , ds_i for each customer i from [1:1] to [1:5]. Besides, each customer has an integer height which is generated randomly from 5 to 14. Moreover, the total number of trucks is 25, and each truck has different energy consumption coefficient and two kinds of capacities. And the speed and energy consumption coefficient of drone on each truck is also different.

Effectiveness of the enhanced employed bee strategy

To testify the effectiveness of the proposed employed bee strategy, we perform a detailed experiment. Table 2 illustrates the comparison results of the enhanced employed bee strategy and the canonical employed bee method. In Table 2, the last two columns introduce the deviation of the objective value of each strategy compared with the best value. According to the results in Table 2, the IABC obtains 35 best values while the improved artificial bee colony with a random employed bee strategy without the enhanced employed bee strategy (IABC-NE) only gets 20 best ones, which demonstrate that the proposed employed

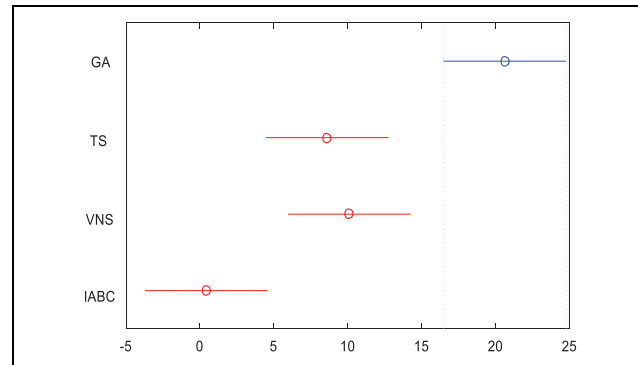


Figure 7. The result of comparing the IABC, GA, TS, and VNS. GA: genetic algorithm; TS: tabu search; VNS: variable neighborhood search; IABC: improved artificial bee colony.

bee strategy of this study indeed contributes to improving the algorithm performance. Figure 5 exhibits the result of analysis of variance (ANOVA) for the two different methods.

Effectiveness of the scout bee strategy

To investigate the performance of the proposed truck replacement strategy in the scout bee method, we also made a detailed comparison of the scout bee strategy. The results are revealed in Table 3, where IABC-NS represents the IABC without the scout bee strategy. It can be concluded from Table 3 that: (1) IABC obtains 37 best values while

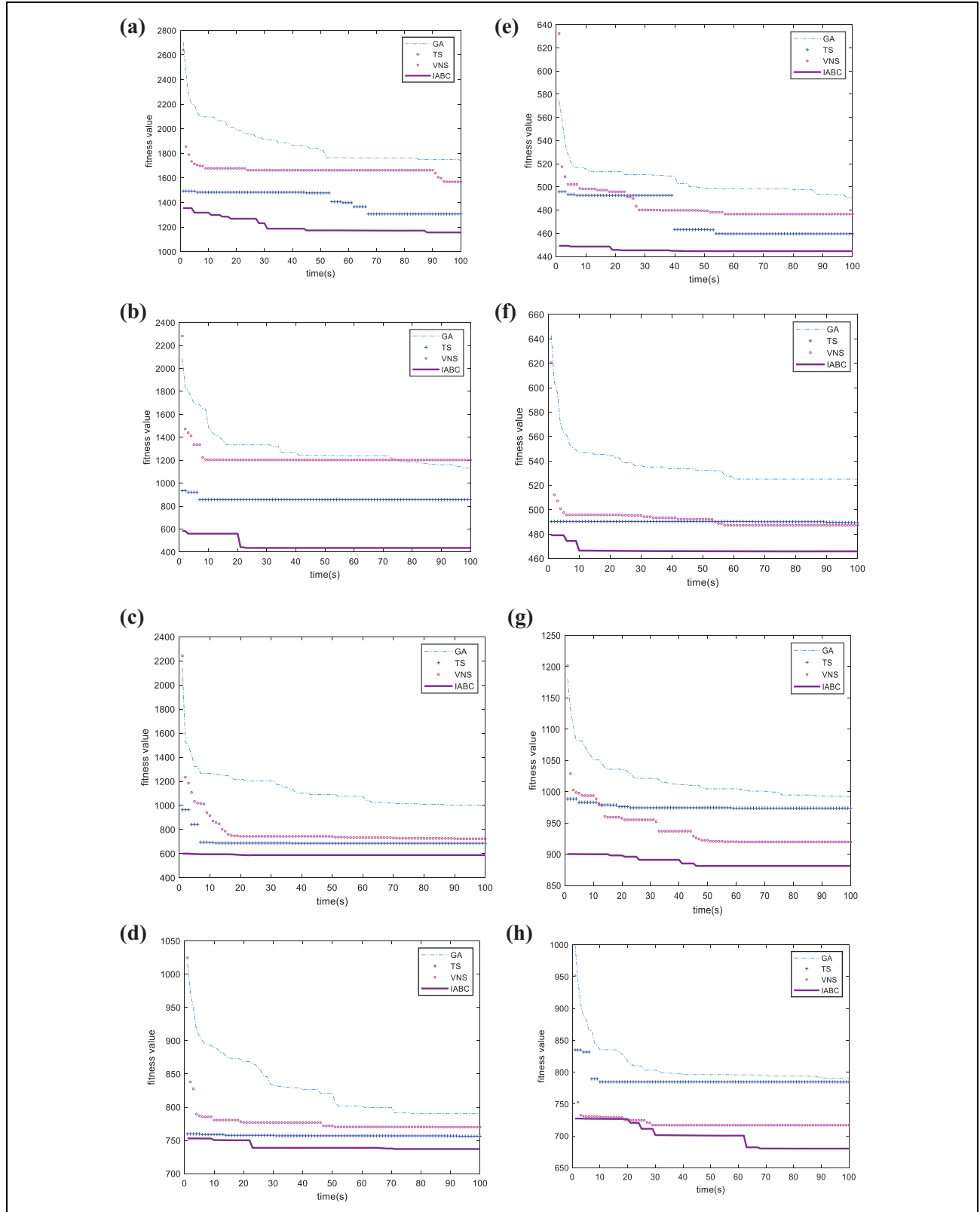


Figure 8. The convergence curves of instances. (a) Convergence curve for Inst 1. (b) Convergence curve for Inst 10. (c) Convergence curve for Inst 16. (d) Convergence curve for Inst 25. (e) Convergence curve for Inst 33. (f) Convergence curve for Inst 40. (g) Convergence curve for Inst 48. (h) Convergence curve for Inst 55.

IABC-NS only obtains 18 optimal solutions; (2) the deviation values from the last two columns also show the superiority of the proposed scout bee strategy; and (3) the average performance given in the last line verify the average performance of the proposed method. The ANOVA comparison of the two methods given in Figure 6 also verifies that the proposed scout bee strategy improved the performance significantly.

Comparison of several efficient algorithms

To further attest the powerful performance of the IABC, we compare the proposed algorithm with GA,¹⁹ TS,²⁰ and VNS.²⁵ These three selected algorithms are efficient for solving the VRPTW. The comparison results are given in Table 4, where the three columns from 3 to 6 correspond to the results obtained by GA, TS, VNS, and IABC, respectively. The best values of these four algorithms are shown in the second column. The deviations between each algorithm and the best values are present in the last four columns. Obviously, IABC gains 38 best values which account for about 70% of the 55 instances. The mean of deviation of IABC is only 0.42 which less than 5% of the second best result. Thus, in terms of the effectiveness and stability of the algorithm, the IABC is superior to the other three algorithms. Figure 7 gives the ANOVA among these four algorithms, which demonstrate that the IABC is significantly better than the three compared algorithms.

Conclusion

This article proposes a novel VRPTW for the special customers must be served by trucks and drones. Based on the problem features, the mathematic model of MO-VRPTW-D is designed and solved by the presented IABC Figure 8. However, there still have many challenges between the model and the practical applications. For trucks and drones, we usually deem the motion states of these as a linear process but the speeds are nonlinear in reality. The route also may be impacted by the emergency and weather. As a result, our future work will be devoted to following works: (1) refer to He et al.,^{58,59} and extend the model from a static scenario to a dynamic scenario; and (2) considering the fuzzy features of the realistic system,^{60,61} and explore the new mode of combining drone with vehicle under uncertain environments.

Declaration of conflicting interests


The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This study is partially supported by National Science Foundation of China under Grants 61773192, 61803192, and 61773246,

Shandong Province Higher Educational Science and Technology Program (J17KZ005), special fund plan for local science and technology development lead by central authority, major basic research projects in Shandong (ZR2018ZB0419), and a Grant of Key Laboratory of Intelligent Optimization and Control with Big Data.

ORCID iD

Jun-qing Li  <https://orcid.org/0000-0002-3617-6708>

References

1. Kusiak A. Smart manufacturing. *Int J Prod Res* 2018; 56(1–2): 508–517.
2. He B, Shao Y, Wang S, et al. Product environmental footprints assessment for product life cycle. *J Clean Prod* 2019; 233(10): 446–460.
3. He B, Liu Y, Zeng L, et al. Product carbon footprint across sustainable supply chain. *J Clean Prod* 2019; 241: 118320.
4. He B, Wang S, and Liu Y. Underactuated robotics: a review. *Int J Adv Robot Syst* 2019; 16(4): 1729881419862164.
5. Floreano D and Wood RJ. Science, technology and the future of small autonomous drones. *Nature* 2015; 521(7553): 460–466.
6. Yurek EE and Ozmutlu HC. A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. *Transp Res Part C Emerg Technol* 2018; 91: 249–262.
7. Otto A, Agatz N, Campbell J, et al. Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: a survey. *Networks* 2018; 72(4): 411–458.
8. Murray CC and Chu AG. The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery. *Transp Res Part C Emerg Technol* 2015; 54: 86–109.
9. Dorling K., Heinrichs J, Messier GG, et al. Vehicle routing problems for drone delivery. *IEEE Trans Syst Man Cybern Syst* 2016; 47(1): 70–85.
10. Wang X, Poikonen S, and Golden B. The vehicle routing problem with drones: several worst-case results. *Optim Lett* 2017; 11(4): 679–697.
11. Agatz N, Bouman P, and Schmidt M. Optimization approaches for the traveling salesman problem with drone. *Transp Sci* 2018; 52(4): 965–981.
12. Karak A and Abdelghany K. The hybrid vehicle-drone routing problem for pick-up and delivery services. *Transp Res Part C Emerg Technol* 2019; 102: 427–449.
13. Dantzig GB and Ramser JH. The truck dispatching problem. *Manag Sci* 1959; 6(1): 80–91.
14. Eksioglu B, Vural AV, and Reisman A. The vehicle routing problem: a taxonomic review. *Comput Ind Eng* 2009; 57(4): 1472–1483.
15. Pillac V, Gendreau M, Gu  ret C, et al. A review of dynamic vehicle routing problems. *Eur J Oper Res* 2013; 225(1): 1–11.
16. Braekers K, Ramaekers K, and Van Nieuwenhuysse I. The vehicle routing problem: state of the art classification and review. *Comput Ind Eng* 2016; 99: 300–313.
17. Wang Z and Sheu JB. Vehicle routing problem with drones. *Transp Res Part B Methodol* 2019; 122: 350–364.

18. Yurek EE and Ozmutlu HC. A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. *Transp Res Part C Emerg Technol* 2018; 91: 249–262.
19. Ghoseiri K and Ghannadpour SF. Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Appl Soft Comput* 2010; 10(4): 1096–1107.
20. Cordeau JF and Maischberger M. A parallel iterated tabu search heuristic for vehicle routing problems. *Comput Oper Res* 2012; 39(9): 2033–2050.
21. Belhaiza S, Hansen P, and Laporte G. A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. *Comput Oper Res* 2014; 52: 269–281.
22. Zulvia FE, Kuo RJ, and Nugroho DY. A many-objective gradient evolution algorithm for solving a green vehicle routing problem with time windows and time dependency for perishable products. *J Clean Prod* 2019; 242: 118428.
23. Goksal FP, Karaoglan I, and Altiparmak F. A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery. *Comput Oper Res* 2013; 65(1): 39–53.
24. Yu B and Yang ZZ. An ant colony optimization model: the period vehicle routing problem with time windows. *Transp Res Part E Logist Transp Rev* 2011; 47(2): 166–181.
25. Ribeiro GM and Laporte G. An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Comput Oper Res* 2012; 39(3): 728–735.
26. Mousavi SM and Tavakkoli-Moghaddam R. A hybrid simulated annealing algorithm for location and routing scheduling problems with cross-docking in the supply chain. *J Manuf Syst* 2013; 32(2): 335–347.
27. Chang YS and Lee HJ. Optimal delivery routing with wider drone-delivery areas along a shorter truck-route. *Expert Syst Appl* 2018; 104: 307–317.
28. Guerriero F, Surace R, Loscri V, et al. A multi-objective approach for unmanned aerial vehicle routing problem with soft time windows constraints. *Appl Math Model* 2014; 38(3): 839–852.
29. Solomon MM. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper Res* 1987; 35(2): 254–265.
30. Phan DH and Suzuki J. Evolutionary multiobjective optimization for the pickup and delivery problem with time windows and demands. *Mobile Netw Appl* 2016; 21(1): 175–190.
31. Chowdhury S, Emelogu A, Marufuzzaman M, et al. Drones for disaster response and relief operations: a continuous approximation model. *Int J Prod Econ* 2017; 188: 167–184.
32. Rabta B, Wankmüller C, and Reiner G. A drone fleet model for last-mile distribution in disaster relief operations. *Int J Disaster Risk Reduct* 2018; 28: 107–112.
33. Karaboga D and Basturk B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *J Glob Optim* 2007; 39(3): 459–471.
34. Karaboga D and Akay B. A comparative study of artificial bee colony algorithm. *Appl Math Comput* 2009; 214(1): 108–132.
35. Tian G, Ren Y, Feng Y, et al. Modeling and planning for dual-objective selective disassembly using AND/OR graph and discrete artificial bee colony. *IEEE Trans Ind Inform* 2018; 15(4): 2456–2468.
36. Li JQ, Pan QK, and Tasgetiren MF. A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities. *Appl Math Model* 2014; 38(3): 1111–1132.
37. Wu XL and Sun YJ. Flexible job shop green scheduling with an energy-saving measure. *J Clean Prod* 2018; 172: 3249–3264.
38. Liu H, Xu B, Lu D, et al. A path planning approach for crowd evacuation in buildings based on improved artificial bee colony algorithm. *Appl Soft Comput* 2018; 68: 360–376.
39. Li JQ, Song MX, Wang L, et al. Hybrid artificial bee colony algorithm for a parallel batching distributed flow shop problem with deteriorating jobs. *IEEE Trans Cybern* 2019: 1–15.
40. Li JQ, Bai SC, Duan PY, et al. An improved artificial bee colony algorithm for addressing distributed flow shop with distance coefficient in a prefabricated system. *Int J Prod Res* 2019; 57(22): 6922–6942.
41. Han YY, Gong DW, and Sun XY. A discrete artificial bee colony algorithm incorporating differential evolution for flow shop scheduling problem with blocking. *Eng Optim* 2015; 47(7): 927–946.
42. Li JQ, Han YQ, Duan PY, et al. Meta-heuristic algorithm for solving vehicle routing problems with time windows and synchronized visit constraints in prefabricated systems. *J Clean Prod* 2019; 250: 119464.
43. Yu S, Tai C, Liu Y, et al. An improved artificial bee colony algorithm for vehicle routing problem with time windows: a real case in Dalian. *Adv Mech Eng* 2016; 8(8): 1687814016665298.
44. Gao KZ, Suganthan PN, Pan QK, et al. Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion. *Knowl-Based Syst* 2016; 109: 1–16.
45. Li JQ and Han Y. A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system. *Clust Comput* 2019; 1–17.
46. Gao K, Yang F, Zhou M, et al. Flexible job-shop rescheduling for new job insertion by using discrete Jaya algorithm. *IEEE Trans Cybern* 2018; 49(5): 1944–1955.
47. Zheng ZX, Li JQ, and Han YY. An improved invasive weed optimization algorithm for solving dynamic economic dispatch problems with valve-point effects. *J Exp Theor Artif Intell* 2019; 1–25.
48. Wu GH, Shen X, Li HF, et al. Ensemble of differential evolution variants. *Inf Sci* 2018; 423: 172–186.
49. Wang GG, Gandomi AH, Zhao XJ, et al. Hybridizing harmony search algorithm with cuckoo search for global numerical optimization. *Soft Comput* 2016; 20(1): 273–285.
50. Li JQ, Pan QK, and Sheng XX. An effective shuffled frog-leaping algorithm for multi-objective flexible job shop

- scheduling problems. *Appl Math Comput* 2012; 218(18): 9353–9371.
51. Wang GG. Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Comput* 2018; 10(2): 151–164.
 52. Wang GG, Deb S, Gandomi AH, et al. Opposition-based krill herd algorithm with Cauchy mutation and position clamping. *Neurocomputing* 2016; 177: 147–157.
 53. Li JQ, Pan QK, and Liang YC. An effective hybrid tabu search algorithm for multi-objective flexible job shop scheduling problems. *Comput Ind Eng* 2010; 59(4): 647–662.
 54. Zhao XC. A perturbed particle swarm algorithm for numerical optimization. *Appl Soft Comput* 2010; 10(1): 119–124.
 55. Li JQ, Tao XR, Jia BX, et al. Efficient multi-objective algorithm for the lot-streaming hybrid flowshop with variable sub-lots. *Swarm Evol Comput* 2020; 52: 100600, doi:10.1016/j.swevo.2019.100600.
 56. Sun J, Miao Z, Gong D, et al. Interval multiobjective optimization with memetic algorithms. *IEEE Trans Cybern* 2020; 1–14. <https://doi.org/10.1109/TCYB.2019.2908485>.
 57. Ropke S and Pisinger D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transp Sci* 2006; 40(4): 455–472.
 58. He B, Huang S, and Wang J. Product low-carbon design using dynamic programming algorithm. *Int J Precis Eng Manuf Green Technol* 2015; 2(1): 37–42.
 59. He B, Zhang P, and Wang J. Automated synthesis of mechanisms with consideration of mechanical efficiency. *J Eng Design* 2014; 25(4-6): 213–237.
 60. Li CD, Yi JQ, Wang HK, et al. Interval data driven construction of shadowed sets with application to linguistic word modelling. *Inf Sci* 2020; 507: 503–521.
 61. Li JQ and Pan QK. Solving fuzzy job-shop scheduling problems by a hybrid optimization algorithm. *Journal of Mechanical Engineering* 2013; 49(23): 142–149.

World Conference on Technology, Innovation and Entrepreneurship

An Efficient Genetic Algorithm for Large Scale Vehicle Routing Problem Subject to Precedence Constraints

Noraini Mohd Razali^{a*}

Faculty of Manufacturing Engineering, Universiti Malaysia Pahang, 26600, Pekan, Pahang, Malaysia

Abstract

The vehicle routing problem is a combinatorial optimization and integer programming problem seeking to service a number of customers with a fleet of vehicles. Vehicle routing problem has many applications in the fields of transportation, distribution and logistics. In this study, the vehicle routing problem subject to precedence constraints has been modeled as the Travelling Salesman Problem (TSP) with precedence constraints. Vehicle routing problem is difficult to solve using exact methods especially for large size instances due to computational expensive. Therefore, metaheuristic method based genetic algorithm has been developed to obtain feasible and optimal solution with less computation time. The conventional genetic algorithm procedure for TSP, with an order-based representation might generate invalid candidate solutions when precedence constraints are involved. In order to obtain feasible solution which does not violate precedence constraints, a route repair based topological sort technique is used and integrated in the genetic algorithm procedure. In this paper, a new algorithm is developed and the performances as well as the quality of the solution were evaluated against large scale test problems. The results from the computational experiments demonstrate that the algorithm with 'earliest position' task selection mechanism, linear order crossover and inversion mutation has better results in terms of number of generations and computation time to converge to optimal solution. The genetic operators used in this study are capable to introduce new fitter offspring and does not trapped in a local optimum. Therefore, it can be stated that the genetic algorithm approach developed in this study is efficient in solving large scale vehicle routing problem subject to precedence constraints with the objective of minimizing the distance travelled.

© 2015 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

Peer-review under responsibility of Istanbul Univeristy.

Keywords: Genetic algorithm; Topological sort; Vehicle routing problem; Precedence constraints

* Corresponding author. Tel.: + 00694245944
E-mail address: norainimbr@ump.edu.my

1. Introduction

The vehicle routing problem represents a very important part of any distribution systems. The vehicle routing problem has received a lot of attention in the past. Such problem are concerned with a set of problems of designing routes, beginning at one depot or some depots and ending at the depot/depots, for a fleet of vehicles serving a set of customers with known or stochastic demands, such as pickup or delivery of mails, foods, coals, weapons. Different management objectives and constraints can form variously specified problems. The common objectives include maximizing profit, minimizing cost or distance, or maximizing number of vehicles. The common constraints include vehicle capacity, time window, number of customers served by one vehicle, precedence constraints where one request must be served before another, compatibility constraints, and types of vehicle. In this paper vehicle routing problems with precedence constraints are considered, where the objectives is to find the best route with the minimum traveled distance to organize the delivery of some goods or service to customers on a map by a vehicle from one depot to a final customers. The vehicle routing problem subject to precedence constraints can be modeled as the Travelling Salesman Problem (TSP) with precedence constraints.

2. Literature Review

The vehicle routing problem subject to precedence constraints can be described as follows. There is a depot and a set of customers. The travelling distance between the depot and the customers, as well as between any pair of customers is known. Then the problem consists of finding a set of route that minimize the total travelling distance required to serve all customers, while satisfying the precedence constraints. The vehicle routing problem with a single vehicle and precedence constraints is commonly seen as a traveling salesman problem (TSP) with precedence constraints (Moon et al., 2002). Fagerholt and Christiansen (2000) use the single vehicle VRSP-TW with additional allocation constraints to solve a subproblem arising in a ship scheduling application. If we introduce capacity constraints to the VRSP-TW, depending on the precedence constraints, we get a pickup and delivery problem with time windows (PDP-TW) (Desrosiers et al., 1995). Sigurd et al. (2004) use precedence constraints for an application that arise in the live animal transport.

Precedence constraints arise whenever one activity or series of activities must occur before beginning another activity or set of activities. Precedence embodies the logical sequence of real life activities that occurs in a problem such as a taxi cab first picking up its customer and, subsequently, delivering the customer. Some practical applications include the dial-a-ride problem, airline scheduling, bus routing, tractor-trailer problem, helicopter support of offshore oil field platforms and logistics and maintenance support. They also arise in less obvious situations such as VLSI circuit design, flexible manufacturing systems and evacuating casualties. For example, during the evacuation of a special needs population in anticipation of a disaster, medical personal might need to visit some patients before they can be transported to shelters.

Real life vehicle routing problems are usually so large that exact methods cannot be used to solve them. As the vehicle routing problem is an NP-hard problem (Lenstra and Rinnooy Kan, 1981), exact algorithms are only efficient for small problem instances. Given the difficulty of vehicle routing problem, many heuristic algorithms do not seek global optimal solutions, but rather seek to provide fast near-optimal solutions and are customized to model specific situations. For the past two decades, the emphasis has been on metaheuristics, which are methods used to find good solutions quickly. Genetic algorithm belongs to the group of metaheuristics. Relatively few experiments have been performed using genetic algorithm to solve the vehicle routing problem, which makes this approach interesting. However, the conventional genetic algorithm procedure for TSP, with an order-based representation might generate invalid candidate solutions when precedence constraints are involved. In order to obtain feasible solution which does not violate precedence constraints, topological sort technique is used and integrated in the genetic algorithm procedure. In this paper, a new genetic algorithm is developed namely 'GAnew' and the performance as well as the quality of the solution is compared with the previous developed algorithm namely 'GAold'. The GAold algorithm which is specifically used to solve TSP with precedence constraints has been developed by Moon et. al (2002).

3. Problem formulation

We assume to have one vehicle available in a depot, and a set of customers to be visited. The vehicle routing problem with precedence constraints (VRP-PC) is one in which a set of v nodes and distances for each pair of nodes are given, the problem is to find a tour from node 1 to node n of minimal length which takes given precedence constraints into account. In VRP-PC some order of customers is given and we ought to visit customers in that order only. VRP-PC differs from traditional VRP whereby in VRP-PC, there is no need to return to the original depot. This is called open vehicle routing problem (Golden & Wasil, 2006). The important feature of this problem, which distinguishes it from the traditional VRP, is that the vehicle is not required to return to the depot, or if it is required to do so, it must be accomplished by revisiting the customers assigned to it in the reverse order. Therefore the vehicle route is not closed path but open ones. An example of directed graph also known as precedence diagram to illustrate VRP-PC is shown in Fig. 1. Each precedence constraint requires that some node i have to be visited before some other node j (Kotecha & Gambhava, 2003). In a directed graph, the vertices (circles) represent locations to serve and the edges represent the precedence relations between locations (Moon et al, 2002).

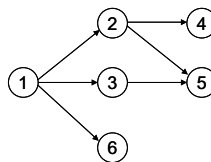


Fig. 1. Example of directed graph

Gambardella and Dorigo (1997) modelled the problem by considering a complete graph $P = (V, R)$ with node set V and arc set R , where nodes correspond to activities $0, \dots, i, \dots, n$. To simplify the explanation, reconsider the diagram in Fig. 1. The nodes set V consist of $(1, 2, 3, 4, 5, 6)$. Precedence constraints given by set R which contain $[(1, 2), (1, 3), (1, 6), (2, 4), (2, 5), (3, 5)]$. This set is ascertained from arc $(i, j) \in R$ if activity i has to precede activity j in any feasible solution. Each arc (i, j) is associated a cost t_{ij} . Given this definition, the VRP-PC can be stated as the problem of finding a tour sequence subject to the precedence constraints which minimise the total travelling distance. There is therefore equivalent to the problem of finding a feasible Hamiltonian path with minimal cost under precedence constraints given by set R (Kotecha & Gambhava, 2003).

Moon et al. (2002) have used the two-commodity network flow model to formulate TSP with precedence constraints. This model can also be used to solve VRP-PC. In this formulation, c_{ij} is the travel distance from vertex v_i to v_j and s is the first selected vertex in the graph. Commodity p is supplied by $(n-1)$ units at a selected starting node and used by one unit at each node that is not the starting node while commodity q is consumed by $(n-1)$ units at the starting node and supplied by one unit at the other nodes. Here n is the number of nodes or cities. Three variables are used for the two-commodity network model: x_{ij}^p is the quantity of commodity p from vertex v_i to v_j and x_{ij}^q is the quantity of commodity q from vertex v_i to v_j .

$$x_{ij} = \begin{cases} 1 & \text{if vertex } j \text{ is visited immediately after vertex } i, \\ 0 & \text{otherwise,} \end{cases}$$

The two-commodity network flow model for the TSP with precedence constraints can be described as follows:

$$\text{Minimise } \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j}}^n \frac{1}{(n-1)} c_{ij} (x_{ij}^p + x_{ij}^q) \quad (1)$$

Subject to

$$\sum_{j=1}^n x_{ij}^p - \sum_{j=1}^n x_{ji}^p = \begin{cases} n-1 & \text{for } i = s, \\ -1 & \text{otherwise,} \end{cases} \quad (2)$$

$$\sum_{j=1}^n x_{ij}^p - \sum_{j=1}^n x_{ji}^p = \begin{cases} -(n-1) & \text{for } i = s, \\ +1 & \text{otherwise,} \end{cases} \quad (3)$$

$$\sum_{j=1}^n (x_{ij}^p + x_{ij}^q) = n-1 \quad \forall i, \quad (4)$$

$$x_{ij}^p + x_{ij}^q = (n-1)x_{ij} \quad \forall i \text{ and } j, \quad (5)$$

$$\sum_{j=1}^n x_{uj}^p - \sum_{j=1}^n x_{vj}^p \geq 1 \quad \text{for } (v_u \rightarrow v_v)(v_v \neq s), \quad (6)$$

$$x_{ij}^p \geq 0 \quad \forall i \text{ and } j, \quad (7)$$

$$x_{ij}^q \geq 0 \quad \forall i \text{ and } j, \quad (8)$$

$$x_{ij} \in \{0,1\} \quad \forall i \text{ and } j. \quad (9)$$

The objective function (1) is to obtain the total travel distance for all vertices, and the sum of commodities p and q between vertices v_i to v_j on any feasible sequence (i.e. $x_{ij}=1$) is equal to $n-1$. Constraints (2) and (7) are used to ensure the feasibility of flow of commodity p while constraints (3) and (8) are for feasibility of commodity q . Constraint (4) ensures a feasible tour, i.e. feasible sequence. Constraint (5) explains that, if $x_{ij}=1$, the sum of commodities p and q between v_i and v_j be $n-1$. Constraint (6) is for the precedence relationship between vertices.

4. Methodology

Genetic algorithms are inspired by the theory of natural selection by Charles Darwin. A population of individuals or solutions is maintained by the means of crossover and mutation operators, where crossover simulates reproduction. The quality of each solution is indicated by a fitness value. This value is used to select a solution from the population to reproduce and when solutions are excluded from the population. The average quality of the population gradually improves as new and better solutions are generated and worse solutions are removed.

In this paper repair operation to encounter precedence constraints is needed to ensure all chromosomes in the population are valid tours which do not violated the precedence constraints added to the tours. Fig. 2 illustrates the genetic algorithm process flow with the sequence repairing operation denoted by 'Route repair'. The genetic algorithm developed in this paper is called GAnew. The algorithm used combination 'earliest position' based topological sort technique for node selection, Roulette Wheel for chromosome selection, linear order technique for crossover and inversion mechanism for mutation. In contrast, the benchmarked algorithm from Moon et al. (2002) which is called GAold has use different combination of genetic operators. GAold utilized 'priority' based topological sort, Roulette Wheel selection, Moon crossover and Exchange mutation. The genetic algorithm procedure for GAnew is described as follows;

- *Initialization & Representation*

For initial population, the random permutation method is used to generate chromosomes. The integer from 1 to N , which is the number of locations or cities, is generated in random sequence. The number of chromosome generated is depending on the size of population. These sequences normally did not satisfy the precedence constraints. Therefore, the infeasible chromosomes must be repaired using the topological sort technique.

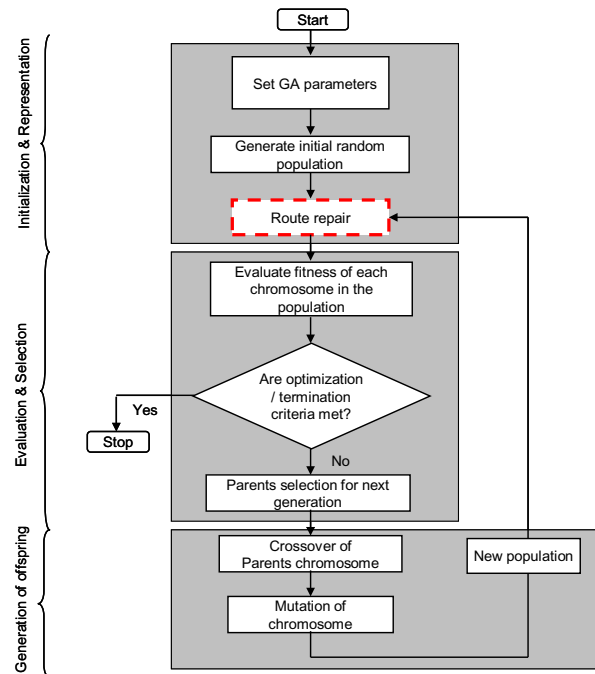


Fig. 2. Genetic algorithm procedure for vehicle routing problem with precedence constraints

- *Route Repair*

Since the chromosomes in the initial population are all randomly generated, the possibility of which generated chromosomes are infeasible due to the breaking of one or more precedence constraints is greater. The procedure of repairing chromosome in the initial population as well as after crossover and mutation operation is necessary before going through the evaluation process. The approach to overcome generating invalid sequence is based on a topological sort, which allows the genetic algorithm to generate only valid solutions in each generation. Every precedence graph has at least one topological sort. The topological sort is a node ordering in a directed graph such that if there is a path from a node v_i and a node v_j , then v_j appears after v_i in the ordering. In a directed graph, the nodes represent tasks or activities and the edges represent the precedence relations between tasks. More than a single sequence of tasks can be derived from a directed graph using the topological sort technique. Route repair based on the topological sort technique is explained in *Step 1* through *Step 3*.

- *Step 1: Check available node*

Initially, a chromosome is generated randomly and may not be feasible. For example the chromosome structure represented as [4 1 3 6 5 2] in Fig. 1 is infeasible because it did not satisfy the precedence constraints. In order to repair the chromosome become a feasible solution, nodes without predecessors are selected and stored in an available set. In this example, node 1 is the only node without a predecessor and therefore node 1 is selected and being stored in sequence. Then the outgoing edges of node 1, which are nodes 2, 3 and 6, are removed. As a result, the new available set consists of nodes 2, 3 and 6 as displayed in Table 1.

- *Step 2: Select node in earliest position on chromosome*

The selection of node to be stored in sequence is based on the ‘earliest position’ found in the chromosome. By referring to the available set [2 3 6], node 3 is firstly found in the chromosome [4 1 3 6 5 2]. Therefore, node 3 is selected as the second string to be stored in sequence and the updated sequence is now consists of [1 3].

- *Step 3: Remove edges from selected node*

When node 3 is selected to be stored in sequence, the outgoing edge of this node should be removed. Therefore, the edge $3 \rightarrow 5$ is removed, and the new available set is consisting of [2 6]. Again, based on ‘earliest position’ selection of node approach, node 6 is first appeared before node 2 in the chromosome [4 1 3 6 5 2] and therefore node 6 is selected to be placed in updated sequence. The selection procedure is repeated until the length of the sequence is equal to N . The final feasible sequence that is generated from this approach is [1 3 6 2 4 5]. An example of initial population with population size of 12 and repaired chromosomes for this population are shown in Fig. 3(a) and Fig. 3 (b), respectively. Most of the chromosomes in Fig. 3(a) are not feasible which is violating the precedence constraints. However the chromosomes in Fig. 3(b) are all feasible after repair operation using topological sort techniques.

Table 1. Selection of tasks using the “earliest position” technique

Chromosome: 4 1 3 6 5 2	
<i>available set</i>	<i>updated sequence</i>
1	[1]
2, 3, 6	[1 3]
2, 6	[1 3 6]
2	[1 3 6 2]
4, 5	[1 3 6 2 4]
5	[1 3 6 2 4 5]

4	1	3	6	5	2
4	1	3	5	6	2
6	3	2	4	1	5
4	2	3	6	1	5
1	4	6	5	3	2
5	4	1	3	2	6
3	5	2	4	1	6
2	5	1	6	3	4
1	6	4	5	2	3
2	6	5	1	3	4
5	1	6	3	4	2
5	1	6	3	4	2

(a)

1	3	6	2	4	5
1	3	6	2	4	5
1	6	3	2	4	5
1	2	4	3	6	5
1	6	3	2	4	5
1	3	2	5	4	6
1	3	2	5	4	6
1	2	6	3	5	4
1	6	2	4	3	5
1	2	6	3	5	4
1	6	3	2	5	4
1	6	3	2	5	4

(b)

Fig. 3 (a) Chromosomes in the initial population; (b) Chromosomes after repairing process

- *Evaluation and selection*

The fitness value of each chromosome in the population is evaluated using the fitness function in Equation (1). This equation is still valid for VRP-PC with excluding the distance of returning to the starting depot. The Roulette Wheel selection is used to select parent chromosomes to be re-generated for the next chromosome. By using the proportional Roulette Wheel, all individuals are given a chance to be selected and the chances of the fitter individual to be selected as a parent for crossover are higher.

- *Generation of offspring*

Linear order crossover is used to generate two new offspring. This operator is the most frequently used for the crossover operation when the chromosome representation is ordinal (Mokhlesian et al, 2010). This crossover operator can preserve both the relative positions between genes and the absolute positions relative to the extremities of parents as much as possible. Mutation operation based on inversion (flip) is applied in the chromosome after crossover process.

5. Results and analysis

Here, we test the performance of GAnew and GAold algorithm on 51 locations (customers) with 71 precedence constraints, and 100 locations with 141 precedence constraints. The first test problem consists of 51 locations and 71 precedence constraints in which the distance between the locations are randomly generated within [1, 15]. In order to assist the experimentation, full factorial design of experiment with 3 parameters each with 2 levels, low (-1) and high (+1) level is implemented. Therefore, the total number of computational runs will be 8 runs for one replication. Due to the large population size used in the experiment, each simulation experiment runs only one time in order to reduce computational time and resources.

The genetic algorithm parameters are set as follow; Population size (Popsiz) at 500 and 1000, probability of crossover (P_c) at 0.6 and 0.9, and probability of mutation (P_m) at 0.001 and 0.2. The maximum number of generation is set to 200 for all the experiments. The experiment is done using GAnew algorithm and the results obtained in each experiment are given in Table 2.

Table 2. Results of experiment for 51 locations and 71 precedence constraints using GAnew algorithm

Experiment #	Popsiz	P_c	P_m	Gen #	best
1	-1	-1	-1	96	209
2	+1	-1	-1	192	204
3	-1	+1	-1	112	224
4	+1	+1	-1	157	194
5	-1	-1	+1	122	209
6	+1	-1	+1	74	218
7	-1	+1	+1	109	219
8	+1	+1	+1	193	184

From this experiment, it can be ascertained that the algorithm with a combination of large population size and high crossover rate as well as the high mutation rate had produced better outcomes. The experiment #8 offers the best sequence of tour with minimum travelling distance. The sequence of this tour is [2-4-39-17-9-1-3-44-10-5-8-13-14-15-20-16-19-21-23-22-25-31-27-28-33-32-30-49-6-11-40-45-24-18-36-12-7-26-38-37-48-50-47-29-34-41-35-42-46-43-51]. This sequence confirmed that the chromosomes at the specified generation represent valid points in the search space, i.e. not violating the precedence constraints. The minimum total travelling distance is 184 and converged at generation 193. Thus, it can be expected that the quality of solution improves with the larger size of population and with a relatively high crossover and high mutation rate. The computation time to obtain the best solution is around 650 sec which is still in an acceptable amount of time to spend. Similar experiments were also carried out using the GAold algorithm in order to investigate and compare the quality of solution and the performance of the algorithm with the proposed GAnew algorithm. The same parameters setting are applied as in the GAnew algorithm.

Table 3. Results of experiment for 51 locations and 71 precedence constraints using GAold algorithm

Experiment #	Popsiz	P _c	P _m	Gen #	best
1	-1	-1	-1	150	268
2	+1	-1	-1	163	256
3	-1	+1	-1	83	277
4	+1	+1	-1	72	265
5	-1	-1	+1	101	272
6	+1	-1	+1	50	282
7	-1	+1	+1	51	279
8	+1	+1	+1	95	270

The results in Table 3 clearly show that the solution approaching minimum value with the utilization of high population size, low crossover rate and low mutation rate. The best solution found from the experiment is 256 and converged at generation 163. In order to check the feasibility of the solution from GAold algorithm, the sequences of tour generated is recorded which is [1-10-2-4-9-15-49-5-6-12-23-3-8-14-22-30-13-19-20-17-21-27-28-33-32-29-34-41-25-31-16-11-24-18-36-35-42-46-39-40-44-45-7-26-38-37-48-50-43-47-51].

Figure 4(a) shows the performance graphs for 200 generations of 1000 chromosomes. The travelling distance are plotted against the number of generations for the two experiments, i.e. experiment number 8 for the GANew algorithm and experiment number 2 for the GAold algorithm. It is observed that the 'Best' curves drop rapidly at the beginning of the run, but then as the population converges on the nearly optimal solution it drops more slowly, and finally flattens at the end. The iteration time for the proposed algorithm is larger than GAold algorithm. This is because a large number of generations are being utilized to converge on the best solution.

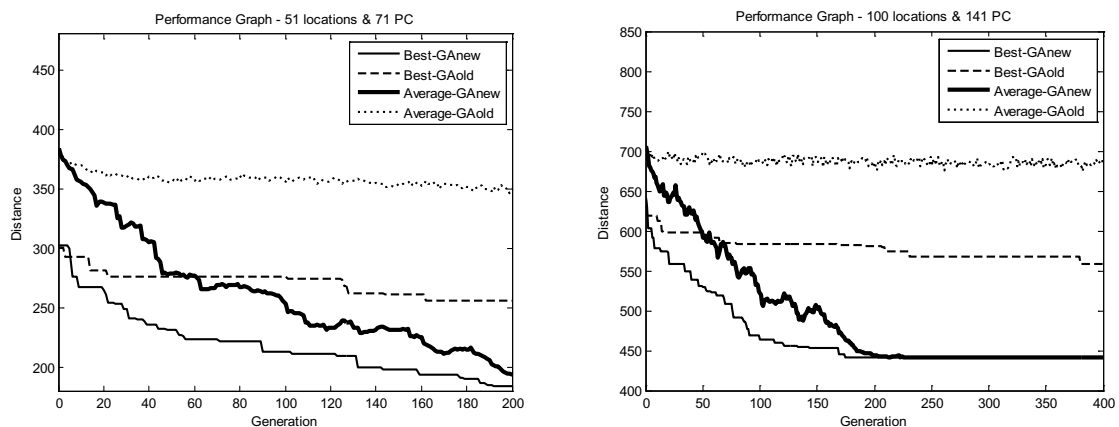


Fig. 4. Performance graph for (a) 51 locations and 71 precedence constraints; (b) 100 locations and 141 precedence constraints

To examine the robustness and the stability of the proposed approach, the experiment is carried out on another large scale test problem which 100 locations with 141 precedence constraints. In order to speed up the computation time, smaller population size and maximum number of generation (for stopping criteria) are used for both experiments (GAold and GANew algorithms). Note that the population size, maximum number of generation, the probability of crossover and probability of mutation are set similar for both algorithms. The parameter settings are as follows, Popsiz=100, P_c=0.9, P_m=0.01 and maximum number of generation is 400.

The performance graph in Fig. 4(b) demonstrates the best and the average travelling distance found by the algorithms in each generation. For the GAnew algorithm, the best and average travelling distance steadily reduced over generations. The results of the computational experiments for both test problems using both algorithms are summarized in Table 4.

From the performance graph it is observed that the 'Average' curve for GAold seems to be stagnant which was apparently being seen in both test problems. This indicates that the crossover technique used in the GAold procedure was not capable to introduce the new fittest offspring and therefore the search space contains almost identical individuals which have the same characteristics as their parents. GAold algorithm showed a very slow progress and finally fails to converge at the best/optimal solution. For all the experiments done, the progress of the curve in the performance graphs indicates that the combinations of linear order crossover and inversion mutation are able to preserve good chromosomes and add new chromosomes in the population. The use of inversion mutation on the other hand provides sufficient variance in fitness across the population to drive further evolution.

Table 4. Summary of results for all tests using GAnew and GAold algorithm

Test Problem	GAnew			GAold		
	Gen#	Best	Convergence time (sec)	Gen#	Best	Convergence time (sec)
51 tasks & 71 PC	193	184	650	163	256	1109
100 tasks & 141 PC	174	441	928	381	559	1240

In order to measure the diversity of the individual fitness in the population, the standard deviation is calculated and plotted. Fig. 5 shows the standard deviation versus the number of generations for 51 locations and 71 precedence constraints. The standard deviation for GAold algorithm does not decrease and looks slightly increase towards the generations. This indicates that the population diversity is maintained, however the algorithm does not introduce new fitter individuals in the population. On the other hand the standard deviation for the proposed algorithm (GAnew) steadily decreases towards the generations. This implies that the population diversity is reduced and the population contains a large number of fitter individuals.

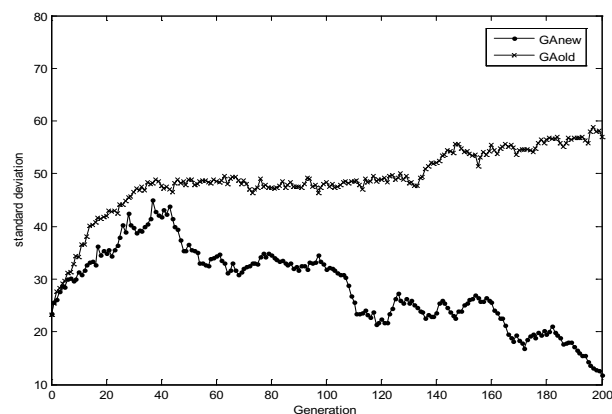


Fig. 5. Graph of standard deviation vs. generation for 51 locations and 71 precedence constraints

6. Conclusion

In this paper, we have developed a genetic algorithm procedure for vehicle routing problem with precedence constraints in which route repair based topological sort is inserted in the procedure in order to generate only feasible chromosomes. The proposed algorithm (GAnew) has used 'earliest position' selection of chromosomes in order to reduce iteration time, hence improved genetic algorithm performance. The proposed algorithm has used simple linear order crossover and inversion mutation to introduce new fitter chromosomes from generations to generations in order to prevent premature convergence. In conclusion, it can be stated that the GAnew algorithm had produced better result in terms of performance and the quality of solution.

In this paper, a single vehicle with precedence constraints is considered in the tour. In many real problem situations, more than a single vehicle with multiple constraints present in the tour. In future it is possible to develop an efficient algorithm which can handle multiple vehicle routing problems with precedence constraints. It is also possible to consider multi-objective optimization for vehicle routing problem. This is an interesting subject to explore and a more realistic problem to be tackled as most of the real problems in logistics involve more than one objective to be achieved such as maximizing the profit while minimizing the cost or maximizing customer satisfaction (e.g. on time delivery) while minimizing on-hand inventories. Since the objective may conflict with each other, a sequence that can optimize both objectives at the same time may not exist. Therefore Pareto front must be constructed and can be used to assist the decision making. Furthermore, this type of problem is NP-hard, and obtaining multi-objective genetic algorithm is a practical option.

- Acknowledgements

This work was partially supported by the Universiti Malaysia Pahang. This support is gratefully acknowledged. Thanks are also due to the referees for their valuable comments.

References

- J. K. Lenstra, J., K., & Rinnooy Kan, A., H., G. (1981). *Complexity of vehicle routing and scheduling problems*, 2, 221–227.
- Li F, Golden, B, & Wasil, E. (2007). The open vehicle routing problem: algorithms, large-scale test problems, and computational results. *Computers and Operations Research*, 10, 2918-2930.
- Fagerholt, K., & Christiansen, M. (2000). A traveling salesman problem with allocation, time window and precedence constraints - an application to ship scheduling. *International Transactions in Operational Research*, 7, 231-244.
- Desrosiers, J., Dumas, Y., Solomon, M., M., & Soumis, F. (1995). Network Routing, *Handbook in Operations Research and Management Science*, 35-139.
- Sigurd, M., Pisinger, D., & Sig., M. (2004). Scheduling transportation of live animals to avoid the spread of diseases. *Transportation Science*, 38(2), 197-209.
- Moon, C., Kim, J., Choi, G., & Seo, Y. (2002). An efficient genetic algorithm for the traveling salesman problem with precedence constraints, *European Journal of Operational Research*, 140, 606-617.
- Gambardella, L., M., & Dorigo, M. (1997). HAS-SOP: Hybrid ant system for the sequential ordering problem.
- Kotecha, K., & Gambhava, N. (2003). Solving Precedence Constraint Traveling Salesman Problem Using Genetic Algorithm. *In Proceedings of National Conference on Software Agents and embedded System*.
- Mokhlesian, M., Ghomi, S., M., T., F., & Jolai, F. (2010). Economic lot scheduling problem with consideration of money time value. *International Journal of Industrial Engineering Computations*, 1, 121-138.