Applied Soft Computing Journal 75 (2019) 441–460

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Modified variable neighborhood search and genetic algorithm for profitable heterogeneous vehicle routing problem with cross-docking

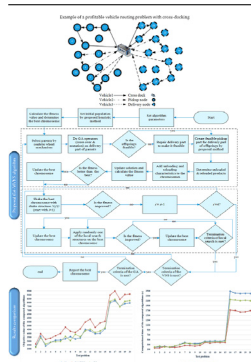
Ali Baniamerian [a], Mahdi Bashiri [a,*], Reza Tavakkoli-Moghaddam [b,c,d]

[a] Department of Industrial Engineering, Shahed University, Tehran, Iran
[b] School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran
[c] Arts et Métiers ParisTech, LCFC, Campus de Metz, France
[d] Universal Scientific Education and Research Network (USERN), Tehran, Iran

HIGHLIGHTS

- A practical version of vehicle routing problem integrated with cross docking is modeled and defined.
- Three meta-heuristic algorithms, nam -ely a hybrid GA with modified VNS, ABC and SA are designed to solve large-size problems.
- A new heuristic algorithm is developed to generate high quality initial solutions.
- Our proposed model and solution methods are evaluated in a set of real-world inspired instances.
- It was proved that the proposed hybrid algorithm has acceptable performance and is significantly better than the others in this problem.

GRAPHICAL ABSTRACT



ARTICLE INFO

Article history:
Received 25 December 2017
Received in revised form 11 November 2018
Accepted 16 November 2018
Available online 22 November 2018

Keywords:
Cross-docking
Profitable vehicle routing
Hybrid algorithm
Purchasing cost
Selling price

ABSTRACT

This paper considers a profitable heterogeneous vehicle routing problem with cross-docking (PHVRPCD). In the real world, it is not possible to serve all customers and suppliers. Based on the purchasing cost and selling price of the products as well as the resource limitation, they will be in the plan only if it is profitable to serve them, so satisfying all demands is not necessary. Cost reduction has been considered in the previous studies as a main objective while neglecting the total profit. In this study, increasing the total profit of a cross-docking system is the main concern. For this purpose, a mixed-integer linear programming (MILP) model is used to formulate the problem mathematically. A new hybrid meta-heuristic algorithm based on modified variable neighborhood search (MVNS) with four shaking and two neighborhood structures and a genetic algorithm (GA) is presented to solve large-sized problems. The results are compared with those obtained with an artificial bee colony (ABC) and a simulated annealing (SA) algorithm. In order to evaluate the performance of the proposed algorithms, various examples of a real data set are solved and analyzed. The computational results reveal that in the small-size test problems, the hybrid algorithm is able to find optimal solutions in an acceptable computational time. Also, the hybrid algorithm needs less computational time than others and could achieve better solutions in large-size instances.

1. Introduction

Competitiveness in global markets has forced organizations to find better policies to manage their costs. In this competitive environment, distributors and manufacturers try to find ways to reduce

* Corresponding author.
E-mail addresses: a.baniamerian@shahed.ac.ir (A. Baniamerian), bashiri@shahed.ac.ir (M. Bashiri), tavakoli@ut.ac.ir (R. Tavakkoli-Moghaddam).

https://doi.org/10.1016/j.asoc.2018.11.029
1568-4946/

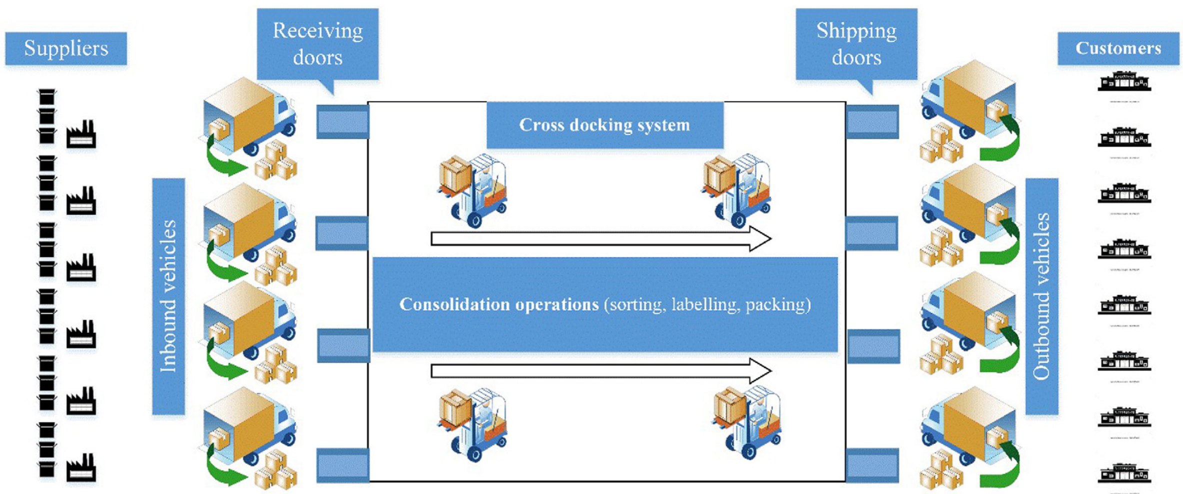


Fig. 1. A cross-docking system.

logistics and production costs. In addition, they want to improve servicing to satisfy their customers' needs simultaneously. It is known that about 30% of supply chain costs is incurred in the distribution part [1]. Therefore, an efficient distribution strategy can significantly help companies to achieve their predetermined goals. Choosing the right strategy helps companies to improve their competitiveness in global markets. An efficient distribution strategy called cross-docking is used to manage the materials flow at a lower cost and higher customer satisfaction. Cross-docking is a new distribution strategy in logistics, which has been recently used by companies practically and widely [2]. Crossdocking has been implemented by the following companies: WallMart, an American multinational company which operates grocery and discount department stores and large number of hypermarkets [3], Toyota Motor Corporation, a Japanese automotive manufacturer [4], United Parcel Service, one of the largest package delivery companies in the world [5], amongst others. The prevalence of the cross-docking strategy attracts many researchers to study and develop all aspects of this strategy in recent years. There are some advantages that make cross-docking an efficient strategy in comparison with traditional warehousing, such as cost reduction (i.e., inventory, holding and transportation), lower inventory level and better turnover, less storage space, shorter lead time and high customer satisfaction [1,6]. In a cross-docking system, the products are received from suppliers to a cross-dock. Different products are consolidated according to their destinations are consolidated (i.e., sorting, labeling and packing), then products with same destination are transferred to customers as soon as possible [6].

The consolidation process leads to a better utilization of the truck capacity. Long storage in a cross-docking system is not allowed and products should be transferred to destinations within less than 24 h [52]. In a cross-dock, there are no facilities for long storage of products. In cases with certain and relatively constant demands, a cross-docking strategy is absolutely applicable. Uncertainty on demands may result in an unsuccessful implementation of this system [29]. There are three parts in a cross-docking strategy to control the materials flow: receiving, sorting and shipping. Therefore, the implementation of a cross-docking system removes storing and order picking from the five usual parts of distribution centers.

In a cross-docking system, a set of vehicles departs from the cross-dock to suppliers' region in order to pick up products. The vehicles pick up products from one or more suppliers and bring them to the cross-docking facilities. Products are carried by automatic conveyors into the cross-dock for the consolidation process. After sorting products by destinations, they are moved to related shipping doors. Then, they are finally loaded into outbound vehicles and delivered to destinations as soon as possible. Fig. 1 illustrates a cross-docking system.

There are some key decisions that are important to better implement a cross-docking system. A vehicle routing problem (VRP) as a well-known combinatorial optimization problem, is an important field of distribution management [45]. Integrating the VRP with cross-docking (VRPCD) as an important decision can significantly increase the efficiency. In addition, companies should improve their servicing to satisfy clients. One of the most important factors in the success of the companies is the profit resulting from their strategies. In this paper, we introduce a profitable heterogeneous VRP with cross-docking (PHVRPCD) for scenarios where servicing of all customers is not necessary. To the best of our knowledge, the profitable version of the VRP has not been considered with cross-docking in the literature so far. Determining a subset of suppliers and customers to be served is the aim of the PHVRPCD.

In the real world, to satisfy customers' demands, we face with many constraints, such as resource limitations. As a result of the limited resources, it may not be possible to satisfy all customers. We should select the best strategy to utilize all resources and gain the best result. Each product in each node has a specific purchasing cost and selling price. Additionally, according to the budget, capacity and customers' time window limitation, the system should decide to serve selected suppliers and customers. A mixed-integer linear programming (MILP) model is presented to maximize the total profit of the system. A new hybrid meta-heuristic algorithm based on modified variable neighborhood search (MVNS) and a genetic algorithm (GA) is developed to solve large-sized problems. The Genetic Algorithm, as one of the most well-known evolutionary algorithms, has a great performance with different optimization problems [53].

It has a great ability of global exploration in the solution space but in its plain form, it does not employ a local search and is relatively slow, so it needs some modifications to improve its performance to reach the optimal solution in the region of convergence [53]. The main advantage of hybridizing global search in form of a GA with a local search is the improvement in the convergence speed to optimal and near optimal solutions. We

Table 1
Review on VRP Variants.

Studied VRP variants	Short definition	Introduced by
Capacitated VRP (CVRP)	Products are delivered by vehicles which have limited capacity	Dantzig and Ramser [7]
Time-dependent VRP (TDVRP)	Considers the travel time between customers and depots depends on the distance and the time of day (e.g. rush hours, weather conditions)	Cooke and Halsey [8]
Pickup and delivery problem (PDP)	A number of goods need to be moved from certain pickup nodes to other delivery ones	Wilson et al. [9]
Multi-depot VRP (MDVRP)	More than one depots are considered and each customer is assigned to one depot by related vehicle	Tillman [10]
Stochastic VRP (SVRP)	Some factors are not certain like demand, travel time and etc.	Tillman [10]
Location routing problem (LRP)	Determining location of depots by opening a single or a set of depots and assigning vehicles and routes to the opened depots	Watson-Gandy and Dohrn [11]
Periodic VRP (PVRP)	Planning for periods of time: visiting customers in different day combinations	Beltrami and Bodin [12]
Dynamic VRP (DVRP)	Online planning of routing according to the dynamic requests	Speidel [13]
VRP with time windows (VRPTW)	Customers must be visited by vehicles in a limited and specific period of time without deviation	Russell [14]
Inventory routing problem (IRP)	Integrating inventory management and vehicle dispatching	Bell et al. [15]
Fleet size and mix vehicle routing problem (FSMVRP)	Heterogeneous vehicles are used by different capacity, speed, equipment, and cost	Golden et al. [16]
Generalized VRP	Customers are clustered and one of each cluster members should be visited once	Tsiligirides [17]
Multi-compartment VRP (MCVRP)	Each vehicle deliver demand of one customer and the demand of a customer for one given product are not splittable. Also each customer request one or multiple types of products	Lawler et al. [18]
Site-dependent VRP	Each customer must be visited by only one set of vehicle types	Nag [19]
Split-delivery VRP (SDVRP)	Each customer can be visited by more than one vehicle	Dror and Trudeau [20]
Fuzzy VRP (FVRP)	Some factors like demand and time windows are ambiguous and should be defined by fuzzy logic	Cheng et al. [21]
Open VRP (OVRP)	Vehicles are not needed to return to the depot after serving all customers	Sariklis and Powell [22]
VRP with Loading constraints (VRPLC)	A pickup and delivery problem which layout of items onto the vehicles are considered	Iori et al. [23]
Selective VRP (SVRP)	Select only profitable pickup and delivery points to visit	Privé et al. [24]
VRP in reverse logistics (VRPRL)	Considering backward flow of materials	Fleichmann et al. [25]
Green VRP (G-VRP)	Concerning energy consumption	Kara et al. [26]
Pollution-routing problem (PRP)	Transportation planning for reducing of pollution	Sbihi and Eglese [27]
Multi-echelon VRP (MEVRP)	Transferring freights in a multi-echelon distribution system with intermediate depots	Crainic et al. [28]

made some modification in the basic structure of the variable neighborhood search (VNS) that we use in our hybrid method to decrease the computational time. Our results show the efficiency of the proposed hybrid algorithm comparing with the artificial bee colony (ABC) and simulated annealing (SA) algorithms. The paper structure is organized as follows. Section 2 presents a review on the literature in our field. The problem description and mathematical formulation are presented in Sections 3 and 4, respectively. Section 5 includes our proposed algorithms. Computational results are described in Section 6. Finally, we draw our conclusions in Section 7.

2. Literature review

2.1. Vehicle routing problem

The VRP is a well-known problem, which consists of designing and planning the best strategy to meet the customers' demands. The classic VRP is defined as an optimization problem to find the most suitable routes to serve customers' demands by a fleet of vehicles [54]. Vehicles start related tours from a distribution center and transfer products to customers. The goal of the problem is to design a tour for each vehicle to satisfy all demands with minimum transportation cost [55]. In the literature, a large number of VRP

Table 2
Brief review on the VRPCD literature.

Articles	Model Type	Objective function	Solution method	Time windows	Direct shipment	Variable vehicles	Heterogeneous vehicles	Single CD	Network Of CD	Splittable freight	Single product
Lee et al. [29]	NLP	Min (travel cost + vehicle fixed cost)	TS			•		•			•
Wen et al. [30]	MIP	Min (travel time)	TS	•				•			•
Liao et al. [31]	–	Min (travel cost + operation cost)	TS			•		•			
Musa et al. [32]	IP	Min (travel cost)	ACO		•				•		
Vahdani et al. [33]	NLP	Min (travel cost + vehicle fixed cost)	Hybrid (PSO+SA+VNS)			•		•			•
Tarantilis [34]	–	Min (travel distance)	Adaptive multi restart TS	•				•			•
Santos et al. [35]	IP	Min (routs cost + load changing)	B&P		•			•			•
Moghadam et al. [36]	MINLP	Min (travel cost + load and unload cost	SA& Hybrid (SA+ACO)	•				•		•	•
Fakhrzada and Esfahanib [37]	MIP	Min (travel cost + operation cost + earliness & tardiness)	TS& VNS	•				•			
Dondo and Cerdá [38]	MILP	Min (travel cost + makespan)	Sweep based approach					•			•
Mousavi et al. [39]	Two phase MIP	Min (cross-dock numbers + travel cost + operation cost)	Fuzzy programming			•			•		•
Dondo and Cerdá [40]	MILP	Min (travel cost + makespan + distribution time)	Sweep based approach					•			•
Agustina et al. [41]	MILP	Min (earliness & tardiness + inventory holding cost +travel cost)	CPLEX	•				•			•
Dondo and Cerdá [42]	MILP	Min (travel cost + makespan + distribution time)	Sweep based approach				•	•			•
Ahmadizar et al. [43]	MINLP	Min (travel cost + purchasing cost + holding cost)	Hybrid GA				•		•		•
Mokhtarinejad et al. [44]	MIP	Min (travel cost + travel time)	GA and Machine-learning-based		•				•		
Yu et al. [45]	MILP	Min (Vehicle hiring cost and transportation cost).	SA			•		•			•
Goodarzi and Zegordi [46]	MINLP	Min (+shipment+ fixed+ variable+ routing+ operational cost)	BBO			•			•		
Grangier et al. [47]	MIP	Min (total delivery cost)	LNS+SPM		•			•			•
Mousavi and Vahdani [48]	MILP	Min (holding cost+ transportation cost)	SAICA	•					•		
Nikolopoulou et al. [49]	MIP	Min (total distance traveled)	AMP+TS					•		•	•
Yin and Chuang [50]	NLP	Min (travel cost + vehicle fixed cost)	AMABC			•		•			•
Baniamerian et al. [51]	MIP	Min (transportation cost + early/tardy deliveries)	Two phase GA	•				•			•
This work	MILP	Max (total profit)	Hybrid (GA+MVNS), SA & ABC	•		•	•	•			•

NLP: Non-linear programming; MIP: Mixed-integer programming; IP: Integer programming; MINLP: Mixed-integer non-linear programming; TS: Tabu search; ACO: Ant colony optimization; PSO: Particle swarm optimization; B&P: Branch and price; BBO: Biogeography-based Optimization; LNS: Large neighborhood search; SPM: Set partitioning and matching problem; SAICA: Self-adaptive imperialist competitive algorithm; AMP: Adaptive memory programming; AMABC: Adaptive memory artificial bee colony.

variants with more complex characteristics and constraints have been investigated over 50 years by researchers. Table 1 indicates the studied VRP variants during these years. The integration of the VRP with different distribution strategies to tackle more real world applications and logistic systems is studied by many researchers. One of the recent applications of this approach is to integrate the VRP with a cross-docking system.

2.2. A cross-docking system

A multi-echelon vehicle routing problem (MEVRP) as an extension of the VRP is to design and plan the materials flow in a multi-echelon distribution system, in which products are transferred from origins to destinations through intermediate depots [56]. The goal of the problem is to minimize the total transportation cost of the system. Multi-echelon systems are widely used practically in many companies, such as newspaper distribution and postal services. Cross-docking is one of the most common instances of multi-echelon distribution strategies, in which products are delivered to customers through an intermediate depot, namely cross-dock. It is an effective strategy to achieve logistic goals, which attracted the attention of many researchers in the last decades. Many key decisions have been investigated in the literature so far. Based on the decision level, the problems are categorized as strategic, tactical and operational. Boysen and Fliedner [57], Van Belle et al. [58] and Buijs et al. [59] classified the problems on cross-docking. Buijs et al. [59] proposed a framework to specify the inter-dependencies between different cross-docking problem aspects and clarified future studies based on inputs and outputs of each problem. According to the categories introduced by Van Belle et al. [58], decisions in cross-docking can be categorized as follows: (1) Location of cross-docks (2) Layout design (3) Cross-docking network design (4) Vehicle routing (5) Dock door assignment (6) Truck scheduling and temporary storage [60].

Freights in a cross-docking system need to be picked up at various origins and have to be delivered to related destinations after internal processes (i.e., consolidation) at the cross-dock. So considering vehicle routing and cross-docking simultaneously as an integrated problem can result in efficient distribution networks. There are a few papers considering the two above-mentioned fields simultaneously. Lee et al. [29] aimed to minimize the transportation and fixed costs of used vehicles and Simultaneous arrival at the cross-dock was assumed. They formulated the problem in an integer programming model and used a Tabu search (TS) algorithm to find solutions. Liao et al. [31] used the same mathematical model and proposed an improved version of the TS to solve it. Up to 36% improvement and less computational time was reported by their proposed algorithm. Wen et al. [30] considered a pickup and delivery problem with assumption of consolidation process. They assumed that a consolidation process is executed at the cross-dock after a pickup phase. So, products are labeled, packed and sorted according to their destinations. Orders were not splittable and a time window was defined for all suppliers and customers. They used a mixed integer linear programming (MILP) to model the proposed problem mathematically. The goal of their proposed model was to minimize the travel time solved by a TS algorithm for large instances. Baniamerian et al. [51] considered a problem which focused on improving customer satisfaction in the distribution centers which the arriving time of products to the customers are an important factor. They proposed a mixed integer programming (MIP) model to minimizing traveling and early/tardy deliveries costs to increase customer satisfaction. They proposed a two phase genetic algorithm to solve large problems. Their proposed method led to 86.6%–100% customer satisfaction.

In the cross-docking literature, there are many characteristics in the VRP variants such as time windows, homogeneity or heterogeneity of vehicles, split delivery, direct shipment, single or multi product and the like. Table 2 indicates some integrated VRP characteristics with cross-docking in the literature. Based on our best knowledge, a profitable vehicle routing problem (PVRP) has not been considered. As illustrated in this table, the main objective of the published papers is usually to minimize the transportation costs while maximizing the total distribution profit has been neglected in the previous studies. In this paper, an MILP model is presented for a profitable vehicle routing problem with cross-docking and a heterogeneous fleet of vehicles is used with different speeds and capacities. Furthermore, three meta-heuristic algorithms are proposed to solve the problem.

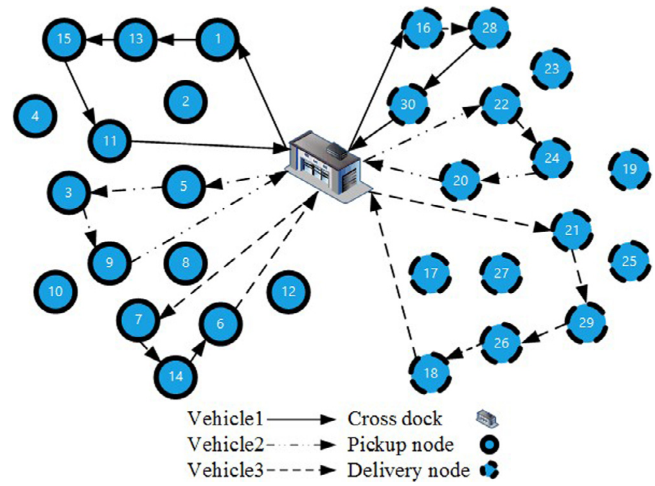


Fig. 2. Example of a profitable vehicle routing problem with cross-docking.

3. Problem definition

This paper considers a three-echelon vehicle routing problem with cross-docking, which includes suppliers, cross-docks and customers. A heterogeneous fleet of vehicles move collected products from suppliers to customers through a cross-dock. We have many suppliers and customers that should be visited; however according to the resource limitation, satisfying all of them is not possible. So the most profitable customers and suppliers are selected from the available customers and suppliers according to their selling/purchasing prices and transportation costs. It is clear that the total purchasing cost of all selected products must not exceed the budget limitation. Fig. 2 represents an example of the problem including 15 suppliers, 15 customers and a cross-dock. Inbound vehicles start their tours from the cross-dock and pick up products from the selected suppliers and comeback to the cross-dock and unload them to the receiving doors. Then, outbound vehicles reload products and deliver them to customers of the selected suppliers after a consolidation process at the cross-dock.

3.1. Assumptions

- A set of heterogeneous vehicles with different capacities and speeds deliver products from suppliers to customers.
- Orders are not splittable.
- Products are bought from suppliers and each one has a specific price. In addition they are sold to customers.
- The budget is limited, so purchasing all orders from suppliers may not be possible.
- Vehicles must start/end their tours from/at the cross-dock in both pickup and delivery parts.
- All products must be unloaded to the cross-dock completely after pickup part without delay.

- Pre-emption during loading/unloading processes is not allowed.
- When the pickup and delivery of a product happens with same vehicle, the consolidation process is not necessary.
- All products must be transferred to the destinations at the end of the time horizon, so there is no inventory at the cross-dock.
- There are enough receiving and shipping doors for unloading and reloading products.

4. Mathematical formulation

An MILP formulation with the following notations is presented for the PHVRPCD problem.

$P = \{1, \ldots, n\}$ stands for the set of pickup nodes and $D = \{n+1, \ldots, 2n\}$ denotes the set of delivery nodes. Each request i is identified by the node pair $(i, i+n)$, where i is the pickup node and $i+n$ is the associated delivery node. Let R be the set of inbound/receiving doors and S be the set of outbound/shipping doors. The set K denotes all available vehicles in the cross-dock. Define $N = P \cup R \cup S \cup D$.

Parameters:

M	an arbitrary big number, used for relating binary routing variable to vehicles departure scheduling.
c_{ij}	traveling cost of node i to j
w_k	average speed of vehicle k ($k \in K$)
d_{ij}	travel distance between node i and j
Pu_i	purchasing price of request i ($i \in P$)
Se_i	selling price of request node i ($i \in D$)
B	available budget
$[a_i, b_i]$	time windows for node i ($i \in N$)
d_i	amount of demand of request i ($i \in P$)
T_k	capacity of vehicle k ($k \in K$)
F	*fixed* set up time for each vehicle
V	variable time for unloading/reloading of a unit product in the cross-dock.
n	number of all available nodes

Variables:

x_{ijk}	binary variable for traveling from node i to j by vehicle k ($i, j \in N$)
u_{ik}	binary variable for unloading request i by vehicle k ($i \in P$, $k \in K$)
r_{ik}	binary variable for reloading request i by vehicle k ($i \in P$, $k \in K$)
q_k	binary variable for necessity of unloading process by vehicle k ($k \in K$)
q'_k	binary variable for necessity of reloading process by vehicle k ($k \in K$)
l_{ik}	departure time of vehicle k from node i ($i \in N$, $k \in K$)
f_k	end time for unloading process of vehicle k ($k \in K$)
s_k	start time for reloading process of vehicle k ($k \in K$)
g_i	end time for unloading process of request i ($i \in P$)

The proposed MILP model is formulated as follows:

$$\text{Max}\, z = \sum_{i\in D}\sum_{j\in D\cup S}\sum_{k\in K} d_i Se_i x_{ijk} - \sum_{i\in P}\sum_{j\in P\cup R}\sum_{k\in K} d_i Pu_i x_{ijk} - \sum_{i\in N}\sum_{j\in N}\sum_{k\in K} c_{ij} x_{ijk} \quad (1)$$

$$\sum_{j\in P\cup R}\sum_{k\in K} x_{ijk} \le 1 \qquad \forall i \in P, i \ne j \quad (2)$$

$$\sum_{j\in D\cup S}\sum_{k\in K} x_{ijk} \le 1 \qquad \forall i \in D, i \ne j \quad (3)$$

$$\sum_{j\in P} x_{Rjk} \le 1 \qquad \forall k \in K \quad (4)$$

$$\sum_{j\in D} x_{Sjk} \le 1 \qquad \forall k \in K \quad (5)$$

$$\sum_{i\in P} x_{iRk} \le 1 \qquad \forall k \in K \quad (6)$$

$$\sum_{i\in D} x_{iSk} \le 1 \qquad \forall k \in K \quad (7)$$

$$\sum_{i\in P}\sum_{j\in P\cup R} d_i x_{ijk} \le T_k \qquad \forall k \in K, i \ne j \quad (8)$$

$$\sum_{i\in D}\sum_{j\in D\cup S} d_i x_{ijk} \le T_k \qquad \forall k \in K, i \ne j \quad (9)$$

$$\sum_{i\in P\cup R} x_{ihk} = \sum_{j\in P\cup R} x_{hjk} \qquad \forall h \in P, \forall k \in K, i \ne j \quad (10)$$

$$\sum_{i\in D\cup S} x_{ihk} = \sum_{j\in D\cup S} x_{hjk} \qquad \forall h \in D, \forall k \in K, i \ne j \quad (11)$$

$$l_{jk} \ge l_{ik} + \left(\frac{d_{ij}}{w_k}\right) - M(1 - x_{ijk}) \qquad \forall i, j \in N, \forall k \in K, i \ne j \quad (12)$$

$$l_{jk} \le M x_{ijk} \qquad \forall i, j \in N, \forall k \in K, i \ne j \quad (13)$$

$$a_i \le l_{ik} \le b_i \qquad \forall i \in N, \forall k \in K \quad (14)$$

$$\sum_{i\in P}\sum_{j\in P\cup R}\sum_{k\in K} d_i Pu_i x_{ijk} \le B \quad (15)$$

$$\sum_{j\in(P\cup R)}\sum_{k\in K} x_{ijk} = \sum_{j\in(D\cup S)}\sum_{k\in K} x_{(i+n)jk} \qquad \forall i \in P \quad (16)$$

$$u_{ik} - r_{ik} = \sum_{j\in(P\cup R)} x_{ijk} - \sum_{j\in(D\cup S)} x_{(i+n)jk} \qquad \forall i \in P, \forall k \in K \quad (17)$$

$$u_{ik} + r_{ik} \le 1 \qquad \forall i \in P, \forall k \in K \quad (18)$$

$$\frac{1}{M}\sum_{i\in P} u_{ik} \le q_k \le \sum_{i\in P} u_{ik} \qquad \forall k \in K \quad (19)$$

$$f_k = l_{Rk} + F \times q_k + V \sum_{i\in P} d_i u_{ik} \qquad \forall k \in K \quad (20)$$

$$s_k \ge f_k \qquad \forall k \in K \quad (21)$$

$$s_k \ge g_i - M(1 - r_{ik}) \qquad \forall i \in P, \forall k \in K \quad (22)$$

$$g_i \ge f_k - M(1 - u_{ik}) \qquad \forall i \in P, \forall k \in K \quad (23)$$

$$\frac{1}{M}\sum_{i\in P} r_{ik} \le q'\_k \le \sum_{i\in P} r_{ik} \qquad \forall k \in K \quad (24)$$

$$l_{Sk} = s_k + F \times q'\_k + V \sum_{i\in P} d_i r_{ik} \qquad \forall k \in K \quad (25)$$

Eq. (1) maximizes the total profit of the system including the terms of the total revenue, purchasing and traveling costs, respectively. Constraints (2) and (3) imply that each selected node is visited once. Constraints (4) and (5) determine that it is not necessary to use all available vehicles. Each inbound and outbound vehicle should start from the cross-dock, if required. Constraints (6) and (7) imply that the last node of each tour is the cross-dock. In the other words, inbound and outbound vehicles must return to the cross-dock. Constraints (8) and (9) specify that the capacity of the vehicles is limited. Constraints (10) and (11) show the continuity of tours. It means, remaining in the nodes is not allowed and a vehicle which travels from node i to h, it should

1	3	10	6	2	5	9	8	4	7	pickup
1	1	1	1	1	0	1	1	1	0	
5	7	2	9	8	1	6	3	10	4	delivery
0	0	1	1	1	1	1	1	1	1	

Fig. 3. Solution representation for an example with 8 suppliers/customers and 3 vehicles.

Phase 1 (Delivery tours)

Step 1

1.1. Determine the set of minimum required vehicles (K) to visit all nodes.

1.2. Sort vehicles according to their index $k_1, k_2 \ldots, k_{min}$.

1.3. Start from the first vehicle and determine all possible routes which it can move without time window, capacity and budget violation.

Step 2

2.1. Calculate the ratio of transportation cost to the minimum transportation cost of routes between the current node and next possible nodes in delivery process (α) .

2.2. Calculate the ratio of transportation cost to the minimum transportation cost of corresponding routes (between corresponding node of the current node and corresponding nodes of next possible nodes) in the pickup process (β) .

2.3. Calculate the ratio of selling price to the purchasing cost of the next possible nodes (μ) .

Step 3 (Candidate list)

3.1. Calculate the selection criterion.

$Q = \mu - \beta - \alpha$

3.2. Make a candidate list of routes which their selection criteria are more than a predefined value of Q' .

3.3. Select a route randomly from the candidate list.

Step 4

4.1. Repeat the steps 2 and 3 until the capacity of the vehicle is reached.

4.2. Select next vehicle and repeat previous steps until last vehicle.

Phase 2 (Pickup tours)

Step 1

1.1 make a copy from delivery tours for their related pickup tours. Vehicles transfer products to corresponding nodes without any consolidation at the cross-dock.

Phase 3

Step 1

1.1. Remove one vehicle from set K and call it as K'.

1.2. Repeat phase 1 and 2.

1.2. Repeat previous steps of phase3 while $|K'| \geq \frac{|K|}{2}$.

Fig. 4. Proposed heuristic method to generate initial solutions.

continue its route from node h. Constraints (12) and (13) determine the traveling time between two consecutive nodes by the same vehicle. Constraint (14) specifies hard time windows limitation. Constraint (15) ensures that the total purchasing cost of products must not exceed the available budget. According to the constraints (16)–(18), if vehicle k pickup order i and deliver to i+n by itself, a consolidation process is not necessary. If vehicle k picks up order i and does not deliver it to i+n, must unload this order i at the cross-dock. If vehicle k does not pick up order i but should deliver it to i+n, must reload it from the cross-dock. Constraint (19) specifies loading/unloading process by vehicles. Constraint (20) determines the total time of loading or unloading process for each vehicle. Constraints (21)–(23) ensure that only after the completion of unloading process of all products, the reloading process can be started. Constraints (24) and (25) show the reloading process like constrains (19) and (20)

5. Solving methods for the PHVRPCD

5.1. Clustering method

A clustering method is an efficient method to solve different vehicle routing problems and widely used by researchers in the VRP literature [61]. There are variant algorithms to cluster nodes based on their closeness to each other. In this section, a K-means clustering method is used to cluster suppliers and customers for both VRPCD and PHVRPCD problems. Since we need to specify the number of clusters to use k-means, it is assumed that these were taken equal to the number of vehicles from the exact solution. The results of our experiments on different instances shows that the output of the K-means clustering method is completely the same as the obtained tours of solving the VRPCD by an exact method. So this method can be applied efficiently to solve large-sized instances of the integrated vehicle routing with cross-docking problems as well as vehicle routing problems while this method is not suitable to apply on the PHVRPCD and the result of clustering is so different from an exact solution. The optimal decision in the PHVRPCD is not just based on the transportation cost and the distance between nodes. So developing efficient algorithms to solve large-sized problems based on the mentioned characteristics is important for the PHVRPCD.

5.2. Metaheuristics for the PHVRPCD

The implementation of evolutionary algorithms for optimization problems plays an important role to solve complex and large-sized problems, which cannot be solved by commercial optimization software in a reasonable computational time [62]. In this paper, three metaheuristic algorithms are designed for effective searching in the solution space of the proposed problem and finding near-optimal solutions in the large-sized instances. A novel genetic algorithm hybridized with modified VNS (GA-MVNS), an artificial bee colony (ABC) algorithm and simulated annealing (SA) are proposed algorithms to solve the PHVRPCD problem.

1. Initialization

 Define the set of neighborhood structures N_k, *for* $k = 1,\ldots,k_{\max}$ that will be used in search;

 Choose a stopping condition, and find the initial solution x

2. Repeat until stopping condition is met

  Set $k \longleftarrow 1$

  3. Repeat until $k = k_{\max}$

   3.a. Shaking: Generate point x' at random from the k^{th} neighborhood of x ($x' \in N_k(x)$)

   3.b. Local search: Apply some local search methods with x' as initial solution

    to obtain a local optimum given by x''

   3.c. Neighborhood change:

    if the local optimum x'' is better than x then $x \longleftarrow x''$ and continue

    the search with N_1 $(k \longleftarrow 1)$; Otherwise, set $k \longleftarrow k+1$

Fig. 5. Pseudo code of the basic VNS.

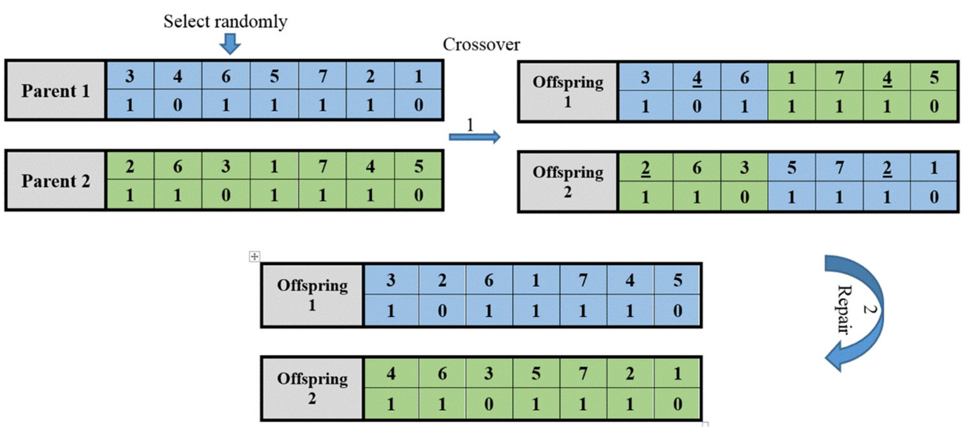


Fig. 6. Crossover operator for the delivery part.

5.2.1. Solution representation

The solution representation of the proposed problem includes four parts. Two parts are for the pickup and delivery process, and two parts determine which nodes should be visited. Fig. 3 shows the proposed solution representation. The first and second strings represent the tours of the pickup part, and the others represent the tours of delivery part. The first string specifies the suppliers' visiting sequence. Considering a pickup part with 8 pickup nodes P $= \{1, \ldots, 8\}$, genes 1–8 are suppliers/customers, while genes 9 and 10 are separators. The second part of the proposed chromosome is a binary string, which indicates the visited suppliers. Visited suppliers/customers are determined by 1, and others are determined by 0. Fig. 3 shows that the first inbound vehicle collects products from suppliers 1 and 3 ($x_{Rjk} = 1 \big| R = 1, j = 1, k = 1$ & $x_{ijk} = 1 \big| i = 1, j = 3, k = 1$ & $x_{iRk} = 1 | i = 3, R = 2, k = 1$). The second inbound vehicle collects products from suppliers 6 and 2 which are separated by two separators 10 and 9 ($x_{Rjk} = 1 \big| R = 1, j = 6, k = 2$ & $x_{ijk} = 1 \big| i = 6, j = 2, k = 2$ & $x_{iRk} = 1 | i = 2, R = 2, k = 2$). The third inbound vehicle collects products from suppliers 8 and 4 ($x_{Rjk} = 1 \big| R = 1, j = 8, k = 3$ & $x_{ijk} = 1 \big| i = 8, j = 4, k = 3$ & $x_{iRk} = 1 | i = 4, R = 2, k = 3$). The third and fourth strings of the proposed chromosome are for the delivery part. The first outbound vehicle delivers products to the customer 2 ($x_{Sjk} = 1 \big| S = 1, j = 2, k = 1$ & $x_{iSk} = 1 | i = 2, S = 2, k = 1$). The second outbound vehicle delivers products to customers 8, 1, 6 and 3. ($x_{Sjk} = 1 \big| S = 1, j = 8, k = 2$ & $x_{ijk} = 1 \big| i = 8, j = 1, k = 2$ & $x_{ijk} = 1 \big| i = 1, j = 6, k = 2$ & $x_{ijk} = 1 \big| i = 6, j = 3, k = 2$ & $x_{iSk} = 1 | i = 3, S = 2, k = 2$). The third outbound vehicle delivers products to the customer 4. ($x_{Sjk} = 1 \big| S = 1, j = 4, k = 3$ & $x_{iSk} = 1 | i = 4, S = 2, k = 3$)

Throughout the proposed algorithm, sub-chromosomes are regenerated by the eight operators of crossover, mutation, one-point swap, N-point swap, insertion, reversion and shake structures.

5.2.2. Initial solution

All meta-heuristic algorithms need to start their exploration in the solution space from an initial solution. Suitable initial solution leads to better performance of the algorithms to achieve high quality solutions. Therefore, in this paper, a set of initial solutions is generated by an algorithmic method to improve the performance of the proposed algorithms. The procedure of generating initial solutions is shown in Fig. 4. The proposed problem is profitable, so visiting all nodes by using all vehicles may not be economic. Therefore, we generate some initial solutions with a different number of used vehicles.

5.2.3. Genetic algorithm

The GA has been used successfully to approximately solve NP-hard problems [63,64]. Many researchers used GAs to solve problems integrating the VRP and cross-docking and found high-quality

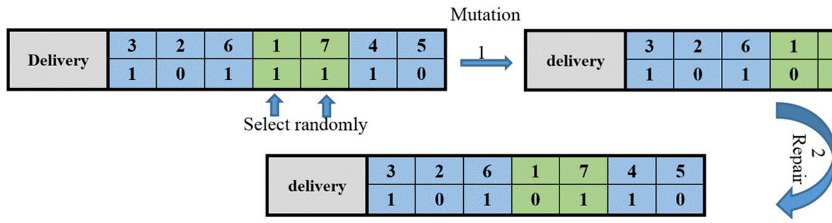


Fig. 7. Mutation operator for the delivery part.

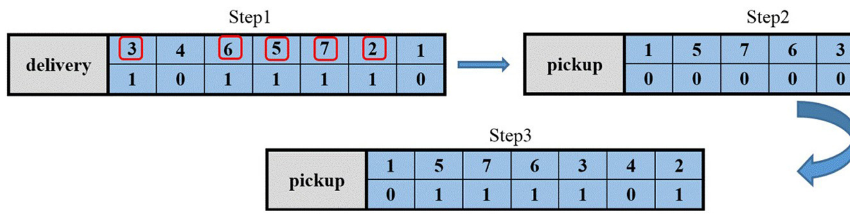


Fig. 8. Example of constructing a pickup part according to its delivery part.

solutions in a reasonable time [43,44]. High robustness, excellent convergence and efficient global search are some advantages of GAs, while weak local search ability, low search efficiency in the final iterations, and high dependency on parameter tuning are some of their disadvantages [65].

5.2.4. Variable neighborhood search algorithm

The VNS algorithm is a local search-based algorithm to systematically search the solution space with more than one type of neighborhood structure [66]. Hansen et al. [67] introduced this algorithm used by a large number of papers to solve combinatorial optimization problems. Additionally, VNS was successfully applied to solve vehicle routing and scheduling problems in a cross-docking system [37]. It starts with an initial solution and improves it by applying operations in two nested levels, called 'shake' and 'local search'. The shaking level is for diversify searching in the solution space while the local search level is for intensifying the search. A shake operator moves to a different part of the solution space by switching from one neighborhood to another and after each switching, a local search operator explores for an improved solution within the current local neighborhood. The pseudo code of the basic VNS is depicted in Fig. 5. The shake and local search operators applied in the VNS in our experiments are the same as the operators discussed later in Sections 5.2.7 and 5.2.8.

5.2.5. Hybrid GA-MVNS algorithm

In the proposed hybrid algorithm, an initial population consisting of feasible pickup and delivery tours is generated by a heuristic algorithm. Solutions are separated into the pickup and delivery parts, and the GA operators are first applied to the delivery part in order to achieve better delivery solutions. Figs. 6 and 7 depict the GA operators on the delivery part of the proposed solution representation. As illustrated in Figs. 6 and 7, during the search process with these operators, some changes make solutions infeasible. Therefore after applying all operators in all algorithms, a repairing process is necessary. Pickup solutions are constructed based on obtained delivery solutions. After applying operators and constructing a complete pickup and delivery solution, consolidation operations are then added to the complete solutions and individuals are updated finally. In other words, in a solution with four strings, all operators are applied to strings 3 and 4 (i.e., delivery part). After generating a new delivery solution, its related feasible pickup solution is then constructed by a heuristic method, which is completely described in Section 5.2.6. The reason of using this approach to search the solution space is that in large-sized problems, finding feasible solutions for integrated pickup and delivery parts is difficult and time consuming. This approach significantly decreases computational time and leads to better results. The proposed hybrid algorithm starts with an initial population generated by a heuristic method. The GA algorithm applies to create new solutions by crossover and mutation operators. Then the best solution obtained from each iteration of the GA is considered as an initial solution of the MVNS. The basic VNS is an efficient and successful method to solve combinatorial optimization problems [67]; however, applying the algorithm is time consuming. So to avoid expending too much computational time, in this paper we propose a modified version of VNS with four shaking and two neighborhood structures.

5.2.6. Constructing pickup parts of solutions according to the delivery parts

As mentioned before, we apply the search operators to the delivery part of the solutions. After generating a new delivery solution by one of the operators, its related feasible pickup part is then constructed by a heuristic method. The process of constructing the pickup part from a delivery part is shown in Fig. 8. As illustrated in this figure, if a customer node in delivery part is selected, its related supplier node will be selected in pickup part. A pickup part is constructed according to its delivery part by the following steps.

Step 1: Identify selected nodes in the delivery part (separators are always selected).

Step 2: Construct the first string of the pickup part randomly by permutation of $(P + K - 1)$.

Step 3: Put 1 in the second string of the pickup part for the selected nodes in Step 1; otherwise, 0.

5.2.7. Shake procedure

In this paper, we design an efficient structure for the proposed VNS. In our version of the VNS, there are four neighborhood structures to move from one solution to another solution. Whenever the proposed VNS starts with the best solution obtained from each iteration of the GA, these four neighborhood structures are applied to diversify the solution. The structures are described in Fig. 9. To further illustration of the shake structures, Figs. 10–13 show examples of applying these structures to a solution.

Step 1 (Shake structure $N_1(S)$)
1.1. Choose two different tours j_1 and j_2.
1.2. Choose two selected nodes i_1 and i_2 from tour j_1 whose transportation cost between them is more than other nodes in j_1.
1.3. Exchange nodes i_1 and i_2 with two randomly selected nodes from tour j_2.
Step 2 (Shake structure $N_2(S)$)
2.1. Choose a set of nodes of the solution.
2.2. Change the condition of the nodes in this way that if each node was selected, change its condition to unselected and if each one was unselected, change its condition to be selected.
Step 3 (Shake structure $N_3(S)$)
3.1. Choose two different nodes i_1 and i_2.
3.2. Reverse the order of nodes between i_1 and i_2 (Reversion)
Step 4 (Shake structure $N_4(S)$)
4.1. Choose two different nodes i_1 and i_2
4.2. Insert node i_1 to the position of i_2 and make them adjacent (Insertion)

Fig. 9. Structure of a shake procedure.

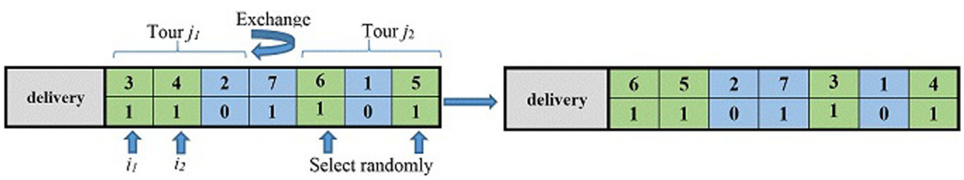


Fig. 10. Shake structure 1 for the delivery part.

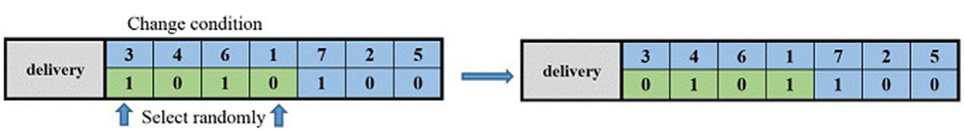


Fig. 11. Shake structure 2 for the delivery part.

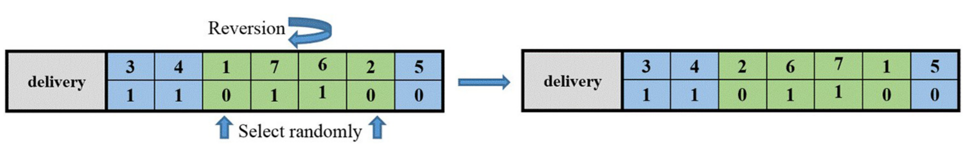


Fig. 12. Shake structure 3 for the delivery part (Reversion)

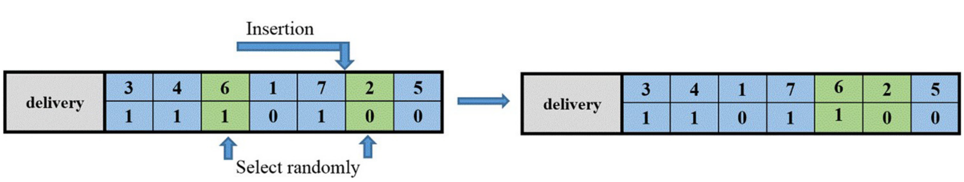


Fig. 13. Shake structure 4 for the delivery part (Insertion).

The shake procedure starts from the first step and it is continued to other shake structures till observing an improvement. Then local search operator starts to explore the neighborhood solution space of the improved solution to find better one. If an improvement was found or the maximum iteration of the local search is reached, the modified VNS is terminated and the solution is reported as the best solution of the current iteration of the proposed hybrid algorithm. In the case that no improvement is made by all four steps in each iteration, the modified VNS is started again from Step 1 until the termination criterion (i.e., maximum number of iterations) is met. This approach significantly decreases the computational time and improve the performance of the algorithm rather than the basic VNS.

5.2.8. Local search procedure

The local search procedure includes two structures to explore the solution space around of the current solution. 1-point and N-point swap operators are used to locally improve the solution. A 1-point swap operator chooses two nodes randomly and exchanges the position of them. An N-point swap operator chooses two different sets with N nodes randomly and exchanges them. Local search operators are indicated in Figs. 14 and 15. When an improved solution obtained from the shake procedure is entered into the local search procedure, one of these operators is randomly selected to apply. The flowchart of the hybrid GA-MVNS algorithm is presented in Fig. 16.

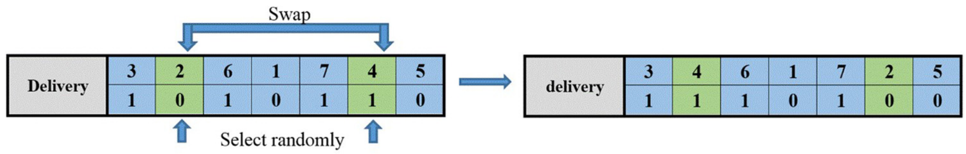


Fig. 14. One-point swap for the delivery part.

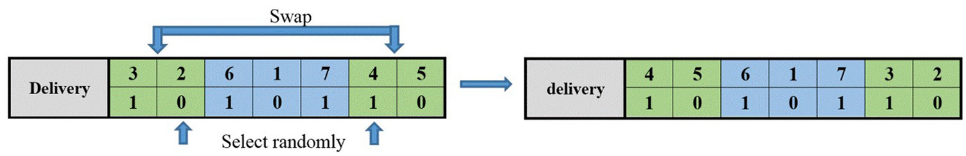


Fig. 15. N-point swap for the delivery part ($N = 2$).

5.2.9. Simulated annealing

Kirkpatrick et al. [68] proposed simulated annealing (SA), which is an iterative and neighborhood-based algorithm to solve combinatorial optimization problems [69]. The SA is a well-known and efficient algorithm in the literature and it has been applied successfully in many cross-docking problems [33,36,45]. SA tries to improve the local search efficiency with a mechanism for escaping from local optima and has successfully been applied to many optimization problems [70]. One of the disadvantages of the SA algorithm is, again, a high dependency on parameter tuning even for the small instances [71]. This algorithm simulates the real process of lowering the temperature by steps. It starts from some initial solutions and search the solution space in a step by step. SA allows to accept worse solutions during the searching process according to a probability of ρ, which is calculated by:

$$\rho = e^{-\frac{\Delta f}{T}}$$

where T is the parameter of temperature. The temperature is decreased in each iteration based on a cooling strategy, which is often a linear equation. Precise settings of some parameters are important in the performance. The proposed SA has the following parameters: the maximum number of iterations ($it_{\max}$), population size (N_{pop}), initial temperature (T_0), final temperature (T_f), number of neighbors generation (nn) and temperature reduction rate (TRR), which is calculated by the following linear equation

$$TRR = \frac{T_0 - T_f}{it_{\max}}$$

In the proposed SA, three search methods (i.e., swap, reversion and insertion) are used to create new solutions. The structure of the proposed SA is shown in Fig. 17.

5.2.10. Artificial bee colony algorithm

The artificial bee colony (ABC) algorithm is a relatively new optimization algorithm introduced by Karaboga [72]. These algorithms are based on behaviors of bees to search regions and find high quality nectar. The groups of bees bring the information of food sources to the hive and share them to the other bees by a waggle dance. After information sharing, the next groups of bees start their search according to the information about food sources. The ABC algorithm is an efficient algorithm since it has some characteristics such as search memory, multi-character to search and local search. The bees in the ABC are divided into three types: employed, onlooker and scout bees. Employed bees explore food sources to gather information and share them with onlookers by a specific waggle dance. Onlooker bees explore the discovered better food sources more preciously. Scout bees are the employed bees that are unable to improve after a specific period of time. Therefore, they leave and start exploring another food source randomly [73]. The ABC algorithm starts by initial solutions and assigns an employed bee to each solution to find better solutions. If the quality of a newly discovered source is better than the previous one, the previous food source is replaced by a new food source. At the end of this process by all employed bees, the information about fitness of all food sources is shared with the onlooker bees. Each onlooker select one food source by a roulette wheel mechanism. An addition, if the quality of the food source is not improved for a predetermined number of iterations, its related employed bee acts as a scout bee and transfers to a new food source randomly and then acts as an employed bee again. This process is repeated until the stopping criteria is met. The ABC algorithm has been applied successfully to solve combinatorial optimization problems [73–79]. Also, the ABC algorithm has been applied to solve different variants of vehicle routing problems with great success [80–83]. It is worth to mention that the ABC algorithm has some disadvantages such as slow convergence speed and low efficiency to escape from local optimal [84].

Since the ABC algorithm is initially designed for continuous numerical problems, we made some modifications to make it suitable for solving proposed PHVRPCD. Therefore, we use the methodology of a random key method proposed by Bean [85]. The random key encodes and decodes a solution with real numbers between (0, 1). In the other words, it can link the continuous and discrete solution spaces by encoding continuous solution to discrete solutions. For a random vector$R = (0.28, 0.11, 0.44, 0.88, 0.39)$, the result of encoding a random key method will be $R' = (2, 1, 4, 5, 3)$, the number 0.11 at position 2 in R is the smallest number in the vector, so it will be encoded to the value 1 at position 2 in R', while 0.28, the second smallest number in R (located at position 1) is encoded to value 2, and so on. An example of random key method for a solution representation of the proposed PHVRPCD is illustrated in Fig. 18.

A feasible solution, is obtained by decoding the vector of food source positions x_i with the random key method. As illustrated in Fig. 18, an employed bee that have food source positions x_i in a 7-dimensional search space is encoded by a random key method to a vector of feasible solutions x'_i with 7 numbers from 1 to 7. Parameters of the ABC algorithm are shown in Table 3. Fig. 19 represents the proposed ABC for the PHVRPCD.

5.3. Fitness evaluation

In order to avoid infeasibility, a penalty method is used to satisfy three main constraints consisting of customers hard time windows, vehicles capacities and available budget, so in the proposed algorithm, Eq. (26) is used as a fitness function.

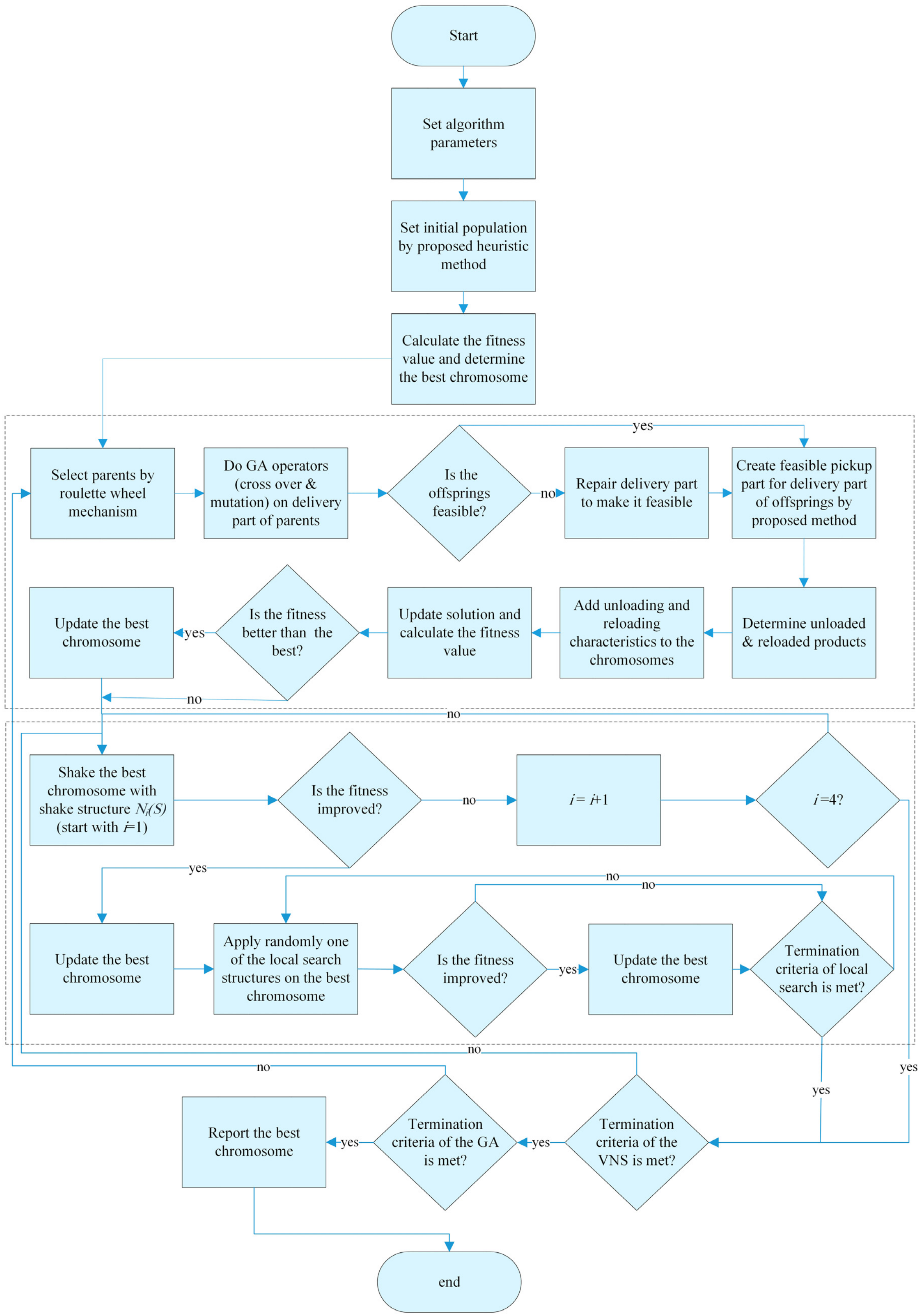


Fig. 16. Flowchart of the proposed hybrid GA-MVNS algorithm.

Step1. (Initialization)
1.1. Set algorithm parameters (it_{max}, T_0, T_f, nn, N_{pop} and TRR)
1.2. Create initial population according to the proposed heuristic method.
Step2. (Evaluation)
2.1. Determine the fitness for all solutions
2.2. Specify the best one
Step3. (Neighbour creation)
3.1. Use roulette wheel to select a solution x
3.2. Create a new delivery part from delivery part of selected solution by one of the operators randomly (Swap, Insertion, and Reversion)
3.3. Create a feasible pickup part for obtained delivery part according to proposed method
3.4. Calculate the fitness value of new solution
3.5. Repeat step3 to create nn neighbours from selected solution
3.6. Determine the best neighbour solution x'_{best}
Step4. (Update)
4.1. If $x'_{best} > x$ replace x with x'_{best}
4.2. If $x'_{best} < x$ replace x with x'_{best} with probability of ρ
Step5. (Temperature reduction)
5.1. Reduce temperature $T = T - TRR$
Step6. (Repeat)
6.1. Repeat previous steps until termination criteria (it_{max}) is met
Step7. (Report)
7.1. Report the best found solution.

Fig. 17. Main structure of the proposed SA.

position	1	2	3	4	5	6	7
delivery	0.4	0.52	0.26	0.11	0.87	0.93	0.2
	1	0	1	0	1	1	0

position	1	2	3	4	5	6	7
delivery	4	5	3	1	6	7	2
	1	0	1	0	1	1	0

Fig. 18. Example of a random key method for the delivery part.

$$\sum_{i\in D}\sum_{j\in D}\sum_{k\in K} d_i Se_i x_{ijk} - \sum_{i\in P}\sum_{j\in P}\sum_{k\in K} d_i Pu_i x_{ijk} - \sum_{i\in N}\sum_{j\in N}\sum_{k\in K} c_{ij} x_{ijk} + \sum_{n=1}^{4}\gamma(TWV_n) + \sum_{m=1}^{2}\lambda(CV_m) + \theta(BV) \tag{26}$$

where γ, λ and θ are the penalty coefficients for time window, capacity and budget violations, respectively. The amounts of violation of these constraints are determined by TWV_n ($n = 1, 2, 3, 4$), CV_m ($m = 1, 2$) and BV for time window, capacity and budget violation, respectively. Therefore, the following equations are defined below:

$$TWV_1 = \sum_{k\in K}\sum_{i\in P} \max\{0, l_{ik} - b_i\}, \tag{27}$$

$$TWV_2 = \sum_{k\in K}\sum_{i\in P} \max\{0, a_{i-} l_{ik}\} \tag{28}$$

$$TWV_3 = \sum_{k\in K}\sum_{i\in D} \max\{0, l_{ik} - b_i\} \tag{29}$$

$$TWV_4 = \sum_{k\in K}\sum_{i\in D} \max\{0, a_{i-} l_{ik}\} \tag{30}$$

where TWV1 and *TWV2* are the total time windows violation (tardy/early delivery) for suppliers, respectively. In addition, TWV_3 and TWV_4 are the same for customers.

According to Constraints (8) and (9), the amount of violation from the inbound and outbound trucks capacity are calculated by:

$$CV_1 = \sum_{k\in K} \max\left\{0, \sum_{i\in P}\sum_{j\in P} d_i x_{ijk} - T_k\right\} \tag{31}$$

$$CV_2 = \sum_{k\in K} \max\left\{0, \sum_{i\in D}\sum_{j\in D} d_i x_{ijk} - T_k\right\} \tag{32}$$

Finally, the amount of violation from budget limitation is calculated by:

$$BV = \max\left\{0, \sum_{i\in P}\sum_{j\in P}\sum_{k\in K} d_i Pu_i x_{ijk} \le B\right\} \tag{33}$$

6. Computational experiments

In this section, numerical results are presented for different instances from a real data set of a Danish logistics consultancy from Copenhagen to show the performance of the proposed algorithms. The data set was introduced by Wen et al. [30] and consists of five sets (a–e) of nodes and their demands. Each set includes 200 pairs of suppliers–customers, which are denoted by their coordination (x, y). For small instances, the mathematical model of the profitable heterogeneous VRP with cross-docking is coded in the General Algebraic Modelling System (GAMS), version 24 using the CPLEX solver. The proposed algorithms are executed on an Intel® Core™ i7 2 GHz CPU and 8 GB of RAM by MATLAB.

6.1. Test problems and parameter setting

In this subsection, used parameters of the problem is presented in details. In the mentioned data set, the time window is considered two hours for each node. The planning for the problem is limited to the time horizon, which is from 6 to 22. Table 4 indicates the considered parameters for solving different instances. In addition, we assume that vehicles drive at a constant speed of 60 km/h and the minimum number of vehicles for a VRPCD are used as available

Table 3
Parameters of the proposed ABC algorithm.

Parameter	Description
Nb	Number of bees (solutions)
$Limit$	Maximum allowable number of iteration without improvement before abandoning a food source
it_{max}	Maximum number of iterations
$x_i;\ i = 1, 2, \ldots Nb$	Index of individual solution
$Ndim$	Dimension of individual solution
x_{ij}	The jth dimension of solution x_i
x_{ub}^j, x_{lb}^j	Upper bound and lower bound of x_{ij}
f_i	Function value of solution x_i
fit_i	Fitness value of solution x_i
v_i	Neighbor solution of solution x_i
$trial(x_i)$	Number of iterations in which solution $x_i$cannot be improved
ω, φ	Random number $0 \leq \omega \leq 1$; $-1 \leq \varphi \leq 1$

Step1. (Initialization)
1.1. Set algorithm parameters
1.2. Create initial solutions x_i, $i=1, 2, \ldots, Nb$ randomly

$$x_{ij} = x_{lb}^{j} + \omega(x_{ub}^{j} - x_{lb}^{j}), \forall j$$

Step2. (Evaluation)
2.1. Determine the fitness for all solutions (fit_i)

$$fit_i = \begin{cases} \dfrac{1}{1+abs(f_i)} & if\ f_i \leq 0 \\ 1+f_i & if\ f_i \geq 0 \end{cases}$$

Step3. (For each employed bee)
3.1. Neighborhood search for x_i

$$v_{ij} = x_{ij} + \varphi(x_{ij} - x_{kj}) \quad \forall i, j \neq k$$

3.2. Calculate fit_i for solution v_i
3.3. If the fitness value of v_i was better than x_i replace x_i with v_i
step4. (For each onlooker bee)
4.1. Select x_i by roulette wheel mechanism and become an employed bee
4.2. Repeat employed step again
Step5. (Scout bees)
5.1. If $trial(x_i) > Limit$
Replace x_i with a new random solution
Step6. (Memory)
6.1. Memorize the best solution
Step7. (Termination)
7.1. Repeat until (it_{max})

Fig. 19. Main structure of the proposed ABC algorithm for the PHVRPCD.

vehicles on the cross-dock. Different examples with 20, 30, 50 and 100 pairs of nodes are randomly selected from five data sets.

In meta-heuristic algorithms, there are some parameters that are significantly effective on the performance of the algorithms. Efficient setting of these parameters can decrease the computational time and improve the quality of searching process. In order to design a robust and efficient algorithm, a good parameter setting is essential, so the best values should be selected in suitable way. In this paper, a response surface methodology (RSM) is used to adjust the parameters of the proposed algorithms to make them more efficient. The RSM method is designed using the Design-Expert 7 software. After analyzing the RSM, the optimum levels for each parameter obtained by solving regression equations are shown in Table 5 for three proposed algorithms. The maximum number of iterations for all proposed algorithms are set to 500 for up to 100 nodes and are set to 1000 for 200 nodes.

6.2. Proposed algorithm evaluation

In the following sections, in order to assess the performance of the proposed algorithms, numerical results of different instances are provided in details. Table 6 indicates the computational experiments of 24 different small test problems. Table 7 indicates

Table 4
Conditions of the considered parameters.

Parameter	Value/distribution
T_k	∼Uniform (30,40)
F	10
V	1
Pu_i	∼Uniform (25,50)
Se_i	∼Uniform (50,100)
B	∼Uniform (6,11) × (2/3) × (number of nodes) × Uniform(25,50)

the computational experiments of 20 different medium and large-sized test problems. Since the problem is NP-hard, it is not possible to guarantee that the medium and large-sized instances have been solved optimally by our metaheuristic methods and the exact algorithms could not be finished within a reasonable time. Therefore in Table 7, only the results of three proposed algorithms are reported. Each instance is run 10 times. The average objective value of 10 runs, best objective values, and the computational time of the best objective value for each algorithm are reported in both tables.

It can be concluded from Table 6, that the hybrid algorithm is able to find optimal solutions in the acceptable computational time. As expected, in some small-sized instances, the GAMS software performs a bit better than the proposed algorithm but when the instance size increases, it becomes clear that the proposed algorithm can find optimal or near optimal solutions with less computational time. The comparison between the results of the proposed algorithms and the exact solutions shows the suitable performance of the proposed algorithm. Note that for better comparison in small-size problems which have been solved by GAMS optimally, the proposed hybrid algorithm continues the search until reaching the optimal solution found by the exact solution, so the termination condition in those instances that their optimal solution are known, is reaching the optimal solution found by GAMS. A time limitation of 10,000 s is considered to solve small-size problems. Figs. 20 and 21 indicate the computational times and the best solutions obtained by the proposed algorithms for different instances. Comparing reported results in Figs. 20 and 21 confirm that the proposed hybrid algorithm performs better than the other two algorithms to reach better solutions in less computational time for all instances. In addition, for a better comparison, Figs. 20 and 21 shows the performance of the proposed algorithms in large size instances. Fig. 20 indicates the computational time for solving large size instances for each algorithm and Fig. 21 shows the same comparison on the best objective value obtained by each one. As indicated in the mentioned figures, the proposed hybrid algorithm has less computational time than others in the most instances. Also, the proposed algorithm could achieve better solutions than

Table 5
Setting values of parameters in different algorithms to solve the PHVRPCD.

Algorithm	Parameter	Optimum value
GA-MVNS	Maximum iteration	500,1000
	Population size	30
	Crossover rate	0.72
	Mutation rate	0.1
	Maximum iteration MVNS	44
ABC	Nb	100
	$Limit$	30
	it_{max}	500,1000
SA	it_{max}	500,1000
	T_0	1000
	T_f	0.1
	nn	5
	N_{pop}	20

Table 6
Small-size problems.

Problem	No. of pickup & delivery nodes	Data set	No. of vehicles	No. of products	GAMS		Proposed hybrid GA-MVNS			Gap
					Run time (s)	Best solution	Run time (s)	Average solution	Best solution	
1	8	A	2	34	0.481	210.87[a]	0.14	210.87	210.87[a]	0
2	8	B	2	41	0.22	197.12[a]	0.14	197.12	197.12[a]	0
3	8	C	2	36	0.28	188.63[a]	0.15	188.63	188.63[a]	0
4	10	A	2	37	0.58	537.47[a]	0.56	537.47	537.47[a]	0
5	10	B	2	53	0.25	844.78[a]	0.43	844.78	844.78[a]	0
6	10	C	2	48	0.31	789.35[a]	0.38	789.35	789.35[a]	0
7	14	A	2	49	4.32	1434.50[a]	0.58	1434.50	1434.50[a]	0
8	14	B	3	71	1.83	1672.67[a]	1.92	1672.67	1672.67[a]	0
9	14	C	3	74	0.63	1094.60[a]	0.3	1094.60	1094.60[a]	0
10	16	A	2	52	2.58	1083.82[a]	0.84	1083.82	1083.82[a]	0
11	16	B	3	75	0.70	1468.08[a]	0.37	1468.08	1468.08[a]	0
12	16	C	3	85	2.14	1318.30[a]	0.52	1318.30	1318.30[a]	0
13	18	A	2	58	351.38	1193.58[a]	2.05	1121.58	1193.58[a]	0
14	18	B	3	85	58.68	1797.78[a]	11.9	1797.78	1797.78[a]	0
15	18	C	3	98	78.52	1674.92[a]	8.25	1674.92	1674.92[a]	0
16	20	A	2	66	419.50	1705.38[a]	21.12	1612.11	1701.71	0.21
17	20	B	4	100	306.40	2100.58[a]	11.86	2004.46	2100.58[a]	0
18	20	C	4	102	433.83	1868.20[a]	16.95	1860.21	1868.20[a]	0
19	26	A	3	87	10000	1605.46	29.94	1523.44	1599.46	0.37
20	26	B	5	134	10000	1985.11	99.51	1895.25	1979.30	0.29
21	26	C	4	122	10000	1799.83	41.59	1801.34	1865.46	−3.64
22	30	A	4	104	10000	1471.63	35.97	1411.33	1427.91	2.97
23	30	B	5	157	10000	1839.98	47.32	1780.24	1807.40	1.77
24	30	C	5	147	10000	1639.34	58.56	1532.84	1609.23	0.25

[a]Optimal solution.

Table 7
Large-size problems.

Problem	No. of pickup & delivery nodes	Data set	No. of vehicles	No. of products	SA			GA-MVNS			ABC		
					Run time (s)	Average solution	Best solution	Run time (s)	Average solution	Best Solution	Run time (s)	Average solution	Best solution
1	40	A	5	145	193.6	1984	2278.4	158.9	2437.2	2611.3	197.3	2091.5	2177.6
2	40	B	7	216	210.7	3130.6	3632.5	160.1	3784.6	3948.4	239.8	3111.1	3404
3	40	C	7	212	231.9	2192.6	2343.9	218.4	2951.2	3218.6	200.2	2412.6	2600.1
4	40	D	5	160	186.4	2631.7	2824.8	193.4	3264.8	3370.2	205.4	2567.4	2857.7
5	40	E	8	222	228.0	2979.8	3115.1	180.9	3458.3	3683.3	241.2	2604.5	2918.6
6	60	A	8	241	310.7	2589.7	3148	211.4	3682.2	3980	301.8	2546.8	2949.4
7	60	B	10	317	384.5	3050.3	3286	259.8	4633.1	4800.4	361.5	3894.8	3604.7
8	60	C	10	312	328.2	3124.2	3564.5	255.3	3838 .7	4261.1	347	3459.5	3066.6
9	60	D	8	240	344.6	3002.8	3254.5	234.9	3762.7	4124.5	313.5	3568	3869.9
10	60	E	10	319	348.3	3195.3	3399.2	261.4	4301.9	4566.9	325.5	3788.8	3565.4
11	100	A	14	442	436.3	4453.7	4512.7	386.4	5112.8	5467.4	412.5	4098.7	4325.9
12	100	B	15	493	442.5	4125.8	4328.6	360.5	5479.3	5845.8	398.6	4323.8	4536.3
13	100	C	16	507	452.1	3654.2	3957.1	384.6	4434.2	4772.7	422.3	4032.5	4125.9
14	100	D	15	463	438.3	3812.3	4002.8	378.2	5167.4	5337.7	416.2	4152.3	4300.8
15	100	E	16	506	469.5	4398.6	4582.6	382.7	5276.4	5582.5	439.7	4479.4	4700.3
16	200	A	29	931	2485.2	7745.3	7952.6	1716.2	8691.6	8972.1	2148.2	7852.1	8125.2
17	200	B	31	994	2412.3	7984.7	8254.1	1696.3	8959.2	9487	2127.7	8024.6	8425.6
18	200	C	31	997	2397.4	7006.8	7485.6	1634.4	8145.6	8426.4	2101.8	7006.9	7452.6
19	200	D	29	941	2403.4	7485.6	7856.2	1643.4	8693.4	9046.2	2115.2	7953.8	8142.3
20	200	E	31	1009	2399.8	7623.1	7942.3	1689.6	8523.8	9092.3	2100.3	7585.5	8253.4

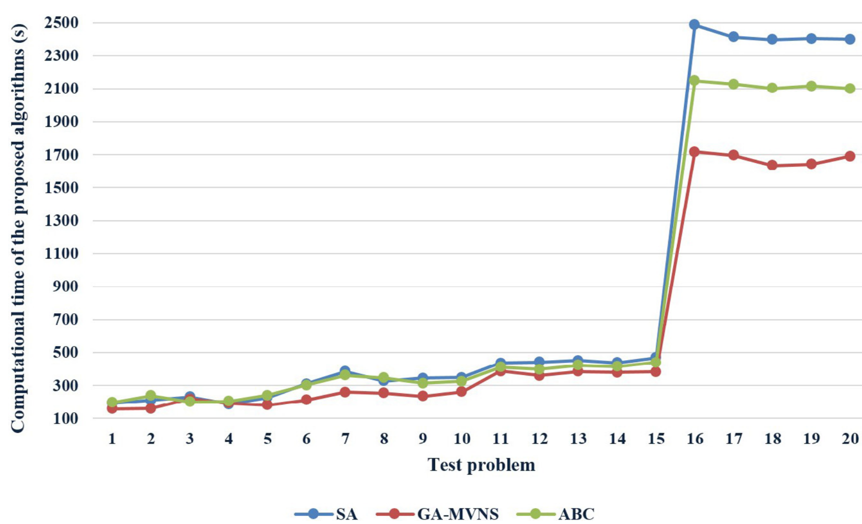


Fig. 20. Comparison between the computational time of the compared algorithms for different test problems.

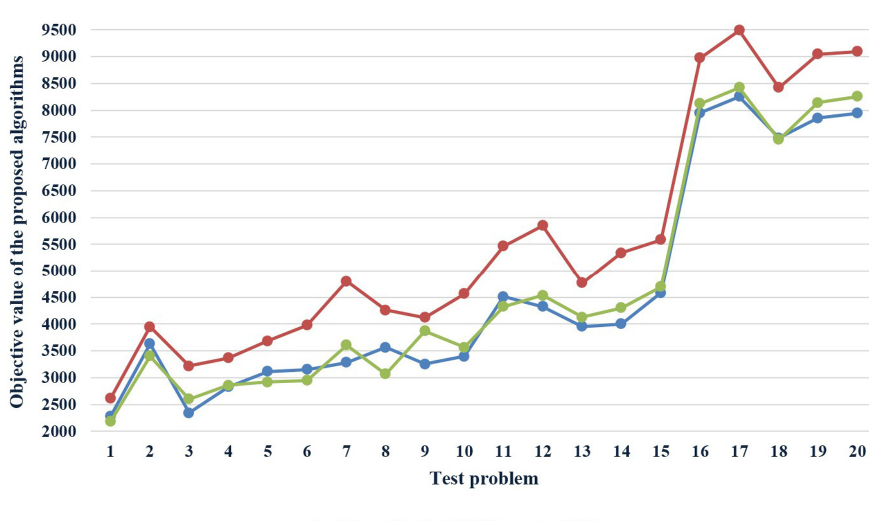


Fig. 21. Comparison between the best solution of the compared algorithms for different test problems.

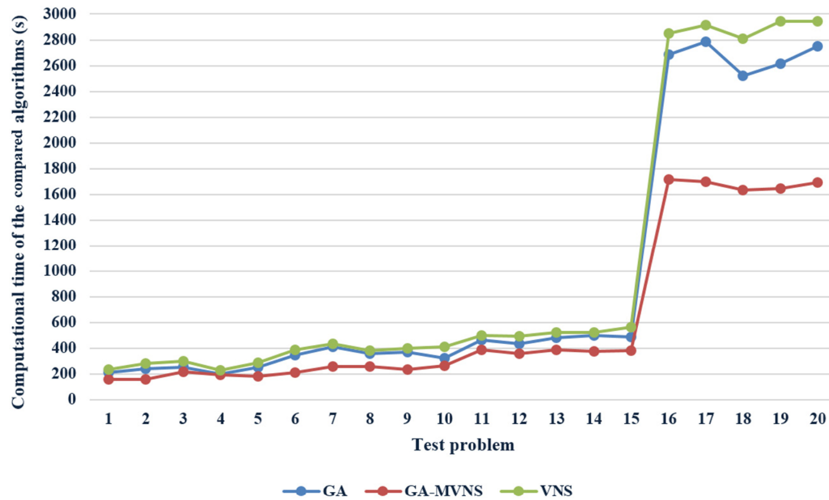


Fig. 22. Comparison between the computational time of the proposed hybrid algorithm with the classic GA and VNS for different test problems.

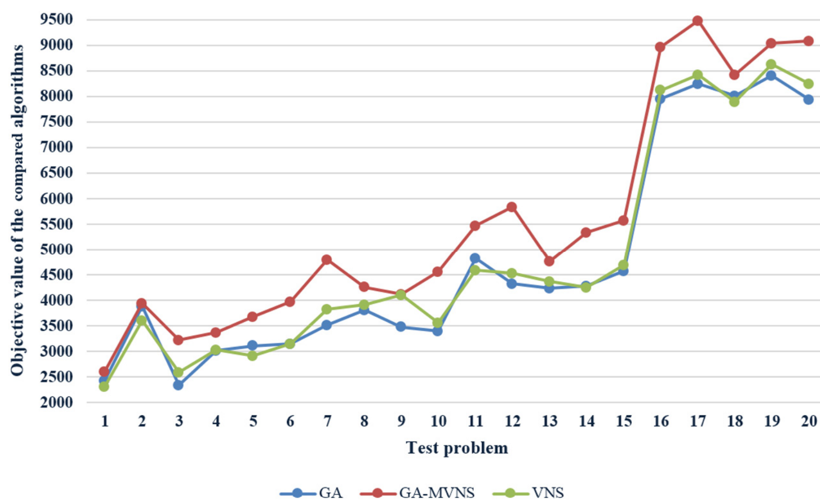


Fig. 23. Comparison between the best solution of the proposed hybrid algorithm with the classic GA and VNS for different test problems.

others in all instances. Figs. 22 and 23 show the same comparison between the proposed hybrid algorithm and the classic GA and VNS.

It can be concluded from computational experiment that the proposed hybrid algorithm, performs better than other compared algorithms (GA, SA, ABC and VNS). Theoretical advances that the proposed hybrid GA-MVNS algorithm can bring to the problem are listed as follows:

1. Finding optimal solutions in the acceptable computational time for small instances.
2. Achieving higher quality solutions than other algorithms in real-world instances, which would bring more profit to the cross docking system.
3. Less computational time than other algorithms for all instances, which results making decisions as soon as possible.
4. In addition, some benefits associated with the proposed hybrid GA-MVNS algorithm are as follows:
5. Starting from better initial solutions using a heuristic algorithm.
6. Excellent convergence speed and global search using the main structure and advantages of the GA.
7. Efficient neighborhood search using four different operators in the shake procedure.
8. Escaping from local optimum using two different operator in local search procedure.
9. Less computational time using modified version of the VNS.

6.3. *Sensitivity analysis of parameters in the proposed mathematical model*

To consider the validity of the proposed model, some sensitivity analyses of different parameters are carried out and the results are reported in following sub sections.

6.3.1. *Sensitivity analysis on the purchasing cost*

In this section, the effect of the purchasing price is investigated on the number of selected nodes as well as the objective function value. As depicted on Fig. 24, it is concluded that by increasing of the purchasing price, the model tends to select less nodes to serve them while by decreasing the mentioned parameter, more demands are satisfied. The same result can be concluded as effect of the purchasing price on the total profit as depicted in Fig. 25.

6.3.2. *Sensitivity analysis on the budget*

The effect of the budget on the total profit of the system and the number of selected nodes are presented in Figs. 26 and 27. It can be concluded that the total profit or selected nodes increases

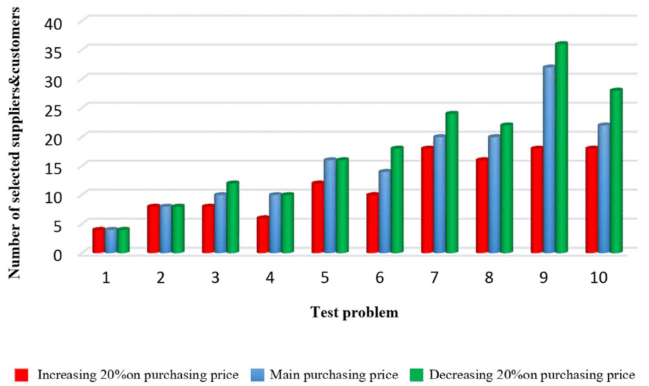


Fig. 24. Effect of changing the purchasing price of products on the number of visited nodes.

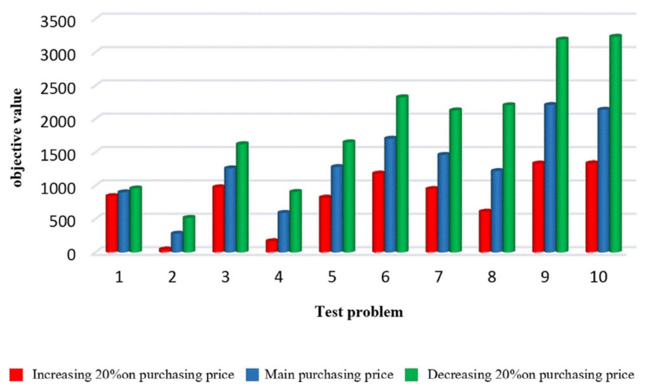


Fig. 25. Effect of changing the purchasing price of products on the total profit.

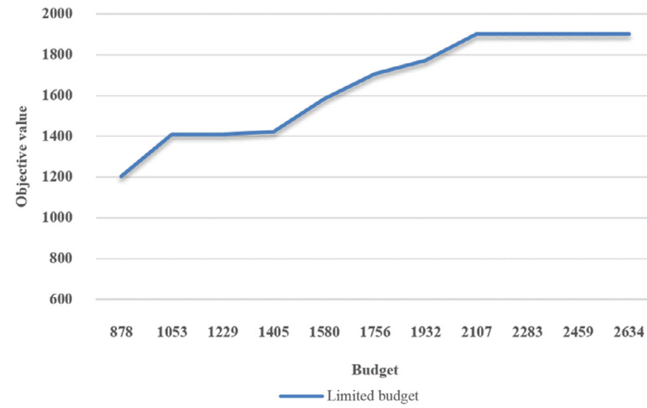


Fig. 26. Effect of increasing the available budget on the total profit in the main problem.

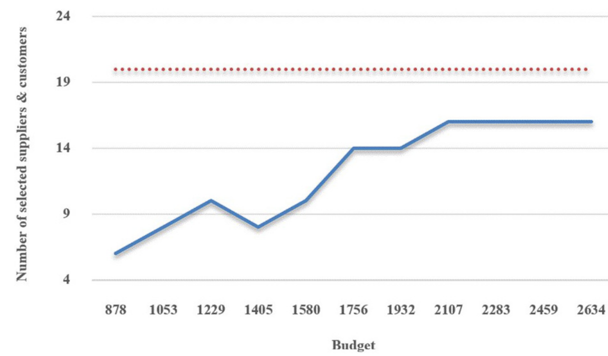


Fig. 27. Effect of increasing the available budget on the number of visited nodes in the main problem.

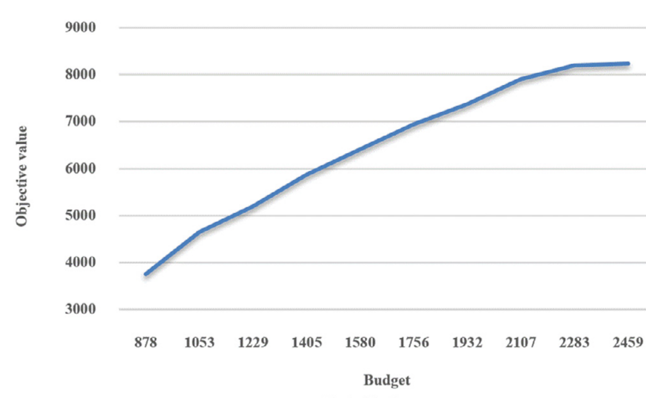


Fig. 28. Effect of budget on total profit in the changed problem with new purchasing cost values.

with an increasing available budget to purchase the products. In the test problem with 20 nodes, it is noted that the total profit and the number of selected nodes are increasing till the budget of 2107 and after that they are stable and have no change by increasing the budget. It means that in this instance with defined parameters, even with no limitation on budget, servicing to all customers is not profitable. In fact, in this instance, the transportation costs of some nodes are higher than the profit of serving them and selling more does not yield more profit, so they never will be in the planning of the cross-dock system. In order to prove this claim, we decide to change the purchasing price from the pre-determined value to a new value. The purchasing price is multiplied by 2. Therefore, Figs. 28 and 29 are the same figures with new value of purchasing price. It is clear that by increasing the budget, the total profit and selected nodes will be increased continuously. As illustrated in Fig. 26, when the available budget is 2459, all of the nodes will be selected to be served and the result of the proposed PHVRPCD converges to the result of the VRPCD. This sensitivity analyses confirm the validity and suitable performance of the presented mathematical model for the profitable vehicle routing problem with cross-docking.

7. Conclusion and future research

Cross-docking as a new distribution strategy has enabled consolidation shipments in order to better manage the physical flow of products in a supply chain. In the literature, there has been no study on a profitable vehicle routing problem with cross-docking. A profitable heterogeneous vehicle routing problem with cross-docking (SHVRPCD) has been introduced in this study. In this problem, it is not necessary to visit all suppliers and customers, so they are selected only if it is profitable to serve them based on the purchasing and selling price with the budget limit. A mixed-integer linear programming (MILP) model has been proposed to formulate the given problem, and different test problems have been solved optimally and analyzed. Some sensitivity analyses with different instances have been performed to prove the validity of the proposed model and assess the effect of some parameters of the model. Also for medium and large-sized instances, three metaheuristic algorithms have been proposed and compared with the commercial software in small-sized instances. Additionally, to show the performance of the proposed algorithms for medium and

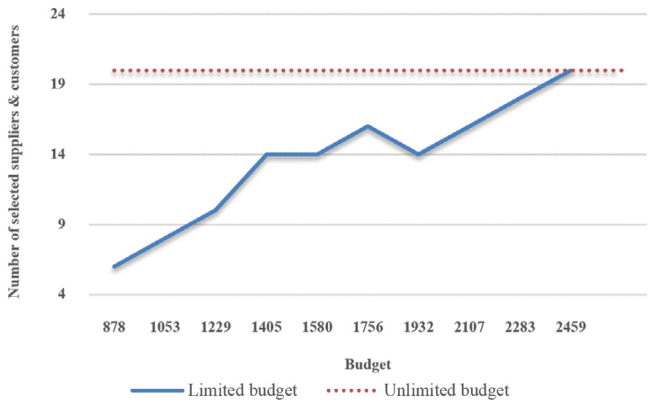


Fig. 29. Effect of budget on visited nodes in the changed problem with new purchasing cost values.

large-sized instances, they have been compared in terms of the objective function value and computational time. It could be concluded from the computational results that the proposed hybrid algorithm has been more efficient than other proposed algorithms to find solutions in less computational time. Using other improved methods and algorithms can be a good direction for future studies. Also as mentioned before, developing a new clustering method for the PHVRPCD to solve larger instances can be interesting for researchers. Furthermore, green vehicle routing problem (Gr-VRP) and reverse logistics can be as other future studies.

References

[1] U.M. Apte, S. Viswanathan, Effective cross docking for improving distribution efficiencies, Int. J. Logist. 3 (2000) 291–302.
[2] Y. Kuo, Optimizing truck sequencing and truck dock assignment in a cross docking system, Expert Syst. Appl. 40 (2013) 5532–5541.
[3] T.W. Liao, P.J. Egbelu, P.C. Chang, Simultaneous dock assignment and sequencing of inbound trucks under a fixed outbound truck schedule in multi-door cross docking operations, Int. J. Prod. Econ. 141 (2013) 212–229.
[4] C.E. Witt, Crossdocking: Concepts demand choice, Mater. Handl. Eng. 53 (1998) 44–49.
[5] G. Forger, UPS starts world's premiere cross-docking operation, Modern Mater. Handl. 36 (1995) 36–38.
[6] E. Kinnear, Is there any magic in cross-docking? Supply Chain Manag.: Int. J. 2 (1997) 49–52.
[7] G.B. Dantzig, J.H. Ramser, The truck dispatching problem, Manag. Sci. 6 (1959) 80–91.
[8] K.L. Cooke, E. Halsey, The shortest route through a network with time-dependent internodal transit times, J. Math. Anal. Appl. 14 (1966) 493–498.
[9] N.H. Wilson, R.W. Weissberg, J. Hauser, Advanced dial-a-ride algorithms research project, (1976) 20–76.
[10] F.A. Tillman, The multiple terminal delivery problem with probabilistic demands, Transp. Sci. 3 (1969) 192–204.
[11] C. Watson-Gandy, P. Dohrn, Depot location with van salesmen—a practical approach, Omega 1 (1973) 321–329.
[12] E.J. Beltrami, L.D. Bodin, Networks and vehicle routing for municipal waste collection, Networks 4 (1974) 65–94.
[13] V. Speidel, EDP-assisted fleet scheduling in tramp and coastal shipping, in: Proceedings of the 2nd International Ship Operation Automation Symposium, Washington, DC, August 30-September 2, 1976.
[14] R.A. Russell, Technical note—An effective heuristic for the M-tour traveling salesman problem with some side conditions, Oper. Res. 25 (1977) 517–524.
[15] W.J. Bell, L.M. Dalberto, M.L. Fisher, A.J. Greenfield, R. Jaikumar, P. Kedia, R.G. Mack, P.J. Prutzman, Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer, Interfaces 13 (1983) 4–23.
[16] B. Golden, A. Assad, R. Dahl, Analysis of a large scale vehicle routing problem with an inventory component, Large Scale Syst. 7 (1984) 181–190.
[17] T. Tsiligirides, Heuristic methods applied to orienteering, J. Oper. Res. Soc. 35 (1984) 797–809.
[18] E.L. Lawler, J.K. Lenstra, A.H. Rinnooy Kan, D.B. Shmoys, The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, Johh Wiley & Sons, New York, NY, 1985.
[19] B.N. Nag, Vehicle Routing in the Presence of Site/vehicle Dependency Constraints, University of Maryland, College Park, 1986, pp. 120–122.
[20] M. Dror, P. Trudeau, Savings by split delivery routing, Transp. Sci. 23 (1989) 141–145.
[21] R. Cheng, M. Gen, T. Tozawa, Vehicle routing problem with fuzzy due-time using genetic algorithms, J. Japan Soc. Fuzzy Theory Syst. 7 (1995) 1050–1061.
[22] D. Sariklis, S. Powell, A heuristic method for the open vehicle routing problem, J. Oper. Res. Soc. 51 (2000) 564–573.
[23] M. Iori, J.S. Gonzalez, D. Vigo, An Exact Approach for the Symmetric Capacitated Vehicle Routing Problem with Two-dimensional Loading Constraints, Università di Bologna, DEIS, Operations Research, 2003.
[24] J. Privé, J. Renaud, F. Boctor, G. Laporte, Solving a vehicle-routing problem arising in soft-drink distribution, J. Oper. Res. Soc. 57 (2006) 1045–1052.
[25] R. Fleichmann, K. Inderfurth, L. Wassenhove, Reverse Logistics: Quantitative Models for Closed-Loop Supply Chains, SpringerVerlag, New York, 2004, pp. 120–153.
[26] I. Kara, B.Y. Kara, M.K. Yetis, Energy minimizing vehicle routing problem, in: International Conference on Combinatorial Optimization and Applications, Springer, 2007, pp. 62–71.
[27] A. Sbihi, R.W. Eglese, Combinatorial optimization and green logistics, 4OR 5 (2007) 99–116.
[28] T.G. Crainic, N. Ricciardi, G. Storchi, Models for evaluating and planning city logistics systems, Transp. Sci. 43 (2009) 432–454.
[29] Y.H. Lee, J.W. Jung, K.M. Lee, Vehicle routing scheduling for cross-docking in the supply chain, Comput. Ind. Eng. 51 (2006) 247–256.
[30] M. Wen, J. Larsen, J. Clausen, J.-F. Cordeau, G. Laporte, Vehicle routing with cross-docking, J. Oper. Res. Soc. 60 (2009) 1708–1718.
[31] C.-J. Liao, Y. Lin, S.C. Shih, Vehicle routing with cross-docking in the supply chain, Expert Syst. Appl. 37 (2010) 6868–6873.
[32] R. Musa, J.-P. Arnaout, H. Jung, Ant colony optimization algorithm to solve for the transportation problem of cross-docking network, Comput. Ind. Eng. 59 (2010) 85–92.
[33] B. Vahdani, R. Tavakkoli-Moghaddam, M. Zandieh, J. Razmi, Vehicle routing scheduling using an enhanced hybrid optimization approach, J. Intell. Manuf. 23 (2012) 759–774.
[34] C.D. Tarantilis, Adaptive multi-restart Tabu Search algorithm for the vehicle routing problem with cross-docking, Optim. Lett. 7 (2013) 1583–1596.
[35] F.A. Santos, G.R. Mateus, A.S. Da Cunha, The pickup and delivery problem with cross-docking, Comput. Oper. Res. 40 (2013) 1085–1093.
[36] S.S. Moghadam, S.F. Ghomi, B. Karimi, Vehicle routing scheduling problem with cross docking and split deliveries, Comput. Chem. Eng. 69 (2014) 98–107.
[37] M. Fakhrzada, A.S. Esfahanib, Modeling the time windows vehicle routing problem in cross-docking strategy using two meta-heuristic algorithms, Int. J. Eng.-Trans. A: Basics 27 (2013) 1113.
[38] R. Dondo, J. Cerdá, A sweep-heuristic based formulation for the vehicle routing problem with cross-docking, Comput. Chem. Eng. 48 (2013) 293–311.
[39] S.M. Mousavi, R. Tavakkoli-Moghaddam, F. Jolai, A possibilistic programming approach for the location problem of multiple cross-docks and vehicle routing scheduling under uncertainty, Eng. Optim. 45 (2013) 1223–1249.
[40] R. Dondo, J. Cerdá, A monolithic approach to vehicle routing and operations scheduling of a cross-dock system with multiple dock doors, Comput. Chem. Eng. 63 (2014) 184–205.
[41] D. Agustina, C. Lee, R. Piplani, Vehicle scheduling and routing at a cross docking center for food supply chains, Int. J. Prod. Econ. 152 (2014) 29–41.
[42] R. Dondo, J. Cerdá, The heterogeneous vehicle routing and truck scheduling problem in a multi-door cross-dock system, Comput. Chem. Eng. 76 (2015) 42–62.
[43] F. Ahmadizar, M. Zeynivand, J. Arkat, Two-level vehicle routing with cross-docking in a three-echelon supply chain: A genetic algorithm approach, Appl. Math. Model. 39 (2015) 7065–7081.
[44] M. Mokhtarinejad, A. Ahmadi, B. Karimi, S.H.A. Rahmati, A novel learning based approach for a new integrated location-routing and scheduling problem within cross-docking considering direct shipment, Appl. Soft Comput. 34 (2015) 274–285.
[45] V.F. Yu, P. Jewpanya, A.A.N.P. Redi, Open vehicle routing problem with cross-docking, Comput. Ind. Eng. 94 (2016) 6–17.
[46] A.H. Goodarzi, S.H. Zegordi, A location-routing problem for cross-docking networks: A biogeography-based optimization algorithm, Comput. Ind. Eng. 102 (2016) 132–146.
[47] P. Grangier, M. Gendreau, F. Lehuédé, L.-M. Rousseau, A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking, Comput. Oper. Res. 84 (2017) 116–126.
[48] S.M. Mousavi, B. Vahdani, A robust approach to multiple vehicle location-routing problems with time windows for optimization of cross-docking under uncertainty, J. Intell. Fuzzy Syst. 32 (2017) 49–62.
[49] A.I. Nikolopoulou, P.P. Repoussis, C.D. Tarantilis, E.E. Zachariadis, Adaptive memory programming for the many-to-many vehicle routing problem with cross-docking, Oper. Res. (2016) 1–38.

[50] P.-Y. Yin, Y.-L. Chuang, Adaptive memory artificial bee colony algorithm for green vehicle routing with cross-docking, Appl. Math. Model. 40 (2016) 9302–9315.
[51] A. Baniamerian, M. Bashiri, F. Zabihi, Two phase genetic algorithm for vehicle routing and scheduling problem with cross-docking and time windows considering customer satisfaction, J. Ind. Eng. Int. (2017) 1–16.
[52] W. Yu, P.J. Egbelu, Scheduling of inbound and outbound trucks in cross docking systems with temporary storage, European J. Oper. Res. 184 (2008) 377–396.
[53] A.S. Wu, H. Yu, S. Jin, K.-C. Lin, G. Schiavone, An incremental genetic algorithm approach to multiprocessor scheduling, IEEE Trans. Parallel Distrib. Syst. 15 (2004) 824–834.
[54] G. Mosheiov, Vehicle routing with pick-up and delivery: tour-partitioning heuristics, Comput. Ind. Eng. 34 (1998) 669–684.
[55] B. Çatay, A new saving-based ant algorithm for the vehicle routing problem with simultaneous pickup and delivery, Expert Syst. Appl. 37 (2010) 6809–6817.
[56] G. Perboli, R. Tadei, D. Vigo, The two-echelon capacitated vehicle routing problem: Models and math-based heuristics, Transp. Sci. 45 (2011) 364–380.
[57] N. Boysen, M. Fliedner, Cross dock scheduling: Classification, literature review and research agenda, Omega 38 (2010) 413–422.
[58] J. Van Belle, P. Valckenaers, D. Cattrysse, Cross-docking: State of the art, Omega 40 (2012) 827–846.
[59] P. Buijs, I.F. Vis, H.J. Carlo, Synchronization in cross-docking networks: A research classification and framework, European J. Oper. Res. 239 (2014) 593–608.
[60] T.W. Liao, P.J. Egbelu, P.C. Chang, Two hybrid differential evolution algorithms for optimal inbound and outbound truck sequencing in cross docking operations, Appl. Soft Comput. 12 (2012) 3683–3697.
[61] T. Crainic, M. Gendreau, Y. Gajpal, Multi-zone multi-trip vehicle routing problem with time windows, in: CIRRELT, 2012.
[62] C.A.C. Coello, Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, Comput. Methods Appl. Mech. Engrg. 191 (2002) 1245–1287.
[63] J. Brotankova, M. Randall, A. Lewis, B. Pressey, A. Wenger, A genetic algorithm solver for pest management control in island systems, in: S.K. Chalup, A.D. Blair, M. Randall (Eds.), Artificial Life and Computational Intelligence: First Australasian Conference, ACALCI 2015, Newcastle, NSW, Australia, February (2015) 5–7 Proceedings, Springer International Publishing, Cham, 2015, pp. 273–285.
[64] M. Bashiri, M. Mirzaei, M. Randall, Modeling fuzzy capacitated p-hub center problem and a genetic algorithm solution, Appl. Math. Model. 37 (2013) 3513–3525.
[65] B. Yan, C. Yan, F. Long, X.-C. Tan, Multi-objective optimization of electronic product goods location assignment in stereoscopic warehouse based on adaptive genetic algorithm, J. Intell. Manuf. 29 (2018) 1273–1285.
[66] V. Roshanaei, B. Naderi, F. Jolai, M. Khalili, A variable neighborhood search for job shop scheduling with set-up times to minimize makespan, Future Gener. Comput. Syst. 25 (2009) 654–661.
[67] P. Hansen, N. Mladenović, D. Urošević, Variable neighborhood search for the maximum clique, Discrete Appl. Math. 145 (2004) 117–125.
[68] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simmulated annealing, Science 220 (1983) 671–680.
[69] X. Yao, A new simulated annealing algorithm, Int. J. Comput. Math. 56 (1995) 161–168.
[70] T. Yang, A.A. Asanjan, M. Faridzad, N. Hayatbini, X. Gao, S. Sorooshian, An enhanced artificial neural network with a shuffled complex evolutionary global optimization with principal component analysis, Inform. Sci. 418 (2017) 302–316.
[71] S.L. Goh, G. Kendall, N.R. Sabar, Simulated annealing with improved reheating and learning for the post enrolment course timetabling problem, J. Oper. Res. Soc. (2018) 1–16.
[72] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
[73] D. Karaboga, Artificial bee colony algorithm, Scholarpedia 5 (3) (2010) 6915.
[74] A. Singh, An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem, Appl. Soft Comput. 9 (2009) 625–631.
[75] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Global Opt. 39 (2007) 459–471.
[76] D. Karaboga, B. Basturk, On the performance of artificial bee colony (ABC) algorithm, Appl. Soft Comput. 8 (2008) 687–697.
[77] D. Karaboga, C. Ozturk, Neural networks training by artificial bee colony algorithm on pattern classification, Neural Netw. World 19 (2009) 279.
[78] N. Karaboga, A new design method based on artificial bee colony algorithm for digital IIR filters, J. Franklin Inst. B 346 (2009) 328–348.
[79] F. Kang, J. Li, Q. Xu, Structural inverse analysis by hybrid simplex artificial bee colony algorithms, Comput. Struct. 87 (2009) 861–870.
[80] N. Moin, H. Halim, Artificial bee colony for inventory routing problem with backordering, 2014.
[81] B. Yao, P. Hu, M. Zhang, S. Wang, Artificial bee colony algorithm with scanning strategy for the periodic vehicle routing problem, Simulation 89 (2013) 762–770.
[82] S. Zhang, C. Lee, K. Choy, W. Ho, W. Ip, Design and development of a hybrid artificial bee colony algorithm for the environmental vehicle routing problem, Transp. Res. D 31 (2014) 85–99.
[83] A. Gomez, S. Salhi, Solving capacitated vehicle routing problem by artificial bee colony algorithm, in: Computational Intelligence in Production and Logistics Systems (CIPLS), 2014 IEEE Symposium on, IEEE, 2014, pp. 48–52.
[84] T. Chang, D. Kong, N. Hao, K. Xu, G. Yang, Solving the dynamic weapon target assignment problem by an improved artificial bee colony algorithm with heuristic factor initialization, Appl. Soft Comput. 70 (2018) 845–863.
[85] Bean, Genetic Algorithms and Random Keys for Sequencing and Optimization, ORSA J. Comput. 6 (1994) 154–160.