Information Sciences 565 (2021) 390–421

Contents lists available at ScienceDirect

Information Sciences

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

An Adaptive Polyploid Memetic Algorithm for scheduling trucks at a cross-docking terminal

Maxim A. Dulebenets

Department of Civil & Environmental Engineering, Florida A&M University-Florida State University (FAMU-FSU) College of Engineering, 2525 Pottsdamer Street, Building A, Suite A124, Tallahassee, FL 32310-6046, USA

ARTICLE INFO

Article history:
Received 29 March 2020
Received in revised form 2 December 2020
Accepted 18 February 2021
Available online 2 March 2021

Keywords:
Cross-docking terminals
Truck scheduling
Evolutionary algorithms
Polyploidy
Hybridization
Service cost savings

ABSTRACT

Many supply chain stakeholders rely on the cross-docking concept, according to which products delivered in specific transportation management units to the cross-docking terminal (CDT) undergo decomposition, sorting based on the end customer preferences, consolidation, and then transported to the final destinations. Scheduling of the inbound and outbound trucks for service at the CDT doors is considered as one of the convoluted decision problems faced by the CDT operators. This study proposes a new Adaptive Polyploid Memetic Algorithm (APMA) for the problem of scheduling CDT trucks that can assist with proper CDT operations planning. APMA directly relies on the polyploidy concept, where copies of the parent chromosomes (i.e., solutions) are stored before performing the crossover operations and producing the offspring chromosomes. The number of chromosome copies is controlled through the adaptive polyploid mechanism based on the objective function improvements achieved and computational time changes. Moreover, a number of problem-specific hybridization techniques are used within the algorithm to facilitate the search process. Computational experiments show that the application of adaptive polyploidy alone may not be sufficient for the considered decision problem. Hybridization techniques that directly consider problem-specific properties are required in order to improve solution quality at convergence. Furthermore, the APMA algorithm developed in this article substantially outperforms some of the well-known state of the art metaheuristics with regards to solution quality and returns truck schedules that have lower total truck service cost.

1. Introduction

1.1. Cross-docking in supply chains

The cross-docking concept and cross-docking terminals (CDTs) have been extensively used by many supply chain stakeholders in order to enhance the effectiveness of various supply chain processes [1,2]. The cross-docking concept can be described as follows. A set of inbound trucks deliver different types of products from various suppliers and/or manufactures to the CDT (see Fig. 1). These products are transported in specific transportation management units (e.g., boxes, barrels, pallets, mini-containers, and others – depending on the product type transported). Certain products may be temperature-sensitive (e.g., food products, pharmaceuticals, chemical products) and need to be transported by trucks with refrigerated

E-mail address: mdulebenets@eng.famu.fsu.edu

https://doi.org/10.1016/j.ins.2021.02.039
0020-0255/

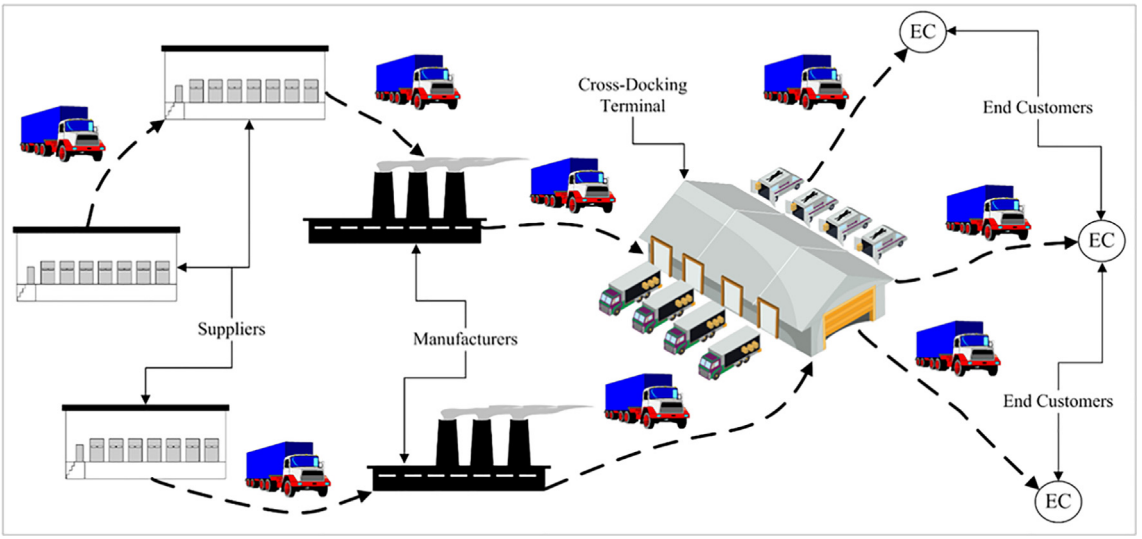


Fig. 1. A supply chain network with a cross-docking terminal.

containers. The arriving trucks are docked at the assigned CDT doors. Next, internal handling equipment is used to unload the inbound trucks and transfer the delivered products to the designated storage areas. Upon delivery to the designated storage areas, the transportation management units are decomposed, and the products are sorted considering the specific preferences of end customers. Finally, these products are consolidated in other transportation management units and transferred by the internal handling equipment to the outbound doors, where the products are loaded into the outbound trucks for delivery to the end customers (see Fig. 1).

One of the key advantages of cross-docking consists in the fact that it substantially reduces the inventory level. According to Ladier and Alpan [1], products delivered by the inbound trucks to the CDT generally do not spend more than 24 hours before these products will be loaded into the designated outbound trucks. There are other advantages from using the cross-docking concept throughout the management of supply chain processes, such as [3]: (i) lower inventory levels at CDTs are associated with the storage space occupancy savings; (ii) cross-docking typically decreases the number of handling operations for the delivered transportation management units as compared to other freight terminals and distribution facilities, which further reduces a potential risk of damaging the delivered products; (iii) cross-docking has been proven to shorten delivery times and make supply chains more agile, which leads to improvements in end customer service; and (iv) carbon footprint of supply chains is reduced due to power savings throughout the transport of products both inside and outside CDTs.

Along with cross-docking advantages, there are some challenges that have to be addressed by the supply chain stakeholders in order to effectively implement the cross-docking concept. One of the most critical challenges is to ensure proper planning and coordination of CDT operations [2,3]. CDT operators face a wide range of decision problems throughout the planning of operations at their facilities. These decision problems include selection of the appropriate CDT shape (e.g., I-shape, T-shape, L-shape), determination of the appropriate number of CDT doors, deployment of the appropriate equipment for internal transportation (e.g., conveyor belts, forklift operators), allocation of the areas for temporary product storage, allocation of loading and unloading tasks for the equipment units available for internal transportation, scheduling of the inbound trucks and outbound trucks for service at the CDT doors available, among others [1]. This study will primarily concentrate on the truck scheduling problem, which is considered as one of the most convoluted decision problems that are faced by CDT operators. In the truck scheduling problem, a given CDT operator aims to allocate the arriving inbound trucks and outbound trucks among the CDT doors available, set the appropriate order of service for the trucks at each CDT door, as well as determine the start and finish service times for each truck. The aforementioned decisions have to take into consideration important constraints from the operational perspective (e.g., number of the equipment units available for internal transportation, existing capacity of the equipment units available for internal transportation, and capacity of the existing areas for temporary product storage).

1.2. Solution approaches for CDT truck scheduling and polyploidy concept

The CDT truck scheduling problem is known to have high computational complexity. The existing CDT truck scheduling studies generally use metaheuristic and heuristic algorithms to solve the CDT truck scheduling problem in a reasonable amount of computational time for the realistic-size problem instances [1,2,4,5]. Evolutionary Algorithms (referred to as "EAs" in this study) have been extensively used in the state of the art to tackle some of the challenging decision problems

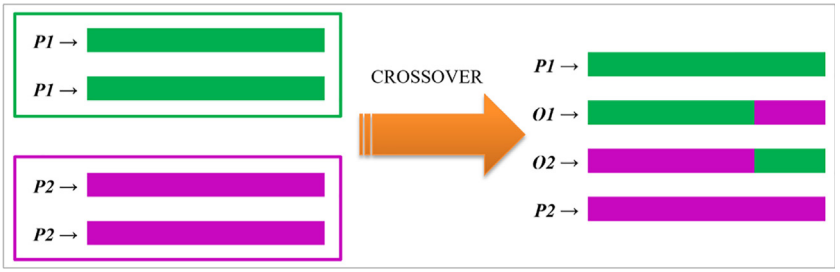


Fig. 2. An example of a crossover operation for diploid organisms.

that have high computational complexity [6–12]. EAs typically start the search process by initializing the population of chromosomes or individuals that represent candidate solutions to the considered decision problem [13]. After evaluation of the fitness for the initial population chromosomes, EAs execute the main loop, where the available chromosomes are being continuously modified by applying different crossover and mutation operators until a pre-defined stopping criterion is satisfied.

Throughout the EA search process, two types of selection are applied: the parent selection and the survivor selection. The purpose of parent selection is to select the parent chromosomes that will undergo crossover and mutation to generate the offspring. On the other hand, the survivor selection aims to select the chromosomes that will be moved to the following generation. Some EAs do not allow competition between the parent and offspring chromosomes, and the next generation chromosomes are identified from the pool of the offspring chromosomes only (e.g., the canonical Genetic Algorithm that was developed by Holland in the 1970s; the canonical Genetic Programming that was developed by Koza in the 1990s) [13]. However, there are some types of EAs that allow competition between the parent and offspring chromosomes, and the next generation chromosomes are identified from the combined pool of the parent and offspring chromosomes (e.g., the canonical Evolutionary Programming that was developed by Fogel in the 1960s; the canonical Evolution Strategies that were developed by Rechenberg in the 1960s-1970s) [13].

The purpose of applying crossover and mutation operators is to discover the chromosomes with higher fitness values. The canonical EAs mostly rely on the haploid structure of chromosomes throughout the crossover operations, where two parent chromosomes will produce two offspring chromosomes that inherit features of both parental chromosomes. Only a limited number of studies considered the polyploidy concept, where copies of the parent chromosomes are being stored before performing the crossover operation and producing the offspring chromosomes [5,14]. Fig. 2 shows an illustrative example of a crossover operation for diploid organisms, where one copy of each parent chromosome (“P1” and “P2”) along with two offspring chromosomes (“O1” and “O2”) are present after the crossover operation. The crossover operations generally result in major genetic changes and may produce the offspring chromosomes that have lower fitness compared to the parent chromosomes. Therefore, storage of the parental genome by means of polyploidy is expected to increase the population diversity, prevent loss of high-fitness chromosomes, and return superior solutions at convergence.

1.3. Focus of this study

Given potential advantages of polyploidy, this study proposes a new Adaptive Polyploid Memetic Algorithm (APMA) for the CDT truck scheduling problem that can assist CDT operators with proper operations planning from the truck scheduling perspective. Unlike the polyploid EA-based algorithms that were previously developed for different decision problems, the proposed APMA algorithm adaptively deploys the polyploidy concept based on the objective function improvements achieved and computational time changes. Although increasing the number of chromosome copies in polyploid individuals enhances the explorative capabilities of the algorithm, computational time can be significantly increased due to polyploidy. Such a challenge would be effectively addressed by introducing the proposed adaptive polyploidy concept. Moreover, a number of problem-specific hybridization techniques are used within the developed algorithm to facilitate the search process. The remaining sections of the manuscript elaborate on the following aspects. The second section contains a detailed review of the relevant studies and highlights the contributions of the present study to the state of the art. The third section formally introduces the CDT truck scheduling problem, while the fourth section proposes a mixed integer linear programming formulation for the problem. Details regarding the design of the APMA algorithm are presented in the fifth section, while computational experiments that were performed are discussed in the sixth section. This study is concluded in the last section. Furthermore, some future research opportunities are outlined in the last section as well.

2. Literature review

This section of the manuscript overviews the relevant literature, including the recent studies on CDT truck scheduling as well as previous studies that deployed polyploid EAs for different decision problems. Furthermore, the critical state of the art shortcomings along with the key contributions of this work will be highlighted.

2.1. Recent studies on CDT truck scheduling

A thorough review of the CDT truck scheduling literature can be found in Ladier and Alpan [1] and Theophilus et al. [2]. The literature review in this study primarily captures recent CDT truck scheduling efforts, which were not reviewed in Ladier and Alpan [1] and Theophilus et al. [2]. Castellucci et al. [15] studied a container loading problem at the distribution facilities with a limited storage space (e.g., CDTs). The proposed model explicitly considered the expected arrival of products and the requested departure time of trucks, aiming to maximize the total output and minimize the truck ready times. The model was solved using a dynamic programming approach. It emphasized that the presented methodology could be used to quantify the effects of delays on the CDT capacity utilization. Fathollahi-Fard et al. [16] applied a set of social engineering optimizers inspired by social engineering phenomena to the CDT truck scheduling problem. The presented mathematical formulation focused on the makespan minimization. Numerical experiments performed for the benchmark test problems confirmed the efficiency of proposed optimizers.

Ardakani et al. [17] addressed the problem of scheduling CDT trucks, considering preemption for the inbound trucks (in other words, some of the inbound trucks could temporarily leave the assigned doors, so the other trucks could be served). The objective minimized the makespan. A set of heuristic algorithms were developed to solve the problem. The heuristics were differentiated based on the criteria that were used for assigning the arriving trucks for service. The number of transferred products was found to be an important criterion to schedule the service of arriving trucks. Wisittipanich et al. [18] developed a mathematical formulation for the problem of truck scheduling in a CDT network. The objective minimized the makespan. LINGO was used to solve the developed mathematical model. It was found that optimizing truck schedules in a CDT network was more effective compared to optimizing truck schedules for each CDT individually. Shahmardan and Sajadieh [19] addressed the problem of scheduling CDT trucks, where the inbound trucks could serve as outbound trucks. The objective minimized the makespan. A hybrid heuristic-simulated annealing was adopted as a resolution approach. A series of numerical experiments clearly showed that partial unloading of the compound trucks could significantly reduce the makespan.

Guemri et al. [20] proposed two probabilistic Tabu Search heuristics for the CDT door assignment problem with an objective to assign the arriving inbound trucks to the inbound doors and the arriving outbound trucks to the outbound doors. The objective minimized the total cost of material handling inside the CDT. The heuristics, evaluated for 99 benchmark problem instances, were able to discover the 53 best-known solutions and required less computational time. Wang and Alidaee [21] also studied the CDT door assignment problem. Due to the problem's complexity, a new solution method was designed and inspired by genetic random-key, multi-start, and very large scale neighborhood search. It was shown that the developed algorithm could tackle very large problem instances that have up to 300 inbound doors and 300 outbound doors.

Some of the recent efforts considered joint truck and workforce scheduling. Corsten et al. [22] presented a mathematical formulation for integrated truck and workforce scheduling at a CDT. Based on the derived truck schedules, the model allocated the available workers to shifts. The objective minimized the total cost due to the assignment of temporary workers. GUROBI was used as a solution approach. Computational experiments indicated that the truck arrival times, capacity of staging areas, duration of truck service time windows, as well as the number of shifts had a significant influence on the total cost. Selma et al. [23] studied the internal operations at automated CDTs that allocate the available loading tasks to robots and ensure timely departures of trucks. A greedy heuristic algorithm was proposed to solve the problem. Numerical experiments underlined the effectiveness of the heuristic compared to CPLEX. Tadumadze et al. [24] focused on integrated truck and workforce scheduling with the major objective to facilitate the service of trucks at CDTs and distribution centers. The objective minimized the total truck service time. A set of heuristic algorithms were proposed in the study to address the problem of interest. Computational experiments demonstrated that integrated truck and workforce scheduling could significantly reduce the total truck service time and ensure punctuality of truck departures.

Several studies addressed the vehicle routing problem, where the transportation network included CDTs as well. For example, Ahkamiraad and Wang [25] studied the vehicle routing problem within a CDT distribution network, taking into account pickup and delivery operations as well as time window constraints. The objective focused on the minimization of total cost of transportation along with the total fixed cost of vehicles. A hybrid metaheuristic was deployed as a solution method, which was based on the EA and Particle Swarm Optimization (PSO) features. Numerical experiments demonstrated the superiority of the algorithm developed over CPLEX for the medium as well as large problem instances. Abad et al. [26] introduced a multi-objective formulation for the vehicle routing problem, where the products were processed and consolidated in a CDT. The considered objectives aimed to minimize the total cost, minimize the total fuel consumption, and maximize the total satisfaction level of suppliers and customers. The problem was solved using the Multi-Objective Imperialist Competitive Algorithm (MOICA) and the Multi-Objective Grey Wolf Optimizer (MOGWO). The conducted experiments revealed that MOGWO generally had more solutions in the Pareto Front. Rahbari et al. [27] formulated a bi-objective model for the problem of vehicle routing with a CDT, considering perishability of the products transported, travel time uncertainty, and product freshness-life uncertainty. The first objective function focused on the minimization of total transportation cost. In the meantime, the second objective maximized the total weighted product freshness. It was found that the travel time uncertainty could substantially affect the vehicle routing decisions.

2.2. Previous studies on polyploid EAs

Bagley [28] was one of the first studies that applied the concept of polyploidy within EAs. However, no comprehensive comparison of haploid and diploid EAs was conducted. A number of subsequent studies introduced various dominance schemes within polyploid EAs [29–31], where the alleles of genes were categorized into dominant alleles and recessive alleles. Kamrani et al. [32] applied a set of EAs for intelligent knowledge acquisition and data mining. It was found that the diploidy concept was more promising than the haploidy concept. However, consideration of dominance and gender might worsen the fitness of chromosomes throughout the algorithmic evolution. Cavill et al. [33] highlighted that many organisms in nature are polyploid, and multiple copies of various chromosomes might improve the EA performance. Numerical experiments conducted for the symbolic regression problem demonstrated that an increasing amount of different chromosomes, as well as the amount of copies of these chromosomes, could substantially improve the fitness values at convergence.

Elshamy et al. [34] assessed the effects of polyploidy on the performance of multi-objective EAs. A series of computational experiments were undertaken using the benchmark test problems. It was found that 2- and 3-ploid EAs had the least distance to the true Pareto Front. On the other hand, 7- and 10-ploid EAs generally performed worse than 2- and 3-ploid EAs with regards to solution quality but were superior to NSGA-II. Furthermore, the diversity of polyploid EAs typically increased with the number of objectives. Pop et al. [35] presented a hybrid diploid EA for the generalized traveling salesman problem, where the vertices of the graph were partitioned in clusters. The key objective was to find the minimum cost tour and visit exactly one vertex for each cluster. The problem was decomposed into two levels. Numerical experiments that were performed as a part of the study showed that the developed diploid EA outperformed the Memetic Algorithm (MA), classical EA, Random-Key EA (RKEA), and PSO with regards to solution quality. A similar decision problem, called "the family traveling salesman problem," was studied by Pop et al. [36], where only a specific number of nodes had to be visited for each family of nodes. It was found that the diploid representation ensured the population diversity. Furthermore, the developed diploid EA showed a competitive performance against the RKEA, Greedy Randomized Adaptive Search procedure (GRASP), and classical EA.

Diploid and polyploid EAs were also found to be effective solution approaches for non-stationary optimization problems with the dynamic environment, where the optimal solution is subject to changes over time [14,29,37–43]. Goldberg and Smith [29] was one of the pioneering studies that relied on the concepts of polyploidy and dominance under the dynamic environment settings. Uyar and Harmanci [37] showed that the diploid representation of individuals, meiotic division of cells, and dynamic dominance map allowed preserving diversity and improving solution quality for a variation of the 0–1 knapsack problem with a changing weight constraint. Uyar and Harmanci [38] presented an innovative approach for adaptive dominance that can be applied to diploid EAs, where the dominance of a given individual was calculated based on the phenotypic and fitness values of individuals in the population. The proposed methodology was compared to the ones presented by Ryan [30] and Ng and Wong [31]. Computational experiments performed for the dynamic 0–1 knapsack problem indicated that the developed methodology was able to better adjust for the dynamic environment.

Uyar and Harmanci [39] developed an adaptive domination change mechanism that can be applied to diploid EAs, aiming to improve the algorithmic capabilities under the dynamic environment settings. A series of experiments were performed for the dynamic bit matching problem. It was found that the application of diploidy and the adaptive domination method were more beneficial for the algorithmic search process compared to the application of diploidy only and application of the adaptive domination method only. Shabash and Wiese [42] proposed a new dominance mechanism for the real valued diploid EA, where a chromosome with higher fitness was considered as dominant. The developed diploid EA outperformed the haploid EA for the optimization model with a dynamic function. Moreover, it was highlighted that the advantage of the diploidy concept primarily stemmed from the genetic information arrangement – not from increasing the amount of genetic information. Petrovan et al. [43] focused on a comprehensive comparative analysis of a diploid EA against a haploid EA for a number of common benchmark functions (i.e., the Sphere function, the Griewank function, the Ackley function, the Rastrigin function, as well as the Schwefel function). Numerical experiments clearly showed the superiority of the diploid EA for the considered benchmark functions.

2.3. State of the art limitations and contributions of this study

A thorough review of the state of the art revealed a variety of nature-inspired algorithms that were applied to the problem of scheduling trucks at CDTs. Nevertheless, the polyploidy concept is generally ignored by the studies on CDT truck scheduling. However, the polyploidy concept has been proven to be effective for a wide range of different decision problems (e.g., the symbolic regression problem, the family traveling salesman problem, the dynamic bit matching problem). In the meantime, many studies on polyploid solution algorithms primarily use solution quality and population diversity as the major performance indicators [14]. Computational time, which can significantly increase with increasing number of copies of each chromosome in polyploid individuals, is often ignored. Moreover, a significant increase in the number of chromosome copies does not necessarily improve solution quality [34]. Acknowledging the aforementioned shortcomings, this study proposes a new Adaptive Polyploid Memetic Algorithm (APMA) for the CDT truck scheduling problem, which adaptively deploys the polyploidy concept and controls the number of chromosome copies for each individual based on both solution quality

and computational time criteria. Along with the adaptive polyploidy concept, this study offers the following contributions to the state of the art and the state of the practice:

- A novel heuristic algorithm (referred to as a Truck Sequence Refinement [TSR] heuristic) is proposed for initializing solutions within the developed algorithm. The developed heuristic directly accounts for the truck service precedence constraints based on practical considerations in cross-docking. A set of computational experiments are performed to evaluate the TSR heuristic against the alternative heuristics that were used in the previous studies on CDT truck scheduling.
- A set of innovative problem-specific hybridization techniques are used to facilitate the search process. A customized local search heuristic (referred to as an Optimal Truck Sequence Identification [OTSI] heuristic) is developed to improve the quality of population chromosomes after applying the crossover and mutation operations. The OTSI heuristic periodically solves the Outbound Truck Sequencing Problem to global optimality (i.e., after a pre-specified number of generations).
- The developed APMA algorithm is evaluated against well-known metaheuristics that have been extensively used in the CDT truck scheduling literature (such as Ant Colony Optimization, Tabu Search, Variable Neighborhood Search, and Simulated Annealing). Moreover, the developed APMA algorithm is evaluated against the exact optimization approach (CPLEX) that has been extensively used in the state of the art for large-scale mixed integer linear programming models.
- A set of managerial insights are presented after evaluating the computational performance of the developed APMA algorithm for the considered problem instances. These insights could potentially assist CDT operators with proper operations planning and truck scheduling as well as reduce the associated costs throughout the service of inbound and outbound trucks.

3. CDT operations description

A description of the CDT modeled in this study and the key CDT operations are presented under this section of the manuscript. The CDT has a set of doors ($D = \{1, \cdots, n\}$) that are available to serve the arriving trucks (see Fig. 3). A set of the inbound trucks and outbound trucks to be served at the CDT will be further denoted as $T = \{1, \cdots, m\}$. Each inbound truck belonging to a subset of the inbound trucks $T^I = \{1, \cdots, m_1\}, T^I \subseteq T$ delivers the products either for one or several outbound trucks belonging to a subset of the outbound trucks $T^O = \{1, \cdots, m_2\}, T^O \subseteq T$. However, the products that are delivered by inbound trucks are not interchangeable. In this study, it is assumed that $T^I \cap T^O = \oslash$ and $T^I \cup T^O = T$, which means that each

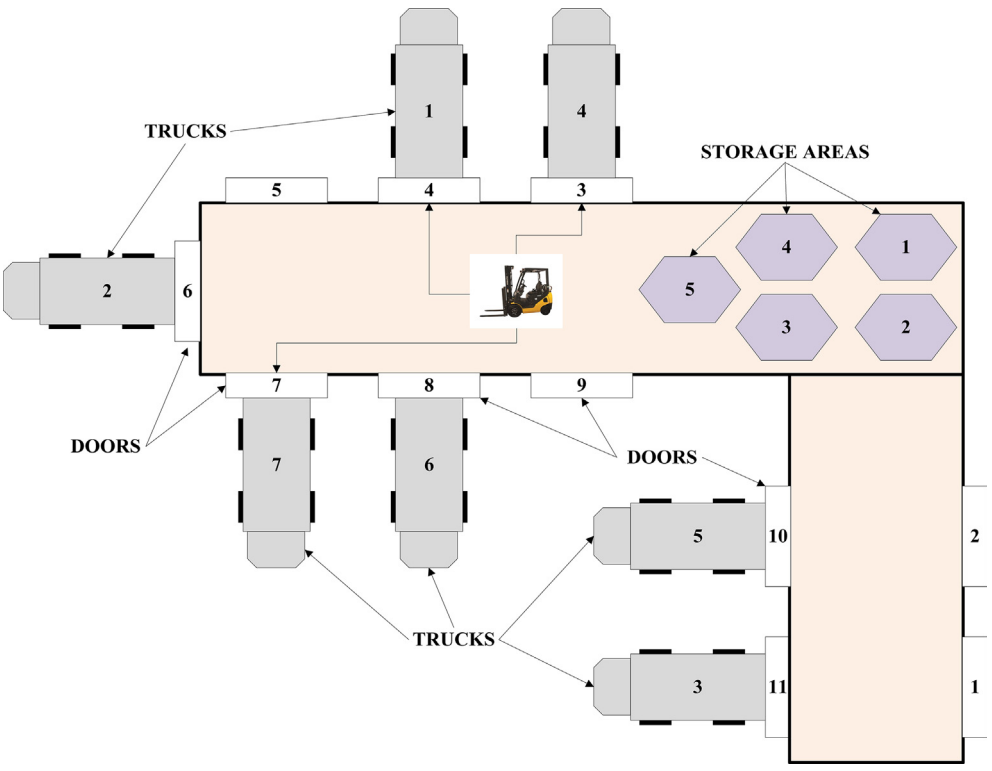


Fig. 3. A schematic illustration of the CDT and truck service.

arriving truck can be considered as either inbound or outbound (but not both). The service of an outbound truck may not be started prior to the service start of the inbound truck delivering the products for that particular outbound truck. Each CDT door operates in a mixed service mode – i.e. the service of inbound trucks and service of outbound trucks can be provided at each CDT door. The arrival times of inbound trucks and outbound trucks ($\tau_t^{AT}, t \in T$ – hours) are assumed to be known to the CDT operator and deterministic in nature, which is in line with the previous studies on CDT truck scheduling [5,44,45]. A given truck may have to wait in a dedicated area of the CDT, when the assigned CDT door is not available immediately. In such situation, the CDT operator will be imposed a truck waiting cost ($\delta_t^{WT}, t \in T$ – USD/hour), since excessive waiting time of trucks may disrupt supply chain processes.

Upon docking at the assigned CDT door, the available handling equipment will be used to unload the products from the inbound trucks and load the products into the outbound trucks. The forklift operators will perform truck loading and unloading operations at the considered CDT as well as the horizontal transfer of the delivered products inside the CDT. This study does not allow truck service preemption. In particular, each truck will have to stay at the CDT door assigned until its service completion. The forklift operators transfer the products unloaded from the inbound trucks either directly to the corresponding outbound trucks or to the designated storage areas in case the delivered products require decomposition, sorting, and/or consolidation based on the end customer preferences. In the truck service example, presented in Fig. 3, the products are transported to the CDT by inbound trucks "1" and "4". The forklift operator transported then delivered products from inbound trucks "1" and "4", which are docked at CDT doors "4" and "3", directly to outbound truck "7", which is docked at CDT door "7", without using the available storage areas. The CDT operator is expected to plan accordingly for the amount of delivered products to ensure that the available storage areas have an adequate capacity for accommodating these products.

Several factors may directly influence the handling time of inbound trucks and outbound trucks ($\tau_{td}^{HT}, t \in T, d \in D$ – hours) at the considered CDT, including the following: (i) quantity of the delivered products; (ii) type of the delivered products; and (iii) location of the CDT door assigned. The handling time of a given inbound truck increases with increasing number of product units to be unloaded from that inbound truck, while the handling time of a given outbound truck increases with increasing number of product units to be loaded into that outbound truck. Furthermore, the products, which are placed in overweight and/or oversized transportation units, may increase the total time that is required by a given forklift operator to transfer these products inside the CDT. In the meantime, location of the CDT door assigned can be considered as an important determinant of the handling time for inbound and outbound trucks. In the truck service example, presented in Fig. 3, inbound trucks "1" and "4" delivering the products for outbound truck "7" are assigned for service at CDT doors "4" and "3", respectively. However, if inbound truck "4" is re-assigned for service at CDT door "1", the handling time of outbound truck "7" will increase. Such an increase can be justified by the fact that CDT door "1" is located farther away from CDT door "7", where outbound truck "7" is assigned for service, compared to CDT door "3", and the forklift operators will require more time for transferring the products between CDT doors "7" and "1".

The handling times of inbound trucks and outbound trucks are presumed to be deterministic in nature, which are in line with the previous studies on CDT truck scheduling [5,44,45]. The CDT operator incurs certain costs throughout the service of inbound trucks and service of outbound trucks. In particular, the handling cost ($\delta_t^{HT}, t \in T$ – USD/hour) is imposed to account for compensation of the CDT employees (e.g., forklift operators) and maintenance of the available equipment, insurance fees, and other operational costs. Moreover, the inventory cost ($\delta_t^{PST}, t \in T$ – USD/hour) is imposed to account for the CDT inventory levels at the available storage areas. Increasing inventory levels is not desirable, as it may disrupt the CDT operations (e.g., CDT congestion).

The CDT operator is also expected to complete the service of inbound trucks and service of outbound trucks in a timely manner, following the scheduled departure time ($\tau_t^{SD}, t \in T$ – hours). The early departure cost ($\delta_t^{ET}, t \in T$ – USD/hour) and the delayed departure cost ($\delta_t^{DT}, t \in T$ – USD/hour) are imposed in case of completing the truck service before and after the scheduled departure time, respectively. Such an assumption forcing the just-in-time completion of truck service is in line with the previous studies on CDT truck scheduling [1,2,45]. The just-in-time completion of truck service is expected to facilitate supply chain processes. The main objective of the CDT operator in this study is to design a truck service schedule that will minimize the total truck service cost, which incorporates the following cost components: (i) the total cost of waiting trucks at the CDT; (ii) the total cost of handling trucks at the CDT; (iii) the total inventory cost for the products to be loaded into the outbound trucks; (iv) the total early departure cost of trucks from the CDT; and (v) the total delayed departure cost of trucks from the CDT.

4. Model formulation

A mixed integer linear programming formulation for the inbound and outbound truck scheduling problem at the CDT (**IOTSP**) is presented in this section of the manuscript. Basic notations, which will be used in the proposed **IOTSP** mathematical model and throughout this manuscript, are described in Table 1.

The objective function (1) of the proposed mathematical formulation for **IOTSP** minimizes the total cost (**Z** – measured in USD) incurred by the CDT operator throughout the service of inbound trucks and service of outbound trucks. The total truck service cost incorporates a number of individual cost components, such as: (i) the total cost of waiting trucks at the CDT; (ii)

Table 1
Basic notations.

Model Component		Description
Type	Notation	
Sets	$T = \{1, \cdots, m\}$	set of the inbound trucks and outbound trucks to be served at the CDT
	$T^I = \{1, \cdots, m_1\}, T^I \subseteq T$	set of the inbound trucks to be served at the CDT
	$T^O = \{1, \cdots, m_2\}, T^O \subseteq T$	set of the outbound trucks to be served at the CDT
	$D = \{1, \cdots, n\}$	set of the CDT doors available to serve the arriving trucks
Decision variables	$\boldsymbol{x}_{td} \in \{0,1\} \forall t \in T, d \in D$	=1 if truck t is assigned for service at door d (=0 otherwise)
	$\boldsymbol{y}^s_{ps} \in \{0,1\} \forall p, s \in T, p \neq s$	=1 if service of truck s is provided immediately after service of truck p at a given CDT door (=0 otherwise)
	$\boldsymbol{y}^f_t \in \{0,1\} \forall t \in T$	=1 if truck t is assigned for service as the first truck at a given CDT door (=0 otherwise)
	$\boldsymbol{y}^l_t \in \{0,1\} \forall t \in T$	=1 if truck t is assigned for service as the last truck at a given CDT door (=0 otherwise)
Auxiliary variables	$\boldsymbol{Z} \in R^+$	total truck service cost incurred by the CDT operator (USD)
	$\boldsymbol{\tau}^{WT}_t \in R^+ \forall t \in T$	waiting time of truck t at the CDT (hours)
	$\boldsymbol{\tau}^{ST}_t \in R^+ \forall t \in T$	start service time of truck t at the CDT (hours)
	$\boldsymbol{\tau}^{FT}_t \in R^+ \forall t \in T$	finish service time of truck t at the CDT (hours)
	$\boldsymbol{\tau}^{PST}_t \in R^+ \forall t \in T$	storage time of the products, which will be loaded into truck t, at the CDT (hours)
	$\boldsymbol{\tau}^{ET}_t \in R^+ \forall t \in T$	early departure of truck t from the CDT (hours)
	$\boldsymbol{\tau}^{DT}_t \in R^+ \forall t \in T$	delayed departure of truck t from the CDT (hours)
Parameters	$\tau^{AT}_t \in R^+ \forall t \in T$	scheduled arrival time of truck t at the CDT (hours)
	$\tau^{DA}_d \in R^+ \forall d \in D$	time when door d becomes available in the considered planning horizon for the first time (hours)
	$\tau^{HT}_{td} \in R^+ \forall t \in T, d \in D$	handling time of truck t at door d of the CDT (hours)
	$\tau^{SD}_t \in R^+ \forall t \in T$	scheduled departure time of truck t from the CDT (hours)
	$r_{ps} \in \{0,1\} \forall p \in T^I, s \in T^O, p \neq s$	=1 if the products delivered by inbound truck p will be further loaded into outbound truck s (=0 otherwise)
	$\delta^{WT}_t \in R^+ \forall t \in T$	unit waiting cost of truck t at the CDT (USD/hour)
	$\delta^{HT}_t \in R^+ \forall t \in T$	unit handling cost of truck t at the CDT (USD/hour)
	$\delta^{PST}_t \in R^+ \forall t \in T$	unit inventory cost of the products, which will be loaded into truck t (USD/hour)
	$\delta^{ET}_t \in R^+ \forall t \in T$	unit cost of early departure for truck t from the CDT (USD/hour)
	$\delta^{DT}_t \in R^+ \forall t \in T$	unit cost of delayed departure for truck t from the CDT (USD/hour)
	$\Gamma \in R^+$	large positive number

the total cost of handling trucks at the CDT; (iii) the total inventory cost for the products to be loaded into the outbound trucks; (iv) the total early departure cost of trucks from the CDT; and (v) the total delayed departure cost of trucks from the CDT.

$$min\ \boldsymbol{Z} = \left[\sum_{t \in T} \left(\boldsymbol{\tau}^{WT}_t \delta^{WT}_t\right) + \sum_{t \in T} \sum_{d \in D} \left(\tau^{HT}_{td} \boldsymbol{x}_{td} \delta^{HT}_t\right) + \sum_{t \in T} \left(\boldsymbol{\tau}^{PST}_t \delta^{PST}_t\right) + \sum_{t \in T} \left(\boldsymbol{\tau}^{ET}_t \delta^{ET}_t\right) + \sum_{t \in T} \left(\boldsymbol{\tau}^{DT}_t \delta^{DT}_t\right)\right] \tag{1}$$

Constraint set (2) guarantees that every truck will be assigned for service at one of the CDT doors available.

$$\sum_{d \in D} \boldsymbol{x}_{td} = 1 \forall t \in T \tag{2}$$

Constraint set (3) assures that every truck will be either assigned for service as the first truck at the considered CDT door or after another truck.

$$\boldsymbol{y}^f_s + \sum_{p \in T: p \neq s} \boldsymbol{y}^s_{ps} = 1 \forall s \in T \tag{3}$$

Constraint set (4) assures that every truck will be either assigned for service as the last truck at the considered CDT door or before another truck.

$$\boldsymbol{y}^l_p + \sum_{s \in T: s \neq p} \boldsymbol{y}^s_{ps} = 1 \forall p \in T \tag{4}$$

Constraint set (5) indicates that only one truck will be assigned for service as the first truck at the considered CDT door.

$$\boldsymbol{y}^f_p + \boldsymbol{y}^f_s \leq 3 - \boldsymbol{x}_{pd} - \boldsymbol{x}_{sd} \forall p, s \in T, p \neq s, d \in D \tag{5}$$

Constraint set (6) indicates that only one truck will be assigned for service as the last truck at the considered CDT door.

$$\boldsymbol{y}^l_p + \boldsymbol{y}^l_s \leq 3 - \boldsymbol{x}_{pd} - \boldsymbol{x}_{sd} \forall p, s \in T, p \neq s, d \in D \tag{6}$$

Constraint sets (7) and (8) state that a given truck can be assigned for service after another truck, if both trucks are to be served at the same door of the CDT.

$$\boldsymbol{y}_{ps}^{s} - 1 \leq \boldsymbol{x}_{pd} - \boldsymbol{x}_{sd} \forall p, s \in T, p \neq s, d \in D \tag{7}$$

$$\boldsymbol{x}_{pd} - \boldsymbol{x}_{sd} \leq 1 - \boldsymbol{y}_{ps}^{s} \forall p, s \in T, p \neq s, d \in D \tag{8}$$

Constraint set (9) guarantees that the service of a given truck at the CDT may start only after its arrival to the assigned CDT door.

$$\boldsymbol{\tau}_{t}^{ST} \geq \tau_{t}^{AT} \forall t \in T \tag{9}$$

Constraint set (10) assures that the service of a given truck at the CDT may start only once the assigned CDT door becomes available in the considered planning horizon.

$$\boldsymbol{\tau}_{t}^{ST} \geq \sum_{d \in D} \tau_{d}^{DA} \boldsymbol{x}_{td} \forall t \in T \tag{10}$$

Constraint set (11) ensures that the service of a given truck at the CDT will not start before completing the service of the preceding trucks that are assigned to the same door of the CDT.

$$\boldsymbol{\tau}_{s}^{ST} \geq \boldsymbol{\tau}_{p}^{ST} + \sum_{d \in D} \tau_{pd}^{HT} \boldsymbol{x}_{pd} - \Gamma\left(1 - \boldsymbol{y}_{ps}^{s}\right) \forall p, s \in T, p \neq s \tag{11}$$

Constraint set (12) states that a given outbound truck can be served at the assigned CDT door only upon the beginning of service of the inbound trucks delivering the products, which will be further loaded into that outbound truck.

$$\boldsymbol{\tau}_{s}^{ST} \geq \boldsymbol{\tau}_{p}^{ST} r_{ps} \forall p \in T^{I}, s \in T^{O}, p \neq s \tag{12}$$

Constraint set (13) calculates the waiting time of every truck at the CDT.

$$\boldsymbol{\tau}_{t}^{WT} \geq \boldsymbol{\tau}_{t}^{ST} - \tau_{t}^{AT} \forall t \in T \tag{13}$$

Constraint set (14) computes the finish service time of every truck at the CDT.

$$\boldsymbol{\tau}_{t}^{FT} \geq \boldsymbol{\tau}_{t}^{ST} + \sum_{d \in D} \tau_{td}^{HT} \boldsymbol{x}_{td} \forall t \in T \tag{14}$$

Constraint set (15) estimates the temporary storage time of the products, which will be loaded into a given outbound truck, at the CDT.

$$\boldsymbol{\tau}_{s}^{PST} \geq \left(\boldsymbol{\tau}_{s}^{ST} - \boldsymbol{\tau}_{p}^{ST}\right) r_{ps} \forall p \in T^{I}, s \in T^{O}, p \neq s \tag{15}$$

Constraint set (16) calculates the early departure hours of every truck from the CDT.

$$\boldsymbol{\tau}_{t}^{ET} \geq \tau_{t}^{SD} - \boldsymbol{\tau}_{t}^{FT} \forall t \in T \tag{16}$$

Constraint set (17) calculates the delayed departure hours of every truck from the CDT.

$$\boldsymbol{\tau}_{t}^{DT} \geq \boldsymbol{\tau}_{t}^{FT} - \tau_{t}^{SD} \forall t \in T \tag{17}$$

5. Solution methodology

The polyploid EA-based solution algorithms were found to be effective for a wide range of different decision problems. Considering potential advantages of polyploidy, an Adaptive Polyploid Memetic Algorithm (APMA) was designed in this study to solve the CDT truck scheduling problem represented by the **IOTSP** mathematical model. A detailed description of the developed APMA algorithm is presented under this section of the manuscript. A set of basic principles behind APMA are introduced first. Then, the key algorithmic steps and the main APMA features are presented.

5.1. Basic principles

A thorough review of the relevant literature revealed that studies employing polyploid solution algorithms mainly consider solution quality and population diversity as the major performance indicators [14] often ignoring computational time. Nevertheless, computational time can significantly increase with increasing number of copies of each chromosome in polyploid individuals. Taking into account the latter feature, the developed APMA algorithm deploys the adaptive polyploid mechanism that controls the number of chromosome copies for each individual based on both solution quality and computational time criteria. Fig. 4 illustrates the basic principles behind the APMA design. The search process starts assuming that the APMA population is composed of Π individuals, where Π – population size. Every individual represented by a chromosome corresponds to a potential solution for the **IOTSP** mathematical model. The concept of polyploidy is explicitly implemented throughout the crossover operations. The parent chromosomes are assumed to be diploid at the beginning of the algorithmic run. After each crossover operation, applied with a specific crossover probability – σ^{cr}, two offspring chromo-

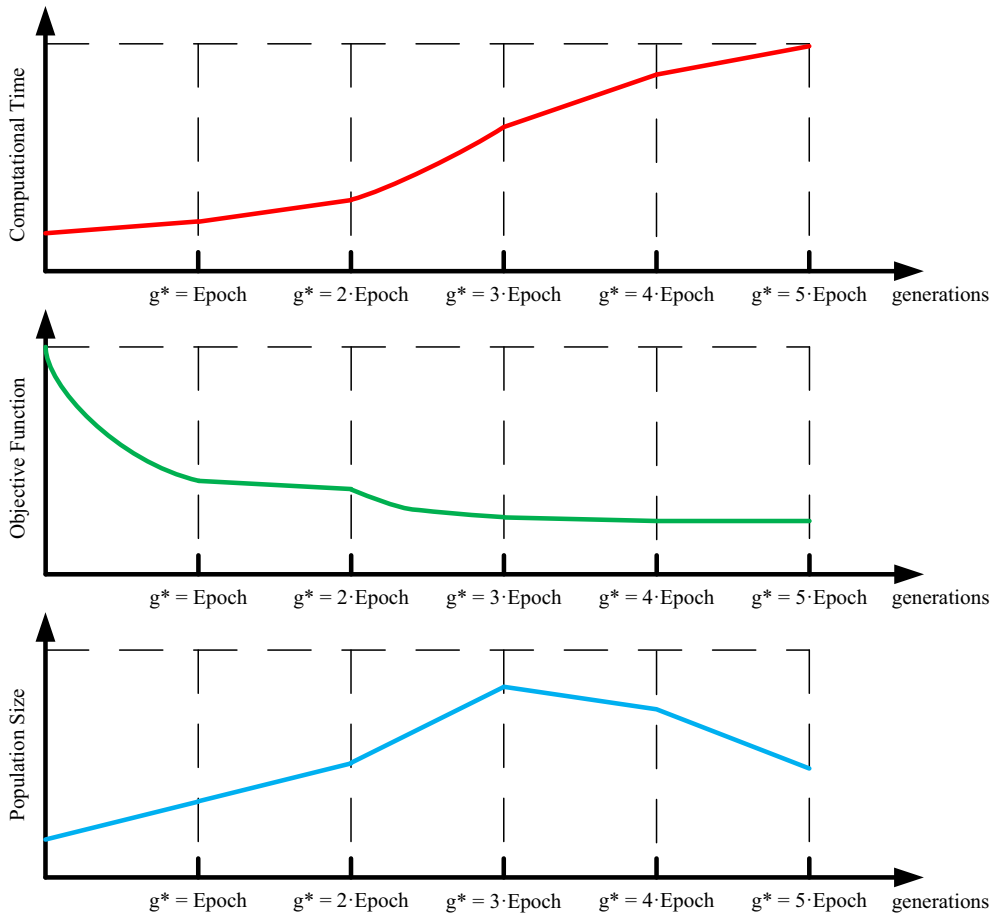


Fig. 4. The basic principles behind the APMA design.

somes will be produced and appended to the APMA population along with one copy of each parent chromosome (see Section 5.7.1 of the manuscript for more details). Therefore, the parent chromosomes will be allowed to compete with the newly produced offspring chromosomes in order to be present in the following generation. The APMA population begins increasing after the polyploidy application, which enhances the explorative capabilities of the algorithm.

After a certain number of generations, referred to as "epoch" in this study, APMA quantifies the changes in the objective function (ΔZ) and changes in the computational time (ΔT). If the computational time increase after *Epoch* generations does not exceed ΔT^* (%) and the objective function improvements do not exceed ΔZ^* (%), APMA will increase the number of copies of each parent chromosome stored before each crossover operation by one chromosome (e.g., diploid parent chromosomes will become triploid, triploid parent chromosomes will become tetraploid, etc.). Therefore, the APMA population starts increasing faster to facilitate the explorative capabilities of the algorithm even further and prevent convergence at the local optimum. The example, presented in Fig. 4, shows that the computational time increase between generations $g^* = Epoch$ and $g^* = 2 \cdot Epoch$ did not exceed ΔT^* (%), and the objective function improvements were not significant (i.e., below the pre-defined threshold ΔZ^*). Hence, APMA started increasing the population size after generation $g^* = 2 \cdot Epoch$ by increasing the number of copies of each parent chromosome stored before each crossover operation. Thus, the parent chromosomes became triploid. Note that along with increasing the population size, APMA deploys a number of problem-specific hybridization techniques (see Section 5.7.3 of the manuscript for more details) to improve solution quality even further.

A constant population size increase is expected to facilitate the search process for promising search space domains and high-quality solutions but will negatively impact computational time. If the computational time increase after *Epoch* generations exceeds ΔT^* (%), APMA will start decreasing the population size by removing all the parent chromosomes selected for the crossover operation (i.e., no parent chromosomes and no offspring chromosomes will be appended to the APMA population). When the computational time stabilizes, APMA will resume performing normal polyploid crossover operations. However, APMA will decrease the number of copies of each parent chromosome stored before each crossover operation by one chromosome (e.g., tetraploid parent chromosomes will become triploid, triploid parent chromosomes will become diploid, etc.). The example, presented in Fig. 4, shows that the computational time significantly increased between generations $g^* = 2 \cdot Epoch$ and $g^* = 3 \cdot Epoch$ (i.e., above the pre-defined threshold ΔT^*), and APMA started decreasing the population size

after generation $g^* = 3 \cdot Epoch$ by removing all the parent chromosomes selected for the crossover operation. Therefore, the developed APMA algorithm directly considers the tradeoff between the objective function improvements and computational time changes throughout the algorithmic run.

5.2. Main algorithmic steps

The main steps of the developed APMA algorithm are illustrated in Fig. 5. In step 0, APMA initializes the data structures that will be used throughout the algorithmic run. In step 1, the generation counter is initialized ($g = 1$). Furthermore, generation g^*, when the polyploidy adjustments are made and the hybridization techniques (i.e., memetic operations) are applied, is set equal to *Epoch* ($g^* = Epoch$). In step 2, APMA initializes the chromosomes and population for generation $g = 1$. In step 3, APMA evaluates fitness of the population chromosomes for generation $g = 1$. After that, APMA executes the main loop, where the stopping criterion is checked first. In case the stopping criterion is not satisfied, APMA proceeds to the next step. The generation counter is updated in step 5, while the parent chromosomes are further identified in step 6. Then, APMA performs basic operations, including crossover and mutation, for the identified parent chromosomes in step 7. In steps 9 and 10, APMA adjusts polyploidy and applies the hybridization techniques after *Epoch* generations (i.e., when $g = g^*$), respectively. In step 11, generation g^* is reset to make sure that the polyploidy adjustments will be made and the hybridization techniques will be applied every *Epoch* generation. In step 12, APMA evaluates fitness of the produced and mutated chromosomes, while the surviving chromosomes to be moved to the following generation are identified in step 13. Steps 5–13 are continuously repeated by APMA until the pre-defined stopping criterion is satisfied.

5.3. Solution representation

Selection of the appropriate solution representation is important as it may directly impact the overall algorithmic performance [13]. Two-dimensional integer chromosomes were adopted to represent the candidate solutions for the **IOTSP** mathematical model. An example of a chromosome is illustrated in Fig. 6, where a total of 8 inbound trucks and outbound trucks are assigned for service at the CDT that has 3 doors available. In particular, trucks "2" and "4" are assigned for service at CDT

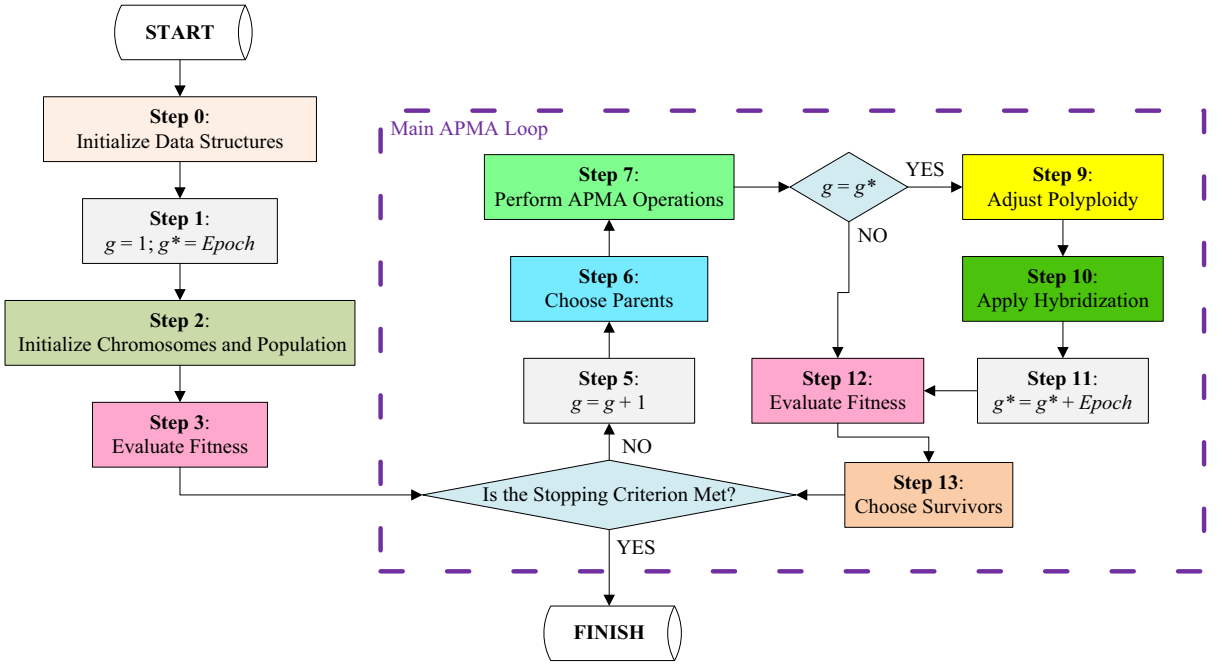


Fig. 5. The main steps of the developed APMA algorithm.

Serv. Order →	1	2	1	2	3	4	1	2
Truck →	2	4	8	3	1	6	5	7
Door →	1	1	2	2	2	2	3	3

Fig. 6. An illustrative example chromosome for the considered problem.

door "1" (truck "2" is served first, while truck "4" is served second). Trucks "8", "3", "1", and "6" are assigned for service at CDT door "2" and have service orders "1", "2", "3", and "4", respectively. On the other hand, trucks "5" and "7" are assigned for service at CDT door "3" (truck "5" is served first, while truck "7" is served second). The adopted chromosome representation does not require encoding truck service orders, as they can be identified based on the sequence of genes in the chromosome (the term "genes" is used for the individual chromosome components representing doors and trucks). Note that the location of a gene along the chromosome will be denoted using the term "locus" [13]. The actual value of a gene will be denoted using the term "allele" [13]. In the presented example (see Fig. 6), the genes with CDT door "2" and truck "8" have locus "3" (i.e., the locus counter starts from the left part of the chromosome and proceeds to the right). The CDT door identifier (i.e., CDT door "2") and the truck identifier (i.e., truck "8") are the alleles of the genes that are placed in locus "3".

5.4. Population initialization

Population initialization is a critical step in the design of metaheuristic algorithms. Random initialization of the population chromosomes can be considered as a common approach [13,46,47]. However, it will not be appropriate for the **IOTSP** mathematical model due to the truck service precedence constraints (since the service of an outbound truck may not be started prior to the service start of the inbound truck delivering the products for that particular outbound truck). A novel Truck Sequence Refinement (TSR) heuristic was developed as a part of this study in order to generate the chromosomes of the initial population. The developed TSR heuristic is based on the First Come First Served (FCFS) policy, according to which the trucks will be assigned to the available doors based on the order of their arrival at the CDT. In the meantime, TSR ensures that each outbound truck will be assigned for service after the last inbound truck delivering the products for that particular outbound truck. The TSR main steps are shown in Algorithm 1, where additional notation $\boldsymbol{\tau}_d^{UDA}, d \in D$ (hours) was adopted to represent the updated availability of CDT door d for truck service. The remaining notations used within Algorithm 1 are described in Table 1.

Algorithm 1. *Truck Sequence Refinement (TSR) Heuristic*

TSR$(T, D, \tau^{AT}, \tau^{DA}, \tau^{HT}, r)$
in: $T = \{1, \cdots, m\}$ - set of trucks; $D = \{1, \cdots, n\}$ - set of doors; τ^{AT} - truck arrival times; τ^{DA} - door availability; τ^{HT} - truck handling times; r - inbound-to-outbound truck assignment
out: $\boldsymbol{x}$ - initial truck-to-door assignment
0: $|T^S| \leftarrow m$; $W \leftarrow \oslash$; $\boldsymbol{\tau}^{UDA} \leftarrow \tau^{DA}$; $|\boldsymbol{x}| \leftarrow m \cdot n$; $|\boldsymbol{\tau}^{ST}| \leftarrow m$; $|\boldsymbol{\tau}^{FT}| \leftarrow m$ ◁ Initialization
1: $T^S \leftarrow$ ***SortAscend***(T, τ^{AT}) ◁ Sort the arriving trucks
2: $i \leftarrow 1$
3: **while** $i \leq |T^S|$ **do**
4: $d \leftarrow argmin_d(\boldsymbol{\tau}_d^{UDA})$ ◁ Identify the first available door at the CDT
5: **if** $max_i(r_{pi}) > 0$ **then**
6: $W \leftarrow W \cup \left\{i, argmax_p(r_{pi})\right\}$ ◁ Store the outbound truck and associated inbound truck
7: **else**
8: **if** $W = \oslash$ **then**
9: $t \leftarrow i$ ◁ Select a truck
10: **else if** $W \neq \oslash$ **and** $min\left(W^I\right) \geq i$ **then**
11: $t \leftarrow i$ ◁ Select a truck
12: **else**
13: $t \leftarrow W^O_{argmin(W^I)}$ ◁ Select a truck
14: $W \leftarrow W - \left\{W_{argmin(W^I)}\right\}$ ◁ Update the data structure with the outbound trucks
15: $i \leftarrow i - 1$
16: **end if**
17: **end if**
18: $\boldsymbol{x}_{td} \leftarrow 1$ ◁ Assign the selected truck to the first available door at the CDT
19: $\boldsymbol{\tau}_t^{ST} \leftarrow max(\tau_t^{AT}, \boldsymbol{\tau}_d^{UDA})$ ◁ Compute the start service time of a truck
20: $\boldsymbol{\tau}_t^{FT} \leftarrow \boldsymbol{\tau}_t^{ST} + \tau_{td}^{HT}$ ◁ Compute the finish service time of a truck
21: $\boldsymbol{\tau}_d^{UDA} \leftarrow \boldsymbol{\tau}_t^{FT}$ ◁ Update availability of the assigned CDT door
22: $i \leftarrow i + 1$
23: **end while**
24: **return** $\boldsymbol{x}$

In step 0, TSR initializes the data structures that will be used by the heuristic. In step 1, TSR sorts the arriving trucks based on their arrival times in ascending order. After that, TSR executes the main loop (steps 3–23). In step 4, TSR identifies the first available CDT door. If the next arriving truck is outbound, TSR stores that truck in the list (W) along with the identifier of the last inbound truck that delivers the products for that outbound truck (since the service of an outbound truck cannot be completed until the last inbound truck delivers the required products) in steps 5–7. If the next arriving truck is inbound, TSR checks a number of conditions before selecting the next truck for service (steps 8–16). If list W is empty, then TSR will select the next arriving inbound truck for service (step 9). Furthermore, if list W is not empty and the next inbound truck arrives before the earliest last inbound truck in list W, TSR will also assign the next arriving inbound truck for service (step 11). Otherwise, TSR will select the outbound truck with the earliest last inbound truck from list W (step 13). In step 14, the assigned outbound truck is removed from list W. In step 18, TSR assigns the selected truck (either inbound or outbound) for service at the first available CDT door. The start and finish service times for the assigned truck are further computed in steps 19 and 20, respectively. In step 21, TSR updates availability of the assigned CDT door. The heuristic is terminated once all the arriving trucks are scheduled for service at the CDT doors available.

The proposed TSR heuristic is expected to be more advantageous compared to the alternative population initialization approaches (e.g., assign the arriving trucks to be served at the CDT doors available randomly; assign the arriving inbound trucks to be served first before assigning any outbound trucks to ensure that the truck service precedence constraints will not be violated), as it not only captures the truck service precedence constraints but can also reduce the total waiting time of outbound trucks. For instance, assume that there are five trucks arriving for service at the CDT in the order "1" → "2" → "3" → "4" → "5". Trucks "2" and "4" are outbound, while trucks "1", "3", and "5" are inbound. Inbound truck "1" is delivering the products for outbound truck "2", while inbound trucks "3" and "5" are delivering the products for outbound truck "4". The TSR heuristic will assign the arriving trucks for service at the CDT in the order "1" → "2" → "3" → "5" → "4", ensuring that inbound truck "1" will be served before outbound truck "2", while inbound trucks "3" and "5" will be served before truck "4". On the other hand, assigning the arriving inbound trucks for service before any outbound trucks (i.e., the order "1" → "3" → "5" → "2" → "4") may result in a significant waiting time for outbound truck "2". Note that the entire APMA population will be created using TSR. The scope of this study includes a comprehensive evaluation of the developed TSR heuristic against the alternative mechanisms for population initialization (Section 6.3.1 of the manuscript presents more details).

5.5. Fitness function

After initializing the population chromosomes, APMA evaluates their fitness. Let $C = \{1, \cdots, a\}$ denote a set of chromosomes in the APMA population; and $G = \{1, \cdots, b\}$ denote a set of generations throughout the APMA evaluation. The developed APMA algorithm relies on the following function to evaluate fitness of chromosome c in generation g:

$$Fit_{cg} = \Omega\left[\sum_{t\in T}\left(\tau_t^{WT}\delta_t^{WT}\right) + \sum_{t\in T}\sum_{d\in D}\left(\tau_{td}^{HT}\boldsymbol{x}_{td}\delta_t^{HT}\right) + \sum_{t\in T}\left(\tau_t^{PST}\delta_t^{PST}\right) + \sum_{t\in T}\left(\tau_t^{ET}\delta_t^{ET}\right) + \sum_{t\in T}\left(\tau_t^{DT}\delta_t^{DT}\right)\right] \forall c \in C, g \in G \tag{18}$$

It can be noticed that the APMA fitness function includes all the components of the objective function $\boldsymbol{Z}$, which was adopted for the proposed **IOTSP** mathematical model. Furthermore, the APMA fitness function has an additional component (denoted as Ω) that is used to penalize the infeasible chromosomes. As a result of the APMA operations (i.e., crossover and mutation that are described under Sections 5.7.1 and 5.7.2 of the manuscript), the APMA population may have some chromosomes, where the service of an outbound truck starts prior to the service start of the inbound truck delivering the products for that particular outbound truck. Since the latter scenario does not replicate realistic CDT operations, such chromosomes will be penalized by APMA. The penalization approach ensures that infeasible chromosomes will have lower survival chances compared to feasible chromosomes in each generation. One of the key challenges with the implementation of the penalization approach consists in selection of the appropriate penalty value. Setting a fairly low penalty value may cause some negative implications throughout the algorithmic search (e.g., propagation of a large quantity of infeasible individuals from one generation to another). The appropriate value of the penalty term (Ω) will be established based on the parameter tuning analysis (Section 6.2 of the manuscript presents more details).

Note that there are some alternative approaches for handling infeasible individuals that include, but are not limited to, the following [13]: (1) maintain the feasibility of individuals by using special genetic operators; (2) maintain the feasibility of individuals by using specific chromosome representations; (3) repair the individuals that become infeasible throughout the search process; (4) remove the individuals that become infeasible from the population; and (5) apply particular types of decoders. The scope of future research for this study includes a detailed evaluation of the APMA performance under different approaches for handling infeasible individuals.

5.6. Procedure for selecting parents

After evaluation of the fitness of chromosomes in the initial population, APMA executes the main loop. In case the stopping criterion is not satisfied, APMA identifies the parent chromosomes using the Ranking Selection mechanism. The main steps performed by the Ranking Selection mechanism are shown in Algorithm 2. In step 0, the data structure for the new

parent chromosomes is initialized by the Ranking Selection mechanism. In step 1, the list of candidate chromosomes is created based on the available population of chromosomes. Then, fitness of the candidate chromosomes is determined in step 2. In steps 3 through 7, the Ranking Selection mechanism executes the main loop. In step 4, the fittest chromosome from the list of candidate chromosomes is identified. In step 5, the fittest chromosome is appended to the data structure with the new parent chromosomes. The list of candidate chromosomes is further updated in step 6. The parent selection procedure is terminated when the required number of the parent chromosomes has been chosen.

Algorithm 2. *Ranking Selection*

Ranking $\left(Pop_g, Fit_g\right)$
in: Pop_g - population in generation g; Fit_g - fitness of population chromosomes in generation g
out: $Parents_g$ - parent chromosomes in generation g
0: $Parents_g \leftarrow \oslash$ ◁ Initialization
1: $Pop_g^{cand} \leftarrow Pop_g \cup Pop_g$ ◁ Create the list of candidate chromosomes
2: $Fit_g^{cand} \leftarrow Fit_g \cup Fit_g$ ◁ Determine fitness of the candidate chromosomes
3: **while** $|Parents_g| \neq |Pop_g|$ **do**
4: $c^* \leftarrow argmin\left(Fit_g^{cand}\right)$ ◁ Identify the fittest chromosome
5: $Parents_g \leftarrow Parents_g \cup \left\{Pop_{c^*g}^{cand}\right\}$ ◁ Assign the fittest chromosome to become a parent
6: $Pop_g^{cand} \leftarrow Pop_g^{cand} - \left\{Pop_{c^*g}^{cand}\right\}$ ◁ Update the list of candidate chromosomes
7: **end while**
8: **return** $Parents_g$

5.7. APMA operations

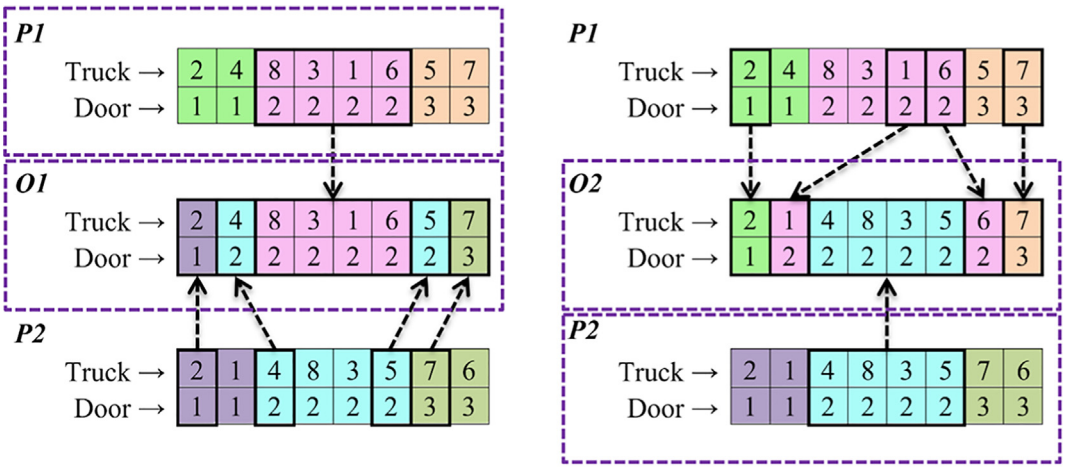
After selecting the parent chromosomes, APMA applies crossover and mutation in order to explore and exploit the search space domains. The offspring chromosomes to be generated by the crossover operator will be mutated. Furthermore, APMA adjusts polyploidy and applies the hybridization techniques to the produced offspring chromosomes after *Epoch* generations. A comprehensive description of the crossover operations, mutation operations, and memetic operations (i.e., application of the adaptive polyploid mechanism and the hybridization techniques) is presented under Sections 5.7.1 to 5.7.3 of the manuscript.

5.7.1. Crossover operations

Crossover operators enable the EA-based algorithms with explorative capabilities, so different domains of the search space can be explored [13]. As two-dimensional integer chromosomes were adopted to represent candidate solutions for the **IOTSP** mathematical model (see Fig. 6), only specific types of crossover operators will be appropriate (e.g., order crossover operator, cycle crossover operator, partially mapped crossover operator). Selection of the alternative crossover operators for the adopted integer chromosomes (e.g., single arithmetic crossover, one-point crossover) may cause infeasibility, since certain trucks may not be even scheduled for service at the CDT doors, while some other trucks can be scheduled for service more than once. APMA deploys the order crossover operator to produce the offspring chromosomes. Fig. 7 illustrates an example of the APMA crossover operation.

First, two parent chromosomes are chosen at random from the pool of available parent chromosomes with a specific crossover probability – σ^{cr}. Since APMA is polyploid, a number of copies of the parent chromosomes will be stored before performing a crossover operation. As it was indicated under Section 5.1 of the manuscript, the parent chromosomes are assumed to be diploid at the beginning of the algorithmic run. Hence, after each crossover operation, two offspring chromosomes will be produced and appended to the APMA population along with one copy of each parent chromosome. Fig. 7 shows how two offspring chromosomes "O1" and "O2" are produced after applying the order crossover operator. In particular, offspring chromosome "O1" is produced by copying a segment of gene arrays from parent chromosome "P1" and pasting the selected gene arrays into offspring chromosome "O1". The length of a segment may vary from one deployment of the crossover operator to another and is set randomly. In the considered example, the gene arrays that contain trucks "8", "3", "1", and "6" are copied directly from parent chromosome "P1" and pasted into offspring chromosome "O1". After that, the order crossover operator will copy the gene arrays with the trucks, which have not been assigned for service, directly from parent chromosome "P2". In particular, the gene arrays that contain trucks "2", "4", "5", and "7" are copied from parent chromosome "P2" and pasted into offspring chromosome "O1". The order crossover operator produces offspring chromosome "O2" in a similar manner.

As indicated earlier, the crossover operations generally cause substantial genetic changes in the population chromosomes that can ultimately worsen their fitness. The application of the polyploidy concept is expected to prevent decreasing fitness



Notes: Chromosomes "P1", "O1", "P2", and "O2" will be appended to the APMA population after completion of the crossover operation.

Fig. 7. An illustrative example of a polyploid crossover operation.

of the chromosomes in the APMA population as a result of applying the crossover operations. The number of copies of each parent chromosome to be stored before performing the crossover operations will be determined by APMA based on the tradeoff between the objective function improvements and computational time changes throughout the memetic operations.

5.7.2. Mutation operations

Mutation operators enable the EA-based algorithms with exploitative capabilities, so the promising domains identified by crossover operators can be further exploited for superior solutions [13]. APMA deploys the mutation operator that is based on the features of swap mutation, insert mutation, and invert mutation. A combination of different types of mutation is expected to enhance the exploitative capabilities of the developed mutation operator. The generated offspring chromosomes along with the stored copies of parent chromosomes will undergo mutation after the application of polyploid crossover operations. The main steps performed by the developed mutation operator are shown in Algorithm 3.

Algorithm 3. *Mutation Operator*

Mutation $\left(Pop_g, \sigma^{mut}\right)$
in: Pop_g - chromosomes in generation g; σ^{mut} - mutation probability
out: $\widetilde{Pop_g}$ - mutated chromosomes in generation g
0: $\left|\widetilde{Pop_g}\right| \leftarrow |Pop_g|$ ◁ Initialization
1: $c \leftarrow 1$
2: **while** $c \leq |Pop_g|$ **do**
3: $\varphi \leftarrow$ ***Rand***$(0,1)$ ◁ Generate a random value
4: **if** $0 \leq \varphi < 1/3$ **then**
5: $\widetilde{Pop}_{cg} \leftarrow$ ***Swap*** $\left(Pop_{cg}, \sigma^{mut}\right)$ ◁ Apply swap mutation
6: **else if** $1/3 \leq \varphi < 2/3$ **then**
7: $\widetilde{Pop}_{cg} \leftarrow$ ***Insert*** $\left(Pop_{cg}, \sigma^{mut}\right)$ ◁ Apply insert mutation
8: **else if** $2/3 \leq \varphi \leq 1$ **then**
9: $\widetilde{Pop}_{cg} \leftarrow$ ***Invert*** $\left(Pop_{cg}, \sigma^{mut}\right)$ ◁ Apply invert mutation
10: **end if**
11: $c \leftarrow c + 1$
12: **end while**
13: **return** $\widetilde{Pop_g}$

In step 0, the data structure is initialized by the mutation operator for the mutated population chromosomes. In steps 2 through 12, the mutation operator executes the main loop. In step 3, the mutation operator generates a random value (φ), which varies between 0 and 1. In case the generated random value falls between "0" and "1/3," swap mutation will be applied to a given population chromosome. Insert mutation will be applied to a given population chromosome when the generated random value falls between "1/3" and "2/3." On the other hand, if the generated random value falls between "2/3" and "1," invert mutation will be applied to a given population chromosome. Note that the selected mutation type will be used either for the gene arrays that represent the CDT doors or the gene arrays that represent the arriving trucks on a random basis (hence, the genes with the CDT doors and the genes with the arriving trucks will have a 50% probability of being mutated). Steps 2–12 are continuously repeated until each chromosome in the APMA population is mutated.

Fig. 8 illustrates several examples of a mutation operation. An illustrative example of performing swap mutation for a given population chromosome is presented in Fig. 8A, where truck "4", which was originally assigned to CDT door "1" after truck "2", is re-assigned to CDT door "2" after truck "3". On the other hand, truck "1", which was originally assigned to CDT door "2" after truck "3", is re-assigned to CDT door "1" after truck "2". An illustrative example of insert mutation is presented in Fig. 8B, where the gene with CDT door "2" is inserted in locus "2", and the remaining genes with CDT doors are shifted to the right. An illustrative example of invert mutation is presented in Fig. 8C, where the genes with trucks "1", "6", "5", and "7" undergo inversion (i.e., trucks "1" and "6", originally assigned to CDT door "2", are re-assigned to CDT door "3"; while trucks "5" and "7", originally assigned to CDT door "3", are re-assigned to CDT door "2"). The mutated two-dimensional chromosomes can be sorted by CDT doors in ascending order to prevent the truck service order disruptions (see Fig. 8B, where truck "4", assigned to CDT door "2", was originally placed by the mutation operator between trucks "2" and "8" that are both assigned to CDT door "1").

5.7.3. Memetic operations

In order to facilitate the search process, APMA applies a set of memetic operations after *Epoch* generations. Note that the memetic operations are not applied in every APMA generation to prevent increasing computational complexity of the algorithm. First, APMA deploys the adaptive polyploid mechanism, which controls the number of copies for each parent chromosome stored before each crossover operation based on both solution quality and computational time criteria. As discussed under Section 5.1 of the manuscript, if the computational time increase after *Epoch* generations does not exceed ΔT^* (%) and the objective function improvements do not exceed ΔZ^* (%), APMA will increase the number of copies of each parent chromosome stored before each crossover operation by one chromosome. Therefore, the APMA population will start increas-

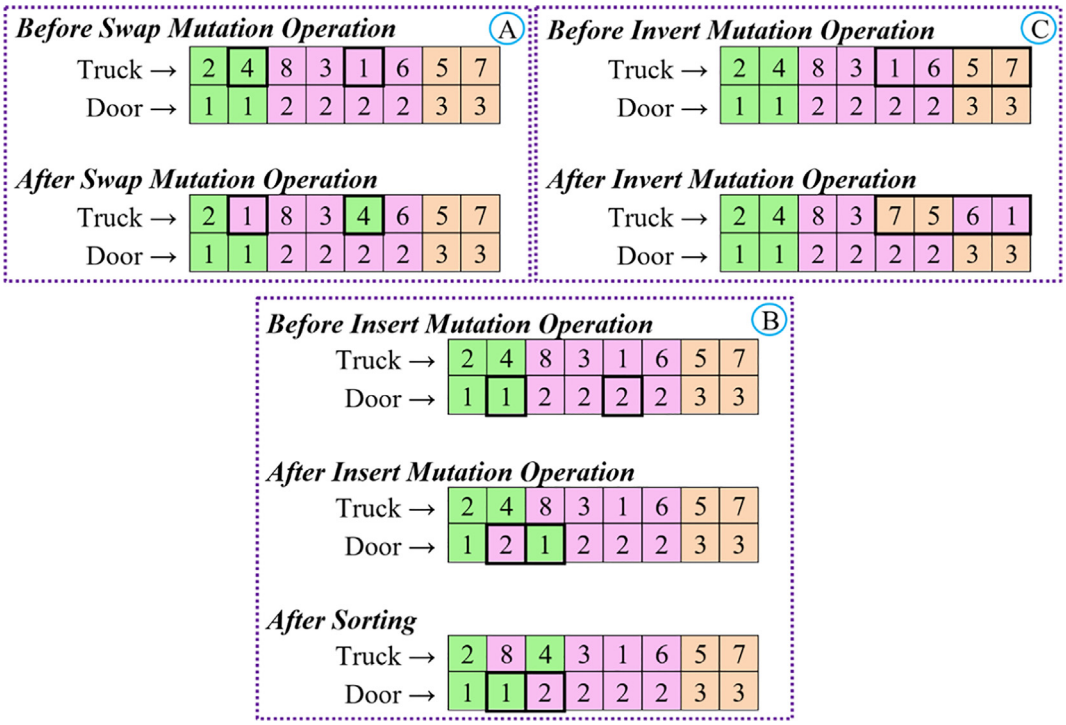


Fig. 8. Illustrative examples of a mutation operation.

ing faster to facilitate the explorative capabilities of the algorithm even further and prevent convergence at the local optimum. The parent chromosomes are assumed to be diploid at the beginning of the algorithmic run.

However, a constant population size increase may drastically increase computational time. Hence, if the computational time increase after *Epoch* generations exceeds ΔT^{*} (%), APMA will start decreasing the population size by removing all the parent chromosomes selected for the crossover operation (i.e., no parent chromosomes and no offspring chromosomes will be appended to the APMA population). When the computational time stabilizes, APMA will resume performing normal polyploid crossover operations. However, APMA will decrease the number of copies of each parent chromosome stored before each crossover operation by one chromosome. The appropriate values of *Epoch*, ΔT^{*}, and ΔZ^{*} will be established based on the parameter tuning analysis (Section 6.2 of the manuscript presents more details).

Second, APMA deploys a local search heuristic to each one of the mutated population chromosomes, aiming to improve their fitness. Throughout the APMA evolution, the arriving trucks will be distributed among the CDT doors available. The problem of the optimal truck sequencing at each CDT door (i.e., the optimal sequence of service for the trucks assigned to a given CDT door) has high computational complexity, and commercial optimization solvers (e.g., GUROBI, MOSEK, CPLEX) may require a significant amount of computational time. However, solving the optimal truck sequencing problem for a subset of trucks (i.e., a "string" of trucks) at a given CDT door will require less computational efforts. Moreover, the truck service precedence constraints (the service of outbound trucks may not be started before the service start of the corresponding inbound trucks) should be directly accounted for in the optimal truck sequencing problem. The developed APMA algorithm focuses on optimizing the truck sequence for the longest string of consecutive outbound trucks ($t \in T^{O}$) at one of the CDT doors (selected randomly); so that the start service times of the inbound trucks ($\tau_{p}^{ST}, p \in T^{I}$) delivering the products for the corresponding outbound trucks can be treated as parameters that will be defined throughout the crossover and mutation operations for each population chromosome. A mathematical formulation for the Outbound Truck Sequencing Problem (**OTSP**) has some similarities with the **IOTSP** mathematical model and can be presented as follows:

Outbound Truck Sequencing Problem (OTSP):

$$min\ \mathbf{Z}^{O} = \left[\sum_{t\in T^{O}}\left(\boldsymbol{\tau}_{t}^{WT}\delta_{t}^{WT}\right) + \sum_{t\in T^{O}}\sum_{d\in D}\left(\tau_{td}^{HT}x_{td}\delta_{t}^{HT}\right) + \sum_{t\in T^{O}}\left(\boldsymbol{\tau}_{t}^{PST}\delta_{t}^{PST}\right) + \sum_{t\in T^{O}}\left(\boldsymbol{\tau}_{t}^{ET}\delta_{t}^{ET}\right) + \sum_{t\in T^{O}}\left(\boldsymbol{\tau}_{t}^{DT}\delta_{t}^{DT}\right)\right] \tag{19}$$

Subject to:

$$\boldsymbol{y}_{s}^{f} + \sum_{p\in T^{O}:p\neq s}\boldsymbol{y}_{ps}^{s} = 1\forall s \in T^{O} \tag{20}$$

$$\boldsymbol{y}_{p}^{l} + \sum_{s\in T^{O}:s\neq p}\boldsymbol{y}_{ps}^{s} = 1\forall p \in T^{O} \tag{21}$$

$$\sum_{t\in T^{O}}\boldsymbol{y}_{t}^{f} = 1 \tag{22}$$

$$\sum_{t\in T^{O}}\boldsymbol{y}_{t}^{l} = 1 \tag{23}$$

$$\boldsymbol{\tau}_{t}^{ST} \geq \tau_{t}^{AT}\forall t \in T^{O} \tag{24}$$

$$\boldsymbol{\tau}_{t}^{ST} \geq \sum_{d\in D}\tau_{d}^{DA}x_{td}\forall t \in T^{O} \tag{25}$$

$$\boldsymbol{\tau}_{s}^{ST} \geq \boldsymbol{\tau}_{p}^{ST} + \sum_{d\in D}\tau_{pd}^{HT}x_{pd} - \Gamma\left(1 - \boldsymbol{y}_{ps}^{s}\right)\forall p, s \in T^{O}, p\neq s \tag{26}$$

$$\boldsymbol{\tau}_{s}^{ST} \geq \tau_{p}^{ST}r_{ps}\forall p \in T^{I}, s \in T^{O}, p\neq s \tag{27}$$

$$\boldsymbol{\tau}_{t}^{WT} \geq \boldsymbol{\tau}_{t}^{ST} - \tau_{t}^{AT}\forall t \in T^{O} \tag{28}$$

$$\boldsymbol{\tau}_{t}^{FT} \geq \boldsymbol{\tau}_{t}^{ST} + \sum_{d\in D}\tau_{td}^{HT}x_{td}\forall t \in T^{O} \tag{29}$$

$$\boldsymbol{\tau}_s^{PST} \geq \left(\boldsymbol{\tau}_s^{ST} - \tau_p^{ST}\right) r_{ps} \forall p \in T^I, s \in T^O, p \neq s \tag{30}$$

$$\boldsymbol{\tau}_t^{ET} \geq \tau_t^{SD} - \boldsymbol{\tau}_t^{FT} \forall t \in T^O \tag{31}$$

$$\boldsymbol{\tau}_t^{DT} \geq \boldsymbol{\tau}_t^{FT} - \tau_t^{SD} \forall t \in T^O \tag{32}$$

The objective function (19) of the **OTSP** mathematical model minimizes the total cost incurred by the CDT operator throughout the service of considered outbound trucks ($\mathbf{Z}^O$ – measured in USD). Constraint set (20) indicates that every truck in the considered string of outbound trucks will be either assigned for service as the first truck at the selected CDT door or after another outbound truck. Constraint set (21) indicates that every truck in the considered string of outbound trucks will be either assigned for service as the last truck at the selected CDT door or before another outbound truck. Constraint set (22) ensures that only one outbound truck will be assigned for service as the first truck in the considered string of outbound trucks at the selected CDT door. Constraint set (23) ensures that only one outbound truck will be assigned for service as the last truck in the considered string of outbound trucks at the selected CDT door. Constraint set (24) guarantees that the service of an outbound truck at the CDT may start only after its arrival to the assigned CDT door. Constraint set (25) indicates that the service of a given outbound truck at the CDT may start only once the assigned CDT door becomes available in the considered planning horizon. Constraint set (26) guarantees that the service of a given outbound truck at the CDT will not start before service completion of the preceding trucks in the considered string of outbound trucks.

Constraint set (27) enforces the condition that a given outbound truck can be served only upon the beginning of the service of the inbound trucks delivering the products, which will be further loaded into that outbound truck. Constraint sets (28) and (29) calculate the waiting time and finish service time of every truck in the considered string of outbound trucks. Constraint set (30) estimates the storage time of the products, which will be loaded into a given outbound truck. Constraint sets (31) and (32) calculate the early departure hours and delayed departure hours of every truck in the considered string of outbound trucks. Note that, unlike the **IOTSP** mathematical model, the **OTSP** mathematical model treats components $x_{td}, t \in T^O, d \in D$ and $\tau_p^{ST}, p \in T^I$ as parameters, since their values will be known after conducting the crossover and mutation operations. The developed APMA algorithm applies the Optimal Truck Sequence Identification (OTSI) heuristic to periodically solve the **OTSP** mathematical model (i.e., every *Epoch* generations) and identify the optimal sequence for outbound trucks at one of the CDT doors (selected randomly) for the longest string of consecutive outbound trucks in each population chromosome. The main steps performed by the OTSI heuristic are shown in Algorithm 4.

In step 0, the data structure for the updated population chromosomes is initialized by OTSI. In step 1, OTSI checks if the current generation counter (g) is equal to generation g^* (i.e., the generation in which the polyploidy adjustments are made and the hybridization techniques are applied). When $g = g^*$, the **OTSP** mathematical model will be solved to global optimality for the longest string of consecutive outbound trucks for each one of the mutated population chromosomes (steps 3–6). In step 7, generation g^* is reset by OTSI to make sure that the **OTSP** mathematical model will be solved to global optimality for each one of the mutated population chromosomes again after *Epoch* generations. In step 9, OTSI returns the updated chromosomes to the APMA population. Note that CPLEX will be used in this study to solve the **OTSP** mathematical model to global optimality. Fig. 9 illustrates an example of how the OTSI heuristic selects the outbound trucks for optimizing their service sequence. In the considered example, a total of 16 trucks are assigned for service at CDT door "2" (therefore, CDT door "2" was selected by OTSI to optimize the truck service sequence for a given population chromosome). There are a total of three strings of outbound trucks. The OTSI heuristic will be executed for outbound trucks "2", "4", "13", "9", "10", and "12", since they form the longest string of consecutive outbound trucks at CDT door "2" of the considered population chromosome.

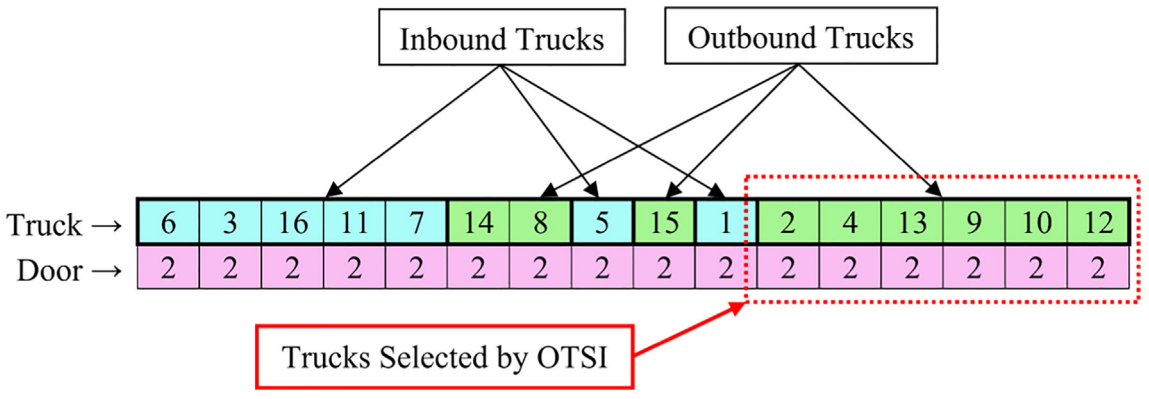


Fig. 9. An illustrative example of outbound truck selection by the OTSI heuristic.

Algorithm 4. *Optimal Truck Sequence Identification (OTSI) Heuristic*

OTSI$\left(InData, \widetilde{Pop_g}, T^O, Epoch\right)$

in: *InData* - input parameters for the **OTSP** mathematical model; $\widetilde{Pop_g}$ - mutated chromosomes in generation g; T^O - string of outbound trucks selected for optimization; *Epoch* - epoch value

out: $\overline{Pop_g}$ - updated chromosomes in generation g

0: $\overline{Pop_g} \leftarrow \widetilde{Pop_g}$ ◁ Initialization
1: **if** $g = g^*$ **then**
2: $c \leftarrow 1$
3: **for all** $c \in \overline{Pop_g}$ **do**
4: $\overline{Pop_{cg}} \leftarrow$ ***OTSP***$\left(InData, \overline{Pop_{cg}}, T^O\right)$ ◁ Solve the **OTSP** mathematical model
5: $c \leftarrow c + 1$
6: **end for**
7: $g^* \leftarrow g^* + Epoch$ ◁ Reset generation g^*
8: **end if**
9: **return** $\overline{Pop_g}$

5.8. Procedure for selecting survivors

After producing the offspring chromosomes via the APMA operations and fitness evaluation, APMA identifies the surviving chromosomes that will represent the next generation population from the pool of available chromosomes. The Stochastic Universal Sampling mechanism is used by APMA to identify the next generation chromosomes. The main steps performed by the Stochastic Universal Sampling mechanism are shown in Algorithm 5. In step 0, the Stochastic Universal Sampling mechanism initializes the data structures that will be used. In steps 2–5, the fitness values of the population chromosomes are adjusted, since the **IOTSP** mathematical model is of a minimization type. In step 6, the Stochastic Universal Sampling mechanism computes the fitness interval (λ). In step 7, the available chromosomes are sorted, based on the adjusted fitness values, in descending order. In step 8, the starting fitness value (φ), varying between "0" and "λ," is chosen. Then, the Stochastic Universal Sampling mechanism executes another loop (steps 10–14). Within the loop, the next generation chromosomes are sampled from the pool of available chromosomes using the evenly spaced fitness intervals. The survivor selection procedure is terminated when the required number of the next generation chromosomes has been chosen.

Algorithm 5. *Stochastic Universal Sampling*

StocUnSampl$\left(Pop_g, Fit_g\right)$

in: Pop_g - chromosomes in generation g; Fit_g - fitness of chromosomes in generation g

out: Pop_{g+1} - chromosomes selected to represent generation $g + 1$

0: $|Pop_{g+1}| \leftarrow |Pop_g|$; $|Fit_g^{adj}| \leftarrow |Pop_g|$ ◁ Initialization
1: $c \leftarrow 1$
2: **while** $c \leq |Pop_g|$ **do**
3: $Fit_{cg}^{adj} \leftarrow 1/Fit_{cg}$ ◁ Compute the adjusted fitness values
4: $c \leftarrow c + 1$
5: **end while**
6: $\lambda \leftarrow \left(\sum_c Fit_{cg}^{adj}\right)/|Fit_g^{adj}|$ ◁ Compute the fitness interval
7: $Pop_g \leftarrow$ ***Sort***$\left(Pop_g, Fit_g^{adj}\right)$ ◁ Sort the chromosomes based on the adjusted fitness
8: $\varphi \leftarrow$ ***Rand***$(0, \lambda)$ ◁ Choose the starting fitness value
9: $c \leftarrow 1$
10: **while** $c \leq |Pop_g|$ **do**
11: $Pop_{c(g+1)} \leftarrow$ ***ChromSel***$\left(\varphi, Fit_g^{adj}, Pop_g\right)$ ◁ Choose the next generation chromosome
12: $\varphi \leftarrow \varphi + \lambda$ ◁ Update the starting fitness value
13: $c \leftarrow c + 1$
14: **end while**
15: **return** Pop_{g+1}

5.9. Elitism

The application of stochastic APMA operators (e.g., crossover operator, mutation operator) does not ensure that the offspring chromosomes with superior values of fitness function (compared to the parent chromosomes) will be produced. In some instances, the stochastic APMA operators may worsen the quality of population chromosomes (despite the application of the polyploidy concept). In order to prevent the loss of good-quality chromosomes throughout the algorithmic run, the elitism strategy has been often deployed in evolutionary computation [13]. The developed APMA algorithm applies the elitism strategy before the parent selection to make sure that the fittest chromosome will be moved to the population in the next generation.

5.10. Stopping criteria

The proposed APMA algorithm will be stopped when one of the following criteria has been satisfied: (1) no improvements in the objective function have been identified by APMA after a pre-specified number of consecutive generations (g^{obj}); (2) APMA reaches the maximum number of generations (g^{max}). The appropriate values of g^{obj} and g^{max} will be established based on the parameter tuning analysis (Section 6.2 of the manuscript presents more details). Technically, an additional parameter could be introduced for the first stopping criterion [e.g., the objective function improvements do not exceed Z^{obj} (%) after a pre-specified number of consecutive generations g^{obj}]. However, the introduction of additional parameters for the stopping criterion is anticipated to increase the amount of computational time required for tuning of all the parameters of the APMA algorithm, which is not considered as desirable from the practical point of view.

6. Computational experiments

The computational experiments, which were performed as a part of this study, are described under this section of the manuscript. During the experiments, the computational performance of the developed APMA algorithm was evaluated in three steps. The first step focused on assessing the effectiveness of the TSR heuristic that was developed for initializing the APMA population against the alternative mechanisms for population initialization. The second step focused on assessing the effects of polyploidy and hybridization within the developed APMA algorithm. The third step focused on assessing the performance of the developed APMA algorithm based on a comprehensive comparative analysis against the alternative solution approaches, including the following: (i) CPLEX; (ii) Ant Colony Optimization (ACO); (iii) Tabu Search (TS); (iv) Variable Neighborhood Search (VNS); and (v) Simulated Annealing (SA). While CPLEX is considered as an effective exact optimization approach for large-scale mixed integer linear programming models, ACO, TS, VNS, and SA are well-recognized state of the art metaheuristics that have been extensively used for solving challenging decision problems in the CDT operations literature as well as the other studies on freight terminal operations [1,2,44,48,49].

This study refers to Boloori Arabani et al. [48], Rajabi and Shirazi [49], and Liao et al. [44] for a more detailed description of the ACO, TS, VNS, and SA algorithms. MATLAB 2016a was used to encode the APMA, ACO, TS, VNS, and SA algorithms. A CPU with 32 GB of RAM, Dell Intel(R) Core™ i7 Processor, and Operating System Windows 10 was used to perform all the computational experiments as a part of this study. The following sections of the manuscript further elaborate on the input data selection for the **IOTSP** mathematical model, the tuning of parameters for the considered solution algorithms, and the aforementioned three steps that were undertaken to evaluate the APMA computational performance.

6.1. Input data selection for the IOTSP mathematical model

The existing studies on cross-docking operations and freight terminal operations were used to assign the appropriate parameter values for the **IOTSP** mathematical model [1,2,4,5,44,45,49]. Table 2 provides more details regarding the input data generation. The inbound and outbound trucks were assumed to arrive at the CDT following an exponential distribution (term "*EXP*" is used in Table 2 to denote the pseudorandom numbers that follow an exponential distribution). The inbound and outbound trucks were assumed to have an inter-arrival time of 10 min on average, which corresponds to approximately 0.1667 h. The CDT doors were assumed to be available for service of the arriving trucks from the beginning of the planning horizon: $\tau_d^{DA} = 0 \forall d \in D$ (hours). The handling time of the arriving trucks was assumed to vary between 0.50 h and 2.50 h and was generated as follows: $\tau_{td}^{HT} = U[0.50; 2.50] \forall t \in T, d \in D$ (hours), where term "$U[Val_1; Val_2]$" is used to denote the pseudorandom numbers that follow a uniform distribution. The scheduled departure time of truck t from the CDT was generated as follows: $\tau_t^{SD} = \tau_t^{AT} + min_d(\tau_{td}^{HT}) \cdot U[1.2; 1.5] \forall t \in T$ (hours). Each inbound truck, which arrives to be served at the CDT, was assumed to deliver the products for no more than 3 outbound trucks ($\sum_{s \in T^O: s \neq p} r_{ps} \leq 3 \forall p \in T^I$).

A set of uniform distributions were used to assign the unit cost components of the **IOTSP** mathematical model (see Table 2). In particular, the unit waiting cost of trucks was assumed to range from 100 USD/hour to 150 USD/hour, while the unit handling cost of trucks varied between 200 USD/hour and 300 USD/hour. The unit product inventory cost ranged from 40 USD/hour to 80 USD/hour. On the other hand, the unit cost of early truck departures and the unit cost of late truck departures were both assumed to vary between 300 USD/hour and 400 USD/hour. The generated parameter values were fur-

Table 2
The input data for the parameters of the **IOTSP** mathematical model.

Parameter	Adopted Values
Amount of the arriving trucks: $\|T\|$ (trucks)	Depends on the problem instance
Amount of the CDT doors available: $\|D\|$ (doors)	Depends on the problem instance
Truck inter-arrival time: $\Delta\tau^{AT}$ (hours)	$\Delta\tau^{AT} = EXP[0.1667]$
Scheduled arrival time of truck t: $\tau_t^{AT}, t \in T$ (hours)	$\tau_t^{AT} = \tau_{t-1}^{AT} + \Delta\tau^{AT} \forall t \in T$
Time when door d becomes available: $\tau_d^{DA}, d \in D$ (hours)	$\tau_d^{DA} = 0 \forall d \in D$
Handling time of truck t at door d: $\tau_{td}^{HT}, t \in T, d \in D$ (hours)	$\tau_{td}^{HT} = U[0.50; 2.50] \forall t \in T, d \in D$
Scheduled departure time of truck t from the CDT: $\tau_t^{SD}, t \in T$ (hours)	$\tau_t^{SD} = \tau_t^{AT} + min_d(\tau_{td}^{HT}) \cdot U[1.2; 1.5] \forall t \in T$
Inbound-to-outbound truck assignment: $r_{ps}, p \in T^I, s \in T^O, p \neq s$	$\sum_{s \in T^O : s \neq p} r_{ps} \leq 3 \forall p \in T^I$
Unit waiting cost of truck t: $\delta_t^{WT}, t \in T$ (USD/hour)	$\delta_t^{WT} = U[100; 150] \forall t \in T$
Unit handling cost of truck t: $\delta_t^{HT}, t \in T$ (USD/hour)	$\delta_t^{HT} = U[200; 300] \forall t \in T$
Unit inventory cost of the products, which will be loaded into truck t: $\delta_t^{PST}, t \in T$ (USD/hour)	$\delta_t^{PST} = U[40; 80] \forall t \in T$
Unit cost of early departure for truck t from the CDT: $\delta_t^{ET}, t \in T$ (USD/hour)	$\delta_t^{ET} = U[300; 400] \forall t \in T$
Unit cost of delayed departure for truck t from the CDT: $\delta_t^{DT}, t \in T$ (USD/hour)	$\delta_t^{DT} = U[300; 400] \forall t \in T$
Large positive number: Γ	$10,000$

ther used to develop a total of 30 problem instances by making the changes in the total amount of the CDT doors available (from 2 CDT doors to 10 CDT doors) and the total amount of the arriving trucks (from 8 trucks to 140 trucks).

6.2. Tuning of the algorithmic parameters

Before analyzing the performance of the considered solution algorithms, it is necessary to set the appropriate values for their parameters. The considered solution algorithms have quite a significant number of parameters. For example, the developed APMA algorithm has a total of 9 parameters, which include the following: (i) population size – Π; (ii) penalty for infeasible individuals – Ω; (iii) crossover probability – σ^{cr}; (iv) mutation probability – σ^{mut}; (v) number of generations between the polyploidy adjustments and application of the hybridization techniques – *Epoch*; (vi) threshold for the computational time increase – ΔT^*; (vii) threshold for the objective function improvements – ΔZ^*; (viii) number of consecutive generations without any improvements in the objective function – g^{obj}; and (ix) maximum number of generations – g^{max}.

The APMA parameters define the APMA computational complexity. Increasing population size enhances the explorative APMA capabilities but will increase its computational time as well (since the APMA procedures will have to be applied to a larger number of chromosomes). Increasing crossover and mutation probabilities is also expected to increase the APMA computational time, as the crossover and mutation operations will be applied more often to the APMA chromosomes. Similarly, increasing the maximum number of generations will increase the APMA computational time, as the APMA procedures will have to be applied for more generations. Furthermore, the adopted parameter values of the **IOTSP** mathematical model can influence the APMA computational time. For example, increasing amount of the arriving trucks will increase the length of the APMA chromosomes, which will further increase the computational time required to conduct the fitness function evaluations, crossover operations, and mutation operations.

Considering the number of APMA parameters, evaluation of all the possible combinations of parameters may not be feasible from the computational time standpoint. A "Taguchi's method" was used in this study for tuning of the algorithmic parameters, where each candidate algorithm was executed only for "the most favorable" combinations of parameters [45,48]. A tradeoff between the computational time required and objective function value, recorded for a given combination of parameters, was the primary criterion for the identification of the most favorable combinations of parameters. The tuning of algorithmic parameters was performed using 4 problem instances, which were sampled at random from the generated 30 problem instances (see Section 6.1 of the manuscript for the description of input data and problem instances). Throughout the analysis, each algorithm was launched 10 times for each most favorable combination of parameters in order to compute the average values of computational time and objective function. Table 3 presents the parameter tuning results for the considered solution algorithms and provides the following data: (a) names of the solution algorithms; (b) parameters of the solution algorithms; (c) candidate values used for each parameter; and (d) the most promising value for each parameter (selected based on a tradeoff between the computational time required and objective function value).

6.3. Algorithmic performance

All the steps that were used to evaluate the computational performance of the developed APMA algorithm are described under this section of the manuscript, including the following: (i) evaluation of the mechanism for population initialization; (ii) evaluation of the effects of polyploidy and hybridization; and (iii) comparative analysis against the alternative solution approaches.

Table 3
The results of parameter tuning for the considered solution algorithms.

Algorithm	Parameter	Candidate Values	Best Value	Algorithm	Parameter	Candidate Values	Best Value
APMA	Population size (Π)	[30; 40; 50]	40	ACO	Maximum number of iterations ($iter^{max}$)	[2,000; 2,500; 3,000]	3,000
APMA	Penalty term (Ω)	[4.00; 5.00; 6.00]	4.00	TS	Number of solutions evaluated during the local search (Π^{TS})	[30; 40; 50]	40
APMA	Crossover probability (σ^{cr})	[0.30; 0.50; 0.70]	0.30	TS	Penalty term (Ω)	[4.00; 5.00; 6.00]	4.00
APMA	Mutation probability (σ^{mut})	[0.01; 0.03; 0.05]	0.01	TS	Exchange rate (σ^{TS})[1]	[2; 4; 6]	2
APMA	Number of generations between the polyploidy adjustments and application of the hybridization techniques (*Epoch*)	[500; 600; 700]	600	TS	Tabu List size (Y)	[10; 15; 20]	10
APMA	Threshold for the computational time increase (ΔT^*, %)	[10; 30; 50]	30	TS	Maximum number of iterations ($iter^{max}$)	[2,000; 2,500; 3,000]	3,000
APMA	Threshold for the objective function improvements (ΔZ^*, %)	[2; 5; 10]	5	VNS	Number of solutions evaluated during the neighborhood search (Π^{VNS})	[30; 40; 50]	40
APMA	Number of consecutive generations without any improvements in the objective function (g^{obj})	[1,000; 1,250; 1,500]	1,000	VNS	Penalty term (Ω)	[4.00; 5.00; 6.00]	4.00
APMA	Maximum number of generations (g^{max})	[2,000; 2,500; 3,000]	3,000	VNS	Exchange rate (σ^{VNS})[1]	[2; 4; 6]	2
ACO	Population size (Π)	[30; 40; 50]	40	VNS	Maximum number of iterations ($iter^{max}$)	[2,000; 2,500; 3,000]	3,000
ACO	Penalty term (Ω)	[4.00; 5.00; 6.00]	4.00	SA	Initial temperature (ω^0)	[1,500; 1,750; 2,000]	1,500
ACO	Evaporation rate (ρ)	[0.20; 0.30; 0.40]	0.20	SA	Temperature interval ($\Delta\omega$)[2]	[0.10; 0.20; 0.50]	0.50
ACO	Initial pheromone amount ($\tau^0_{ps}, p, s \in T, p \neq s$)	[0.25; 0.30; 0.35]	0.30	SA	Penalty term (Ω)	[4.00; 5.00; 6.00]	4.00
ACO	Pheromone trail parameter (α)	[1.00; 1.50; 2.00]	1.50	SA	Exchange rate (σ^{SA})[1]	[2; 4; 6]	2
ACO	Heuristic parameter (β)	[1.00; 1.50; 2.00]	1.50	SA	Maximum number of iterations ($iter^{max}$)	[2,000; 2,500; 3,000]	3,000

1 – The exchange rate within the TS, VNS, and SA algorithms defines the number of truck-to-door assignments to be altered in the considered solution in order to create a new solution as a result of the local search; 2 – The proposed SA algorithm establishes the temperature in iteration *iter* as follows: $\omega_{iter} = \omega^0 - \Delta\omega \cdot iter$ (temperature units).

6.3.1. Evaluation of the mechanism for population initialization

As it was indicated under Section 5.4 of the manuscript, a novel TSR heuristic was designed to generate the initial population chromosomes for the developed APMA algorithm. As a part of the computational experiments, the TSR heuristic was compared against the alternative mechanisms for population initialization that have been used in the CDT truck scheduling literature, such as: (i) ITPC – the entire APMA population is generated using the Inbound Truck Precedence Constraint (ITPC) heuristic, where the arriving inbound trucks are scheduled for service first before scheduling any outbound trucks to ensure that the truck service precedence constraints will not be violated; (ii) R – the entire APMA population is generated based on a random assignment of the arriving inbound trucks and outbound trucks to the CDT doors available; (iii) TSR-ITPC – half of the APMA population is generated using TSR, and the other half is generated using ITPC; (iv) TSR-R – half of the APMA population is generated using TSR, while the other half is generated randomly; and (v) ITPC-R – half of the APMA population is generated using ITPC, while the other half is generated randomly.

The APMA variations, which were differed based on the population initialization mechanism used, were evaluated for all the generated 30 problem instances (see Section 6.1 of the manuscript for the description of input data and problem instances). Throughout the analysis, each APMA variation was launched 10 times in order to compute the average values of computational time and objective function. Table 4 presents the analysis results for the considered APMA variations with different population initialization mechanisms and provides the following data: (i) the problem instance number; (ii) the total amount of CDT doors available; (iii) the total amount of arriving inbound trucks and outbound trucks; (iv) the objective function values (average over 10 replications); and (v) the computational time values (average over 10 replications). Furthermore, Fig. 10 shows the convergence diagrams, which were recorded for the last replication of each APMA variation. Note that the convergence diagrams are provided for the problem instances with the largest number of CDT doors available and the largest number of arriving trucks (i.e., problem instances 21 through 30). However, similar tendencies for the remaining problem instances were noticed.

The conducted analysis shows that APMA with the TSR population initialization mechanism clearly outperformed the other APMA variations that rely on the alternative population initialization mechanisms. In particular, TSR demonstrated

Table 4
The objective function values and computational time required for the population initialization mechanisms considered.

Instance	#Doors	#Trucks	TSR		ITPC		R		TSR-ITPC		TSR-R		ITPC-R	
			Z, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec
1	2	8	15.359	58.12	15.359	57.66	15.359	61.98	15.359	63.78	15.359	64.71	15.359	60.71
2	2	10	21.748	63.63	21.748	64.48	21.748	66.00	21.748	66.09	21.748	67.58	21.748	65.93
3	2	12	26.304	66.11	26.304	70.56	26.304	73.54	26.304	72.05	26.304	71.74	26.304	67.02
4	2	14	34.817	72.20	34.817	78.54	34.817	73.58	34.817	74.92	34.817	76.02	34.817	73.89
5	2	16	46.577	76.15	46.577	83.14	46.967	79.88	46.577	79.56	46.577	81.49	46.577	78.25
6	4	8	8.229	61.54	8.229	61.62	8.229	59.52	8.229	65.28	8.229	65.74	8.229	60.60
7	4	10	11.417	65.65	11.417	70.51	11.417	68.41	11.417	68.47	11.417	66.33	11.417	67.03
8	4	12	13.747	68.54	13.747	68.32	13.747	76.33	13.747	72.82	13.747	70.45	13.747	70.81
9	4	14	17.810	67.49	17.810	76.10	17.928	76.02	17.810	73.01	17.810	74.33	17.810	83.06
10	4	16	23.167	73.07	23.271	83.23	23.414	85.83	23.207	80.91	23.254	84.93	23.292	90.23
11	8	50	73.464	142.83	74.411	163.87	74.861	160.52	73.523	159.16	73.599	166.62	74.669	179.82
12	8	60	100.123	158.46	100.371	192.64	100.450	184.94	100.274	189.85	100.328	197.36	100.394	172.82
13	8	70	130.386	191.53	131.020	205.01	133.093	201.56	130.591	218.56	130.642	222.21	132.644	190.65
14	8	80	172.358	234.94	174.213	213.80	175.921	214.23	173.854	219.33	173.994	224.02	174.738	217.03
15	8	90	211.981	262.65	215.953	239.07	218.648	240.92	212.802	236.10	214.749	250.97	217.085	242.29
16	8	100	258.503	277.05	259.787	271.70	261.025	256.70	259.102	278.46	259.343	270.87	260.519	287.89
17	8	110	312.743	286.45	316.391	290.53	1208.910	311.87	314.517	295.78	315.178	290.93	318.260	299.18
18	8	120	371.247	301.09	377.948	320.11	1421.538	324.87	375.847	313.37	377.461	316.16	380.289	334.35
19	8	130	438.150	347.27	454.383	335.49	1717.813	345.68	441.690	328.71	443.908	332.55	459.769	360.56
20	8	140	506.418	349.83	526.677	367.76	1973.577	386.92	514.529	362.00	515.820	372.96	532.194	392.73
21	10	50	52.758	142.34	53.382	144.74	53.801	155.88	53.027	145.59	53.234	151.30	53.792	162.89
22	10	60	69.956	169.44	70.834	181.61	72.185	179.50	70.701	178.00	70.782	186.83	70.994	189.32
23	10	70	90.476	184.26	92.991	196.70	94.720	205.67	91.589	200.23	92.842	204.76	93.792	201.55
24	10	80	118.249	219.86	119.378	235.64	121.385	212.13	119.112	223.71	119.211	234.04	120.207	226.40
25	10	90	144.192	242.82	147.905	240.33	153.374	248.69	145.578	263.93	145.877	246.28	150.324	245.77
26	10	100	176.635	265.24	178.789	273.73	184.012	269.00	177.859	273.33	178.283	277.09	180.159	270.71
27	10	110	209.148	277.77	217.357	317.26	673.880	311.26	211.041	294.11	214.155	291.15	223.438	280.26
28	10	120	251.723	304.73	260.021	323.68	800.073	301.31	258.756	322.65	259.067	342.88	268.215	312.73
29	10	130	293.604	333.93	312.095	337.74	1190.785	368.04	303.618	344.04	311.675	356.23	330.873	354.48
30	10	140	344.095	362.53	375.069	345.79	1343.195	398.12	354.818	373.80	367.202	392.46	387.441	383.63

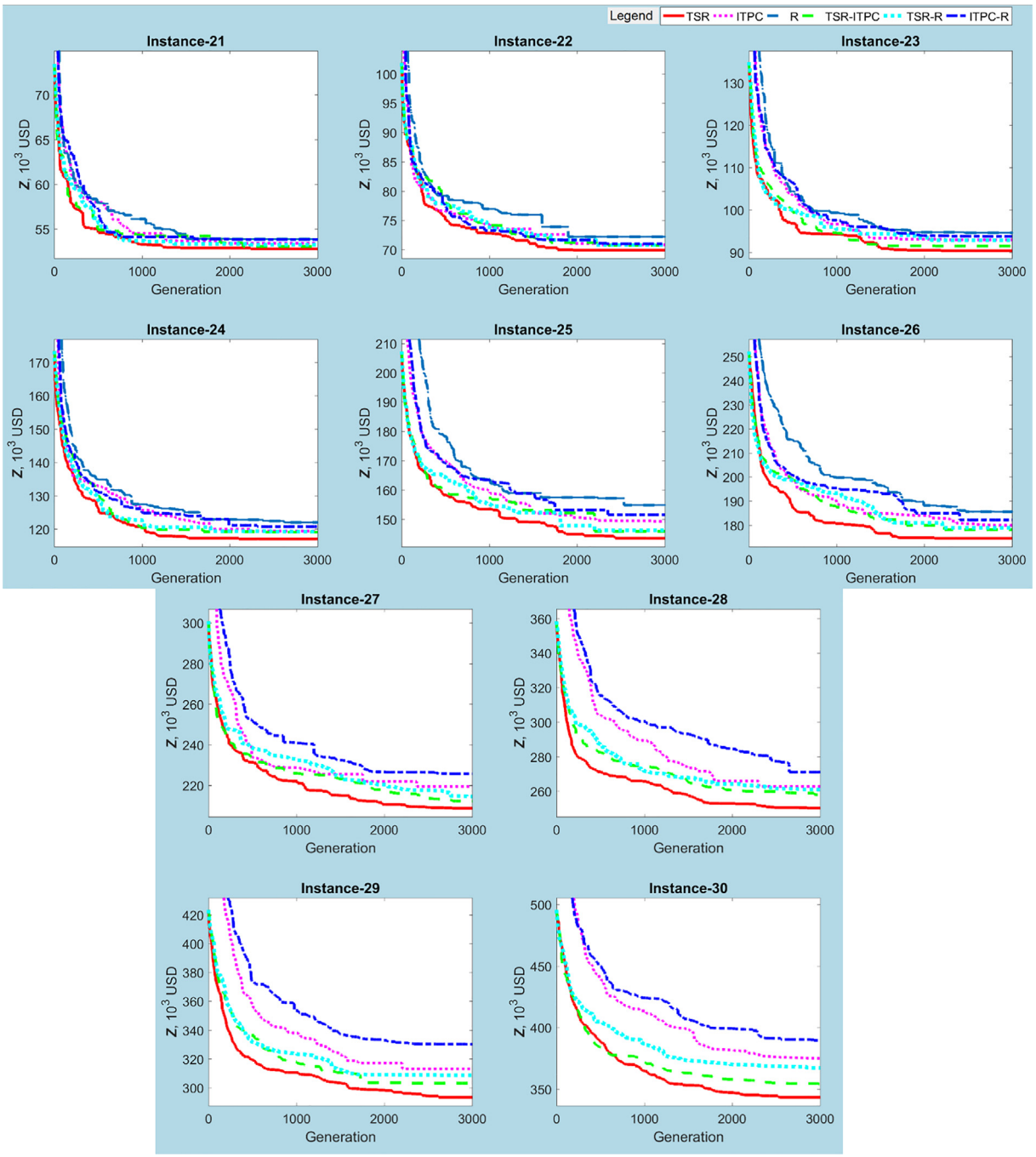


Fig. 10. The recorded convergence diagrams for APMA with different population initialization mechanisms [problem instances 21 through 30].

objective function improvements of up to 9.00%, 3.41%, 6.72%, and 12.69% over the ITPC, TSR-ITPC, TSR-R, and ITPC-R population initialization mechanisms, respectively (see Table 4). The superiority of TSR over ITPC can be explained by the fact that TSR assigns the arriving trucks to the available doors based on the order of their arrival at the CDT and ensures that each outbound truck will be assigned for service after the last inbound truck delivering the products for that outbound truck. On the other hand, ITPC simply assigns the arriving inbound trucks for service first before assigning any outbound trucks to ensure that the truck service precedence constraints will not be violated, which may substantially increase the total waiting time of outbound trucks. The convergence diagrams show that the APMA variations that relied on the TSR population initialization mechanism (i.e., TSR, TSR-ITPC, and TSR-R) began the search process with the initial solutions, which typically have superior values of the objective function compared to the alternative APMA variations (i.e., ITPC, R, and ITPC-R).

Furthermore, APMA with the R population initialization mechanism was not able to identify any feasible solutions throughout the search process and returned the solutions with infeasible truck-to-door assignments at convergence for some of the large problem instances – i.e., problem instances 17 through 20 and problem instances 27 through 30 (see Table 4 and Fig. 10). Therefore, it can be concluded that random population initialization mechanisms demonstrate poor performance for the CDT truck scheduling problems that have the truck service precedence constraints (e.g., as the **IOTSP** mathematical model that was adopted in this study), which highlights the need for problem-specific heuristics at the population initialization stage. As for the required computational efforts, the average computational time values were 190.92 sec, 197.04 sec, 199.96 sec, 197.92 sec, 201.70 sec, and 200.75 sec for the APMA variations with the TSR, ITPC, R, TSR-ITPC, TSR-R, and ITPC-R population initialization mechanisms, respectively. Hence, the changes in the population initialization mechanism did not cause any substantial fluctuations with regards to the APMA computational time.

6.3.2. Evaluation of the effects of polyploidy and hybridization

As a part of the computational experiments, advantages of applying the adaptive polyploid mechanism and hybridization techniques were evaluated by means of comparing the developed APMA algorithm against the alternative EA-based algorithms. In particular, the following EA-based algorithms were considered throughout the analysis: (i) APMA – an EA that relies on the adaptive polyploid mechanism throughout the crossover operations and deploys the OTSI heuristic (i.e., periodically solves the **OTSP** mathematical model to global optimality – see Section 5.7.3 of the manuscript for more details); (ii) APEA – an EA that relies on the adaptive polyploid mechanism throughout the crossover operations without deploying the OTSI heuristic; (iii) MA – an EA that relies on the haploidy concept throughout the crossover operations and deploys the OTSI heuristic; and (iv) EA – an EA that relies on the haploidy concept throughout the crossover operations without deploying the OTSI heuristic.

The APMA, APEA, MA, and EA algorithms were evaluated for all the generated 30 problem instances (see Section 6.1 of the manuscript for the description of input data and problem instances). Throughout the analysis, each EA-based algorithm was launched 10 times in order to compute the average values of computational time and objective function. Table 5 presents the results of performed analysis for the considered EA-based algorithms and provides the following data: (i) the problem instance number; (ii) the total amount of CDT doors available; (iii) the total amount of arriving inbound trucks and outbound trucks; (iv) the objective function values (average over 10 replications); and (v) the computational time values (average over 10 replications). Furthermore, Fig. 11 shows the convergence diagrams, which were recorded for the last replication of each EA-based algorithm. Note that the convergence diagrams are provided for the problem instances with the largest number of

Table 5
The objective function values and computational time required for APMA, APEA, MA, and EA.

Instance	#Doors	#Trucks	APMA		APEA		MA		EA	
			Z, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec
1	2	8	15.359	58.12	15.359	38.42	15.359	53.80	15.359	35.10
2	2	10	21.748	63.63	21.748	44.02	21.748	58.36	21.748	39.01
3	2	12	26.304	66.11	26.304	51.11	26.304	62.23	26.304	43.94
4	2	14	34.817	72.20	34.817	55.05	34.817	67.72	34.817	48.18
5	2	16	46.577	76.15	46.577	57.50	46.577	70.37	46.687	51.73
6	4	8	8.229	61.54	8.229	40.92	8.229	54.81	8.229	36.03
7	4	10	11.417	65.65	11.417	42.54	11.417	58.55	11.417	41.31
8	4	12	13.747	68.54	13.747	46.81	13.747	63.81	13.747	44.14
9	4	14	17.810	67.49	17.810	53.57	17.854	67.21	17.897	48.04
10	4	16	23.167	73.07	23.189	53.96	23.256	71.76	23.295	52.33
11	8	50	73.464	142.83	73.814	131.15	73.877	135.69	73.921	124.37
12	8	60	100.123	158.46	100.911	150.54	101.212	153.02	101.983	149.81
13	8	70	130.386	191.53	131.956	175.62	131.994	178.77	134.257	167.20
14	8	80	172.358	234.94	174.435	196.90	174.724	199.35	176.917	186.37
15	8	90	211.981	262.65	216.429	216.27	216.700	232.49	221.472	204.72
16	8	100	258.503	277.05	266.773	236.96	267.629	265.13	274.643	231.06
17	8	110	312.743	286.45	323.983	261.49	325.497	272.13	331.857	257.64
18	8	120	371.247	301.09	385.294	286.04	393.655	286.04	399.483	286.04
19	8	130	438.150	347.27	454.799	323.26	465.448	325.80	478.424	321.03
20	8	140	506.418	349.83	535.662	343.87	545.461	346.18	557.551	332.34
21	10	50	52.758	142.34	52.815	131.39	53.665	136.69	54.051	126.61
22	10	60	69.956	169.44	70.822	154.95	71.103	160.97	72.281	136.97
23	10	70	90.476	184.26	92.660	174.58	93.338	175.05	95.755	157.24
24	10	80	118.249	219.86	121.352	197.09	122.238	207.33	125.284	178.31
25	10	90	144.192	242.82	149.680	215.98	153.940	230.68	160.304	199.45
26	10	100	176.635	265.24	184.386	239.54	188.877	253.69	197.870	222.35
27	10	110	209.148	277.77	225.326	259.03	229.936	263.88	233.193	245.03
28	10	120	251.723	304.73	273.991	280.68	279.830	303.11	283.838	266.56
29	10	130	293.604	333.93	326.018	315.39	332.366	331.63	340.383	288.24
30	10	140	344.095	362.53	385.375	350.91	395.580	354.57	402.456	331.30

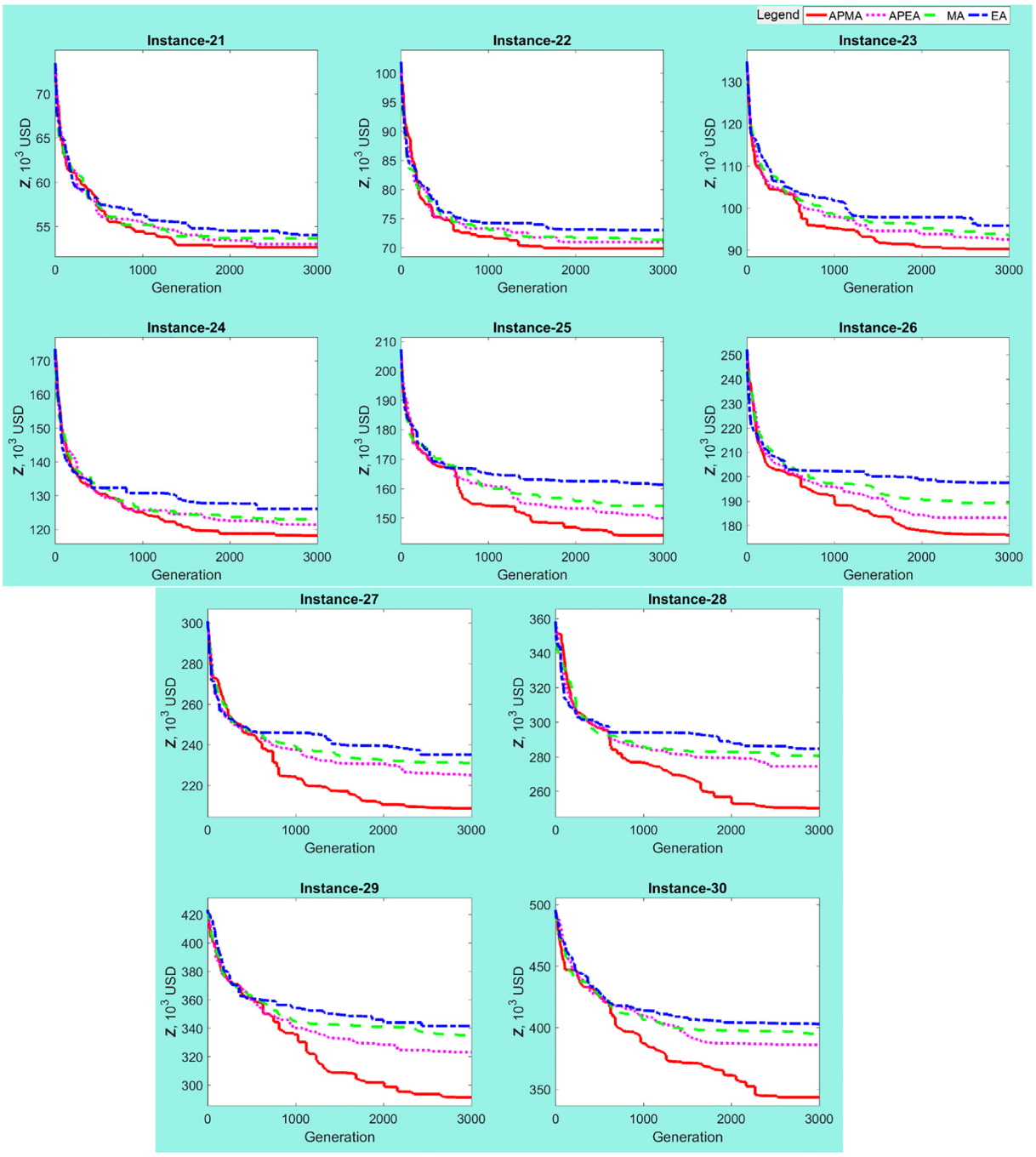


Fig. 11. The recorded convergence diagrams for APMA, APEA, MA, and EA [problem instances 21 through 30].

CDT doors available and the largest number of arriving trucks (i.e., problem instances 21 through 30). However, similar tendencies for the remaining problem instances were noticed.

The conducted analysis shows that the APMA algorithm that relies on the adaptive polyploid mechanism throughout the crossover operations and deploys the OTSI heuristic clearly outperformed the APEA, MA, and EA algorithms. In particular, the APMA algorithm demonstrated objective function improvements of up to 12.00%, 14.96%, and 16.96% over the APEA, MA, and EA algorithms, respectively (see Table 5). Therefore, the application of the adaptive polyploid mechanism and hybridization techniques were favorable for the search process and allowed the identification of superior solutions. The convergence diagrams show that the APMA, APEA, MA, and EA algorithms began the search process with the same initial solutions, since the TSR population initialization mechanism was used for each algorithm. However, after ≈600 ÷ 700 generations APMA started moving more effectively along the search space, which further allowed the identification of good-quality domains of the

search space as well as superior solutions (see Fig. 11). Hence, the adaptive polyploidy and hybridization techniques facilitated the APMA explorative and exploitative capabilities.

Furthermore, based on the performed analysis, it can be stated that the application of adaptive polyploidy alone may not be sufficient for decision problems of high computational complexity (e.g., as the **IOTSP** mathematical model that was adopted in this study), and the hybridization techniques that directly consider problem-specific properties are required to improve the search process. Such a finding can be supported by the fact that APMA consistently outperformed APEA for the majority of the generated 30 problem instances (the same values of objective function were observed only for the small-size problem instances 1 through 9 with up to 4 CDT doors and 14 arriving trucks). As for the required computational efforts, the average computational time values were 190.92 sec, 170.85 sec, 181.36 sec, and 161.75 sec for the APMA, APEA, MA, and EA algorithms, respectively. Hence, the application of the adaptive polyploid mechanism and hybridization techniques increased the computational time. Nevertheless, the APMA computational time did not surpass ≈6 min even for the large problem instances with up to 10 CDT doors and 140 arriving trucks. Such a performance can be considered as satisfactory from the practical perspective.

6.3.3. Comparative analysis against the alternative solution approaches

As a part of the computational experiments, the APMA algorithm was compared against the alternative solution approaches, including the following: (i) CPLEX; (ii) ACO; (iii) TS; (iv) VNS; and (v) SA. All the considered solution approaches were evaluated for all the generated 30 problem instances (see Section 6.1 of the manuscript for the description of input data and problem instances). Throughout the analysis, each solution approach was launched 10 times in order to compute the average values of computational time and objective function. The maximum allowable computational time was set to 7,200 sec for CPLEX, while the target optimality gap was set to 0.10%. Table 6 presents the analysis results for the considered solution approaches and provides the following data: (i) the problem instance number; (ii) the total amount of CDT doors available; (iii) the total amount of arriving inbound trucks and outbound trucks; (iv) the objective function values (average over 10 replications); and (v) the computational time values (average over 10 replications).

The conducted analysis shows that the performance of the adopted exact optimization approach (CPLEX) clearly declines with increasing problem size. In particular, only the small-size problem instances 1 through 8 with up to 4 CDT doors and 12 arriving trucks could be solved within the imposed computational time limit to global optimality. Such a finding highlights high computational complexity of the proposed **IOTSP** mathematical model and underlines the necessity for the deployment of effective metaheuristic or/and heuristic algorithms in order to handle large problem instances in a timely manner. The considered metaheuristics returned either the same or near-optimal solutions when comparing to CPLEX for the small-size problem instances 1 through 8, which confirms their accuracy. Moreover, the APMA algorithm clearly outperformed the ACO, TS, VNS, and SA algorithms. More specifically, the APMA algorithm demonstrated the average objective function improvements of 8.06%, 13.84%, 10.88%, and 17.11% over the ACO, TS, VNS, and SA algorithms, respectively, for the large-size problem instances 11 through 30 (see Table 6). The superiority of APMA demonstrates the effectiveness of the developed adaptive polyploid mechanism and the OTSI heuristic, which was deployed throughout the memetic operations.

A weaker performance of the ACO, TS, VNS, and SA algorithms can be justified by the fact that they are stochastic search algorithms that do not deploy any operators, which account for problem-specific properties. Lack of problem-specific hybridization techniques negatively affects the search process of the solution methods that mainly rely on stochastic search operators, especially for decision problems of high computational complexity (e.g., as the **IOTSP** mathematical model that was adopted in this study). Furthermore, ACO was typically superior with regards to the values of objective function at convergence compared to TS, VNS, and SA, since ACO is a population-based metaheuristic that performs the search process using a population of solutions. On the contrary, TS, VNS, and SA are classified as single-solution-based metaheuristics that perform the search process using just a single solution and its neighbor(s).

Throughout the computational experiments, a comprehensive analysis of the solutions returned by the considered solution algorithms with regards to the objective function components was performed as well. Fig. 12 shows the average values of objective function components for the APMA, ACO, TS, VNS, and SA truck schedules over the problem instances generated, which include the following: (i) the average total waiting time of trucks – $\boldsymbol{ATWT} = \sum_{t\in T}(\boldsymbol{\tau}_t^{WT})/30$; (ii) the average total handling time of trucks – $\boldsymbol{ATHT} = \sum_{t\in T}\sum_{d\in D}(\tau_{td}^{HT}\boldsymbol{x}_{td})/30$; (iii) the average total storage time of products – $\boldsymbol{ATPST} = \sum_{t\in T}(\boldsymbol{\tau}_t^{PST})/30$; (iv) the average total early departure time of trucks – $\boldsymbol{ATET} = \sum_{t\in T}(\boldsymbol{\tau}_t^{ET})/30$; and (v) the average total delayed departure time of trucks – $\boldsymbol{ATDT} = \sum_{t\in T}(\boldsymbol{\tau}_t^{DT})/30$. It can be observed that APMA yielded a substantial reduction in the total waiting time of trucks, total handling time of trucks, total storage time of products, total early departure time of trucks, and total delayed departure time of trucks over the alternative solution algorithms with the average time savings of up to 18.40%, 5.78%, 52.76%, 62.82%, and 19.01%, respectively.

As for the required computational efforts, the average computational time values were 190.92 sec, 147.46 sec, 103.71 sec, 108.77 sec, and 33.31 sec for the APMA, ACO, TS, VNS, and SA algorithms, respectively. Hence, the application of the adaptive polyploid mechanism and hybridization techniques increased the computational time. However, as it was pointed out under Section 6.3.2 of the manuscript, the APMA computational time did not surpass ≈6 min even for the large problem instances with up to 10 CDT doors and 140 arriving trucks. Such a performance can be considered as satisfactory from the practical perspective.

Since the APMA, ACO, TS, VNS, and SA algorithms are stochastic in their nature due to the application of probabilistic operators (e.g., crossover operator, mutation operator), the values of objective function at convergence may differ from

Table 6
The objective function values and computational time required for the solution approaches considered.

Instance	#Doors	#Trucks	CPLEX		APMA		ACO		TS		VNS		SA	
			Z, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec	**Z**, 10^3 USD	CPU, sec
1	2	8	15.359	0.39	15.359	58.12	15.359	34.88	15.359	21.93	15.359	24.73	15.359	6.14
2	2	10	21.748	0.63	21.748	63.63	21.748	38.65	21.748	24.41	21.748	26.94	21.748	7.13
3	2	12	26.304	2.00	26.304	66.11	26.304	42.89	26.304	27.13	26.304	28.73	26.304	8.05
4	2	14	34.817	7.81	34.817	72.20	34.817	46.45	34.817	30.18	34.817	31.24	34.817	8.89
5	2	16	46.577	41.40	46.577	76.15	46.577	50.56	46.898	32.87	46.677	33.92	46.957	9.78
6	4	8	8.229	4.77	8.229	61.54	8.229	36.00	8.229	22.79	8.229	24.39	8.229	6.58
7	4	10	11.417	94.81	11.417	65.65	11.417	40.08	11.417	25.78	11.417	27.41	11.417	7.55
8	4	12	13.747	2671.93	13.747	68.54	13.747	44.93	13.747	28.15	13.747	29.91	13.747	8.12
9	4	14	19.340	7201.65	17.810	67.49	17.810	48.32	17.960	30.72	17.880	31.62	17.990	9.20
10	4	16	25.183	7201.56	23.167	73.07	23.397	49.45	23.717	33.22	23.567	35.00	23.847	10.24
11	8	50	95.631	7202.96	73.464	142.83	74.198	113.99	76.351	79.36	75.088	80.38	76.365	23.72
12	8	60	134.035	7200.66	100.123	158.46	102.394	136.02	104.359	92.50	102.454	95.94	104.429	28.77
13	8	70	182.989	7200.78	130.386	191.53	135.396	155.27	140.277	108.84	138.981	112.83	142.789	33.67
14	8	80	254.767	7200.94	172.358	234.94	180.977	174.38	182.526	120.41	181.581	131.46	189.874	38.45
15	8	90	357.001	7201.23	211.981	262.65	225.764	185.86	230.851	137.31	227.721	145.37	236.115	42.59
16	8	100	579.944	7201.55	258.503	277.05	280.828	206.24	287.581	150.42	286.086	158.96	292.107	46.71
17	8	110	949.441	7201.64	312.743	286.45	337.260	226.18	353.031	163.57	345.931	174.69	354.553	52.46
18	8	120	1100.148	7201.98	371.247	301.09	400.555	248.28	425.693	177.03	413.586	190.79	427.919	58.45
19	8	130	1632.189	7202.21	438.150	347.27	473.572	266.33	510.650	193.80	490.431	201.75	529.787	63.98
20	8	140	1619.329	7201.23	506.418	349.83	558.192	287.88	598.599	214.28	577.251	214.63	607.715	66.93
21	10	50	61.946	7200.62	52.758	142.34	53.382	110.71	57.065	79.91	55.559	80.39	57.381	23.70
22	10	60	88.176	7201.05	69.956	169.44	72.957	129.56	76.783	91.30	73.473	94.88	78.515	29.29
23	10	70	118.825	7201.05	90.476	184.26	95.381	148.44	101.259	108.48	100.982	110.53	106.477	33.95
24	10	80	191.477	7201.10	118.249	219.86	127.767	169.04	130.135	122.71	128.837	123.69	142.972	38.82
25	10	90	265.134	7201.38	144.192	242.82	161.540	187.44	166.276	135.56	164.999	138.68	177.896	44.47
26	10	100	318.584	7201.63	176.635	265.24	199.446	207.63	205.751	141.97	199.731	152.03	216.057	48.98
27	10	110	669.405	7202.09	209.148	277.77	236.887	228.25	257.316	157.23	244.695	167.10	258.858	54.36
28	10	120	757.213	7202.44	251.723	304.73	283.737	249.31	309.408	172.25	294.970	183.75	320.648	60.20
29	10	130	1126.973	7202.76	293.604	333.93	339.496	270.48	376.595	185.97	356.359	197.16	386.729	63.61
30	10	140	1195.247	7203.22	344.095	362.53	393.540	290.20	436.209	201.22	416.887	214.18	454.551	64.65

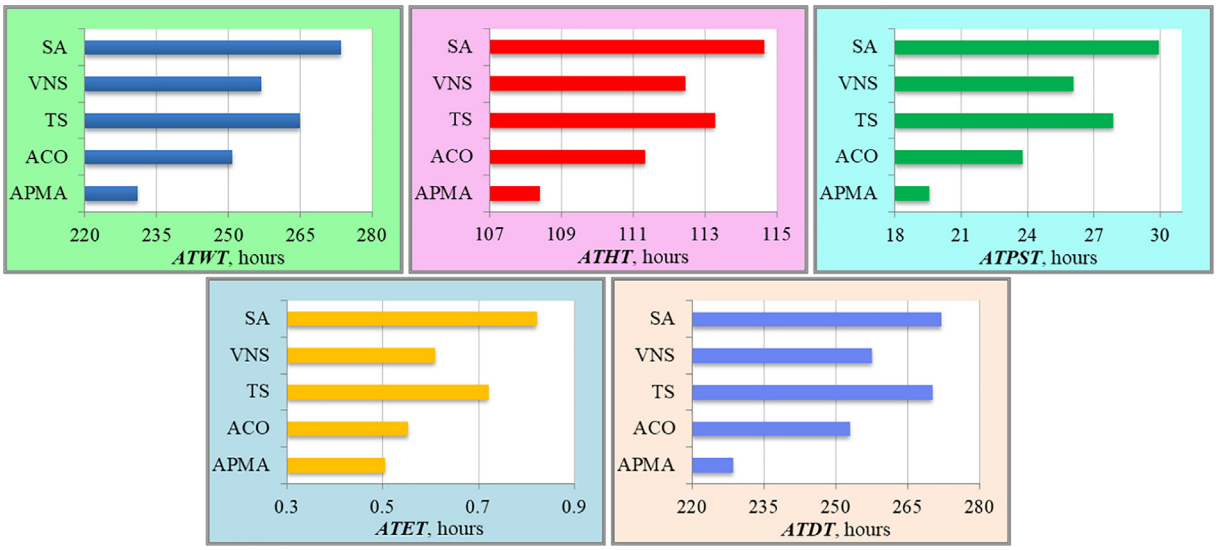


Fig. 12. The average values of objective function components for APMA, ACO, TS, VNS, and SA over the generated problem instances.

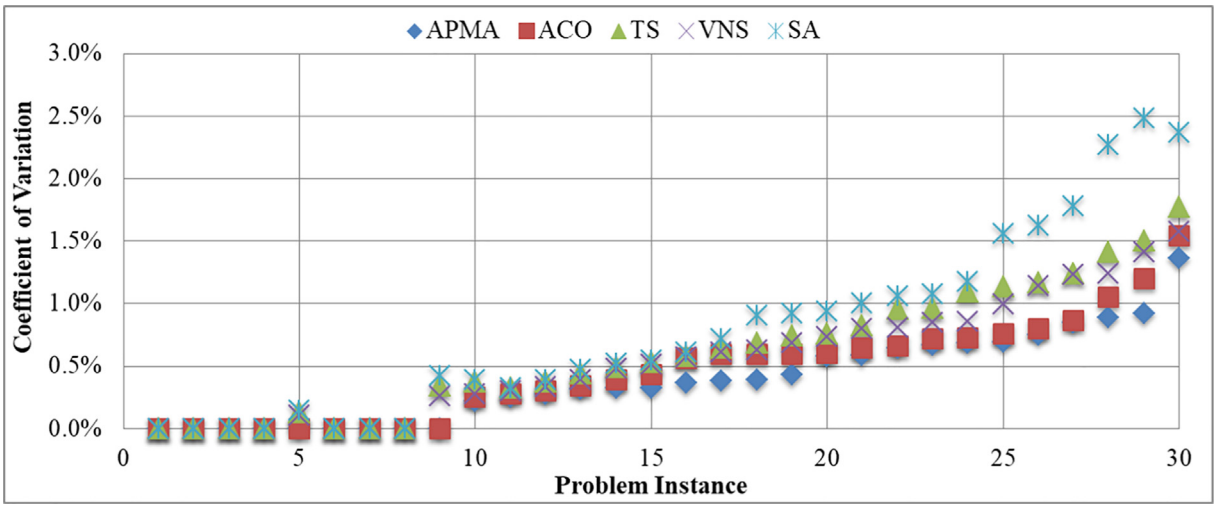


Fig. 13. The coefficient of variation of objective function for APMA, ACO, TS, VNS, and SA [problem instances 1 through 30].

one replication to another. Significant differences in the values of objective function, which are returned by the given solution algorithms at convergence, are not desirable from the practical perspective, as such algorithms cannot be viewed as reliable decision support tools for practitioners (e.g., CDT operators). As a part of the numerical experiments, the coefficient of variation of objective function over 10 replications was computed for each of the considered solution algorithms and all the generated 30 problem instances. Fig. 13 shows the results from conducted analysis, where it can be noticed that the coefficient of variation of objective function did not surpass ≈2.50% for the considered solution algorithms. Hence, it can be concluded that the considered solution algorithms have an acceptable stability level. However, the APMA and ACO algorithms generally demonstrated higher stability levels with the coefficient of variation of objective function not surpassing ≈1.50%. Despite fairly high stability levels of the considered solution algorithms, APMA can be still considered as the most promising solution approach for the **IOTSP** mathematical model, since it substantially outperformed the alternative solution approaches with regards to solution quality.

7. Summary of findings and future research

The cross-docking concept and cross-docking terminals (CDTs) have been extensively used by many supply chain stakeholders, aiming to enhance the effectiveness of various supply chain processes. Cross-docking offers many advantages to CDT

operators, including reduced inventory levels, reduced handling operations, shortened delivery times, enhanced agility of supply chain operations, decreased carbon footprint, just to name a few. However, CDT operators face a wide range of different decision problems throughout planning of operations at their facilities. The truck scheduling problem is considered as one of the most convoluted decision problems at CDTs, where a given CDT operator has to allocate the arriving inbound trucks and outbound trucks among the CDT doors available, set the appropriate order of service for the trucks at each CDT door, as well as determine the start and finish service times for each truck. This study proposed a new Adaptive Polyploid Memetic Algorithm (APMA) for the CDT truck scheduling problem. Unlike the polyploid EA-based algorithms that were previously developed for different decision problems, the proposed APMA algorithm adaptively deployed the polyploidy concept based on the objective function improvements achieved and computational time changes. Moreover, a number of problem-specific hybridization techniques were used within the developed algorithm to facilitate the search process.

Computational experiments that were performed as a part of this study indicated that the application of adaptive polyploidy alone may not be sufficient for the considered decision problem, and the hybridization techniques that directly consider problem-specific properties are required in order to improve solution quality at convergence. It was also found that the performance of the adopted exact optimization approach (CPLEX) clearly declined with increasing problem size. Furthermore, the developed APMA algorithm substantially outperformed Ant Colony Optimization, Tabu Search, Variable Neighborhood Search, and Simulated Annealing, which are viewed as the state of the art metaheuristics that have been extensively used for solving challenging decision problems in the CDT operations literature, with regards to the values of objective function at convergence. As for the required computational efforts, the APMA computational time did not surpass $\approx$6 min even for the large problem instances with up to 10 CDT doors and 140 arriving trucks. Such a performance can be considered as satisfactory from the practical perspective. Therefore, the APMA algorithm developed can assist CDT operators with proper operations planning and truck scheduling as well as reduce the associated costs throughout the service of inbound trucks and outbound trucks.

A number of simplifying assumptions were applied within the proposed mathematical formulation for the problem of scheduling CDT trucks. Moreover, additional steps could be undertaken to better assess the computational performance of the APMA algorithm developed and improve its design. Hence, as a part of the future research, the following activities can be conducted: (i) assess the effects of polyploidy and hybridization in multi-objective settings (e.g., when the CDT operator deals with conflicting objectives throughout scheduling of the arriving inbound trucks and outbound trucks); (ii) evaluate the proposed adaptive polyploidy concept against a self-adaptive polyploidy concept (a self-adaptive polyploidy will decrease the number of APMA parameters required to perform the polyploidy adjustments); (iii) consider perishability for certain product types throughout transfer of these products inside the CDT by internal handling equipment; (iv) consider truck arrival time uncertainties that may occur due to inclement weather, traffic congestion, vehicle breakdowns, and other factors; (v) consider truck handling time uncertainties that may occur due to the CDT congestion, lack of the available internal handling equipment, internal handling equipment breakdowns, and other factors; (vi) evaluation of the APMA algorithm for different stopping criteria; (vii) evaluation of the APMA performance under different approaches for handling infeasible individuals; and (viii) assess the computational performance of the APMA algorithm developed based on a comprehensive comparative analysis against some other solution approaches that have been extensively deployed for solving challenging decision problems in the CDT operations literature as well as the other freight terminal operations studies (e.g., Particle Swarm Optimization, Keshtel Algorithm, Stochastic Beam Search, Differential Evolution, Artificial Bee Colony, Stochastic Fractal Search, Grey Wolf Optimizer).

CRediT authorship contribution statement

Maxim A. Dulebenets: Conceptualization, Methodology, Data curation, Visualization, Investigation, Writing - Original draft, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This study has been partially supported by the grant CMMI-1901109 from the National Science Foundation (USA). The findings, opinions, and conclusions, expressed in this study, are those of the author and may not necessarily represent the views of the National Science Foundation. Furthermore, the author would like to express his gratitude to Prof. John J. Garcia and Ms. Gabriella Cerrato from the English Department at Florida State University for their involvement in preparing this article.

References

[1] A.L. Ladier, G. Alpan, Cross-docking operations: current research versus industry practice, Omega 62 (2016) 145–162.
[2] O. Theophilus, M.A. Dulebenets, J. Pasha, O.F. Abioye, M. Kavoosi, Truck scheduling at cross-docking terminals: a follow-up state-of-the-art review, Sustainability 11 (19) (2019) 5245.

[3] Mecalux, 2019. Advantages and disadvantages of cross-docking: when to apply it in your warehouse? Retrieved on 24 February 2020 from https://www.interlakemecalux.com/.
[4] M.A. Dulebenets, A comprehensive evaluation of weak and strong mutation mechanisms in evolutionary algorithms for truck scheduling at cross-docking terminals, IEEE Access 6 (2018) 65635–65650.
[5] M.A. Dulebenets, A diploid evolutionary algorithm for sustainable truck scheduling at a cross-docking facility, Sustainability 10 (5) (2018) 1333.
[6] H. Zhao, C. Zhang, An online-learning-based evolutionary many-objective algorithm, Inform. Sci. 509 (2020) 1–21.
[7] J. Qiao, F. Li, S. Yang, C. Yang, W. Li, K. Gu, An adaptive hybrid evolutionary immune multi-objective algorithm based on uniform distribution selection, Inf. Sci. 512 (2020) 446–470.
[8] Z.Z. Liu, Y. Wang, P.Q. Huang, AnD: A many-objective evolutionary algorithm with angle-based selection and shift-based density estimation, Inf. Sci. 509 (2020) 400–419.
[9] S. Jiang, H. Li, J. Guo, M. Zhong, S. Yang, M. Kaiser, N. Krasnogor, AREA: An adaptive reference-set based evolutionary algorithm for multiobjective optimisation, Inf. Sci. 515 (2020) 365–387.
[10] Y. Su, N. Guo, Y. Tian, X. Zhang, A non-revisiting genetic algorithm based on a novel binary space partition tree, Inf. Sci. 512 (2020) 661–674.
[11] S. Liu, Q. Yu, Q. Lin, K.C. Tan, An adaptive clustering-based evolutionary algorithm for many-objective optimization problems, Inf. Sci. 537 (2020) 261–283.
[12] G. D'Angelo, R. Pilla, C. Tascini, S. Rampone, A proposal for distinguishing between bacterial and viral meningitis using genetic programming and decision trees, Soft. Comput. 23 (22) (2019) 11775–11791.
[13] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer International Publishing, 2015.
[14] H. Bhasin, S. Mehta, On the applicability of diploid genetic algorithms, AI & Soc. 31 (2) (2016) 265–274.
[15] P.B. Castellucci, F.M. Toledo, A.M. Costa, Output maximization container loading problem with time availability constraints, Oper. Res. Perspect. 6 (2019) 100126.
[16] A.M. Fathollahi-Fard, M. Ranjbar-Bourani, N. Cheikhrouhou, M. Hajiaghaei-Keshteli, Novel modifications of social engineering optimizer to solve a truck scheduling problem in a cross-docking system, Comput. Ind. Eng. 137 (2019) 106103.
[17] A. Ardakani, J. Fei, P. Beldar, Truck-to-door sequencing in multi-door cross-docking system with dock repeat truck holding pattern, Int. J. Industrial Eng. Comput. 11 (2020) 201–220.
[18] W. Wisittipanich, T. Irohara, P. Hengmeechai, Truck scheduling problems in the cross docking network, Int. J. Logist. Syst. Manage. 33 (3) (2019) 420–439.
[19] A. Shahmardan, M.S. Sajadieh, Truck scheduling in a multi-door cross-docking center with partial unloading–reinforcement learning-based simulated annealing approaches, Comput. Ind. Eng. 139 (2020) 106134.
[20] O. Guemri, P. Nduwayo, R. Todosijević, S. Hanafi, F. Glover, Probabilistic tabu search for the cross-docking assignment problem, Eur. J. Oper. Res. 277 (3) (2019) 875–885.
[21] H. Wang, B. Alidaee, The multi-floor cross-dock door assignment problem: rising challenges for the new trend in logistics industry, Transp. Res. E: Logist. Transport. Rev. 132 (2019) 30–47.
[22] H. Corsten, F. Becker, H. Salewski, Integrating truck and workforce scheduling in a cross-dock: analysis of different workforce coordination policies, J. Business Econ. (2019) 1–31.
[23] C. Selma, S. Thevenin, N. Mebarki, O. Cardin, D. Tamzalit, D. Thiériot, L. Bruggeman, Heuristics for robots-humans tasks assignment in a containers loading center, IFAC-PapersOnLine 52 (10) (2019) 13–18.
[24] G. Tadumadze, N. Boysen, S. Emde, F. Weidinger, Integrated truck and workforce scheduling to accelerate the unloading of trucks, Eur. J. Oper. Res. 278 (1) (2019) 343–362.
[25] A. Ahkamiraad, Y. Wang, Capacitated and multiple cross-docked vehicle routing problem with pickup, delivery, and time windows, Comput. Ind. Eng. 119 (2018) 76–84.
[26] H. Abad, B. Vahdani, M. Sharifi, F. Etebari, A multi-objective optimization model for split pollution routing problem with controlled indoor activities in cross docking under uncertainty, J. Quality Eng. Prod. Optim. 4 (1) (2019) 99–126.
[27] A. Rahbari, M.M. Nasiri, F. Werner, M. Musavi, F. Jolai, The vehicle routing and scheduling problem with cross-docking for perishable products under uncertainty: two robust bi-objective models, Appl. Math. Model. 70 (2019) 605–625.
[28] Bagley, J.D., 1967. The behavior of adaptive systems which employ genetic and correlation algorithms. Doctoral Dissertation, University of Michigan.
[29] Goldberg, D.E. and Smith, R.E., 1987, October. Nonstationary Function Optimization Using Genetic Algorithms with Dominance and Diploidy. In ICGA (pp. 59-68).
[30] Ryan, C., 1994. The degree of oneness. In Proceedings of the First Online Workshop on Soft Computing (WSC1) (pp. 43-48).
[31] K.P. Ng, K.C. Wong, A new diploid scheme and dominance change mechanism for non-stationary function optimization, in: Proceedings of the 6th international conference on genetic algorithms, 1995, pp. 159–166.
[32] A. Kamrani, W. Rong, R. Gonzalez, A genetic algorithm methodology for data mining and intelligent knowledge acquisition, Comput. Ind. Eng. 40 (4) (2001) 361–377.
[33] Cavill, R., Smith, S. and Terrell, A., 2005, September. The performance of polyploid evolutionary algorithms is improved both by having many chromosomes and by having many copies of each chromosome on symbolic regression problems. In 2005 IEEE Congress on Evolutionary Computation (Vol. 1, pp. 935-941). IEEE.
[34] Elshamy, W., Emara, H.M. and Bahgat, A., 2013. Polyploidy and discontinuous heredity effect on evolutionary multi-objective optimization. arXiv preprint arXiv:1302.7051.
[35] Pop, P., Oliviu, M. and Sabo, C., 2017, June. A hybrid diploid genetic based algorithm for solving the generalized traveling salesman problem. In International Conference on Hybrid Artificial Intelligence Systems (pp. 149-160). Springer, Cham.
[36] P. Pop, O. Matei, C. Pintea, A two-level diploid genetic based algorithm for solving the family traveling salesman problem, in: Proceedings of the Genetic and Evolutionary Computation Conference, 2018, pp. 340–346.
[37] A.S. Uyar, A.E. Harmanci, in: October. Preserving diversity through diploidy and meiosis for improved genetic algorithm performance in dynamic environments, Springer, Berlin, Heidelberg, 2002, pp. 314–323.
[38] S. Uyar, A.E. Harmanci, in: Comparison of domination approaches for diploid binary genetic algorithms, Springer, Berlin, Heidelberg, 2004, pp. 75–80.
[39] A.S. Uyar, A.E. Harmanci, A new population based adaptive domination change mechanism for diploid genetic algorithms in dynamic environments, Soft. Comput. 9 (11) (2005) 803–814.
[40] S. Yang, in: On the design of diploid genetic algorithms for problem optimization in dynamic environments, IEEE, 2006, pp. 1362–1369.
[41] D. Yazdani, B. Nasiri, A. Sepas-Moghaddam, M.R. Meybodi, A novel multi-swarm algorithm for optimization in dynamic environments based on particle swarm optimization, Appl. Soft Comput. 13 (4) (2013) 2144–2158.
[42] B. Shabash, K.C. Wiese, Diploidy in evolutionary algorithms for dynamic optimization problems, Int. J. Intell. Comput. Cybern. 8 (4) (2015) 312–329.
[43] Petrovan, A., Pop-Sitar, P. and Matei, O., 2019, September. Haploid Versus Diploid Genetic Algorithms. A Comparative Study. In International Conference on Hybrid Artificial Intelligence Systems (pp. 193-205). Springer, Cham.
[44] 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 (1) (2013) 212–229.
[45] M.A. Dulebenets, A Delayed Start Parallel Evolutionary Algorithm for just-in-time truck scheduling at a cross-docking facility, Int. J. Prod. Econ. 212 (2019) 236–258.
[46] M.A. Dulebenets, Application of evolutionary computation for berth scheduling at marine container terminals: parameter tuning versus parameter control, IEEE Trans. Intell. Transp. Syst. 19 (1) (2018) 25–37.

[47] M.A. Dulebenets, An Adaptive Island Evolutionary Algorithm for the berth scheduling problem, Memetic Computing 12 (1) (2020) 51–72.
[48] A. Boloori Arabani, S. Fatemi Ghomi, M. Zandieh, Meta-heuristics implementation for scheduling of trucks in a cross-docking system with temporary storage, Expert Syst. Appl. 38 (3) (2011) 1964–1979.
[49] M. Rajabi, M. Shirazi, Truck scheduling in a cross-dock system with multiple doors and uncertainty in availability of trucks, J. Appl. Environ. Biol. Sci. 6 (7) (2016) 101–109.