A flow based formulation and a reinforcement learning based strategic oscillation for cross-dock door assignment

Mingjie Li[a,b], Jin-Kao Hao[b], Qinghua Wu[a,*]

[a]*School of Management, Huazhong University of Science and Technology, No. 1037, Luoyu Road, Wuhan, China*
[b]*LERIA, Université d'Angers, 2 bd Lavoisier, 49045 Angers Cedex 01, France*

Accepted to European Journal of Operational Research, July 2023.

Abstract

Cross-dock door assignment is a critical warehouse optimization problem in supply chain management. It involves assigning incoming trucks to inbound doors and outgoing trucks to outbound doors to minimize the total pallet-handling cost inside the warehouse. This study investigates a flow based formulation and a reinforcement learning based heuristic approach to solve this problem. The flow based formulation relies on the flow of goods. It is significantly smaller than the existing mixed integer programming formulations in the literature. The proposed heuristic algorithm relies on a Q-learning reinforced procedure to guide the search toward promising areas, and a strategic oscillation method to adaptively explore feasible and infeasible search spaces. It also relies on an improved tabu strategy using attributive and explicit memories. The formulation and proposed heuristic algorithm were tested on two sets of benchmark instances widely used in the literature and compared with several state-of-the-art algorithms. The computational results demonstrated the high competitiveness of the proposed methods in solution quality and computation time. In particular, the flow based formulation can optimally solve more and larger instances and produce better lower and upper bounds than the existing mixed integer programming formulations in the literature. The heuristic approach improved the best solutions (new upper bounds) for 43 of the 99 tested instances while matching the other best-known solutions, except in two cases. The key components of the algorithm were analyzed to justify the algorithm design. The code of the proposed algorithm will be publicly available.

Keywords: Heuristics; formulation; cross-docking assignment; reinforcement learning; strategic oscillation.

1. Introduction

The past few decades have witnessed tremendous growth in global trade volumes, which has greatly stimulated the development of the world's logistics industry and promoted technical changes in supply chain management. Cross-docking is a fast-growing trend in supply chain management (Boysen & Fliedner, 2010; Van Belle et al., 2012; Bodnar et al., 2017; Gaudioso et al., 2021). At a cross-docking terminal, which serves as an intermediary node in the distribution network, the commodities of incoming trucks are unloaded, sorted, moved across the terminal, and finally loaded onto outgoing trucks that immediately depart for the next destination in the distribution network. In contrast to traditional warehouses, cross-docking techniques can significantly accelerate the delivery of commodities and efficiently reduce warehousing, transportation, and other costs associated with pallet handling.

*Corresponding author
Email addresses: lmj@hust.edu.cn (Mingjie Li), jin-kao.hao@univ-angers.fr (Jin-Kao Hao), qinghuawu1005@gmail.com (Qinghua Wu)

Many significant decision-making problems arise during the life cycle of cross-docking terminals. These decision problems concern specific cross-dock settings and must be carefully treated to make the transshipment processes as efficient and economical as possible (Boysen & Fliedner, 2010; Van Belle et al., 2012). Boysen & Fliedner (2010) introduced a comprehensive classification of decision problems arising from cross-docking. Some of these problems involve strategic or tactical decisions, such as determining the optimal location of cross-dock terminals and planning the best terminal layout. Other problems address operational decisions and are abundant during daily cross-dock operations, including the assignment of trucks to dock doors, vehicle routing, truck scheduling, resource scheduling inside the terminal, and load packing into trucks. Among these decision problems, the truck scheduling problem has gained increased attention due to its practical significance. Given a set of incoming and outgoing trucks arriving at the yard and a set of dock doors at the terminal, the truck scheduling problem is to decide on the assignment of trucks to doors to keep the intermediate storage inside the terminal as low as possible and ensure on-time deliveries. In particular, considering different organizational and technical implementations in real-world settings and studying from different planning horizons, leads to a variety of possible truck scheduling problem variants in practice (Boysen & Fliedner, 2010).

This study focuses on an important operational decision problem widely explored in the literature, known as the cross-dock door assignment problem (CDAP) (Boysen & Fliedner, 2010; Van Belle et al., 2012; Gelareh et al., 2020). On a daily basis, several incoming trucks (origins) with pallets of commodities arrive at the yard from different sources. They unload pallets of commodities through inbound doors. After the pallets are sorted and organized according to their destination in a terminal staging area, they are transported directly within the cross-dock (using pallet-handling devices, such as forklifts) to outbound doors, where they are loaded onto outgoing trucks (destinations). The CDAP aims to find the best truck-to-door assignment such that the overall cost of transporting pallets from inbound to outbound doors within the cross-dock terminal is minimized while meeting certain constraints (Van Belle et al., 2012). Both the CDAP and the truck scheduling problem involve determining the optimal truck-to-door assignment to minimize the pallet-handling cost inside the cross-dock. However, one significant difference between these two problems is whether the time aspects are taken into account when assigning trucks to doors. The CDAP assigns trucks to doors at a specific moment and assumes that the door capacity is sufficiently large to accommodate all trucks such that each arriving truck can be assigned to a door. In the truck scheduling problem, trucks arrive at the yard at different times, and the truck-to-door assignment is decided over the planning horizon. The CDAP is proved to be NP-hard since it includes the NP-hard generalized assignment problem as a subproblem (Guemri et al., 2019).

The CDAP has gained significant attention in recent years due to its practical applications at the operational level of supply chain management. Many CDAPs have been defined considering different practical constraints and operational environments (Boysen & Fliedner, 2010; Van Belle et al., 2012). For example, different assignment restrictions are available in the literature, such as each door being able to serve only one truck (Tsui & Chang, 1990, 1992; Tarhini et al., 2016) or each door having the capacity to serve more than one truck (Zhu et al., 2009). Different capacity constraints have also been considered, such as those imposed only on outbound doors (Tsui & Chang, 1990, 1992) or on both inbound and outbound doors (Gelareh et al., 2020). Moreover, different shapes of cross-docking terminals can also be found in the literature (Bartholdi & Gue, 2004). Among them, the I-shaped layout is one of the most often considered shapes (Tsui & Chang, 1990, 1992; Cohen & Keren, 2009; Gue, 1999; Bartholdi III & Gue, 2000; Oh et al., 2006). An I-shaped cross-dock is rectangular, with receiving doors on one side and outbound doors on the other. Therefore, rectilinear distances can be applied to accurately simulate the distances traversed by forklifts following marked lanes. Other shapes, such as the L-shape, U-shape, T-shape, H-shape, and E-shape, have also been studied in the literature (Bartholdi & Gue, 2004).

This study follows the majority of the works on the CDAP and explores the most studied CDAP model in the literature. Specifically, in the CDAP model considered, each door can serve more than one truck simultaneously, capacity constraints are imposed on both sides of the inbound and outbound doors, and an I-shaped cross-dock is considered. To enrich the approaches for solving this problem, we investigate a flow based formulation and propose a reinforcement learning (RL) based strategic oscillation (SO) approach, denoted as RL-SO.

1.1. Related work

This section briefly reviews some representative solution methods for the CDAP and discusses relevant research related to the proposed RL-SO algorithm.

1.1.1. The CDAP formulations and solution approaches

Zhu et al. (2009) introduced the CDAP model considered in this study. They extended the model presented by Tsui & Chang (1990) by considering the realistic constraints arising in practice. They also established a relationship between the generalized quadratic three dimensional assignment problem (GQ3AP) and the CDAP. They showed that the CDAP could be solved as GQ3AP. However, their formulation is based on a quadratic model, which is less effective when solved using commercial mixed integer programming (MIP) solvers, such as CPLEX. Guignard et al. (2012) developed two heuristics to solve this problem. One is the ad hoc, based on the multistart local search, and the other is the convex hull relaxation (CHR), designed explicitly for the quadratic 0-1 problems with linear constraints. Nassief et al. (2016) proposed an MIP formulation strengthened by some valid inequalities for the CDAP, which is concerned with determining the optimum paths for commodities from origins to destinations via inbound and outbound doors. Due to the many variables in their MIP formulation, they also developed a Lagrangian relaxation heuristic to address large-scale instances of the problem. Guemri et al. (2019) proposed two probabilistic tabu search (PTS1 and PTS2) algorithms that differ in how they construct a list of candidate neighboring solutions and accept new incumbent solutions. The results demonstrated that the proposed probabilistic tabu searches exhibited excellent performance by reporting new upper bounds (UBs) for 53 tested instances. However, these tabu search algorithms generally require long computing times to obtain their reported results. Moreover, the gap between the best and average objective results is relatively large. Gelareh et al. (2020) compared 11 CDAP MIP formulations. They further proved the equivalence of all formulations and identified their integral properties. These formulations use four indexed variables to linearize the quadratic term in the objective function, resulting in many variables and constraints. They also performed an extensive comparative study to detect the best formulation among these 11 formulations on benchmark instances from the literature by applying the CPLEX MIP solver to solve each formulation.

1.1.2. Reinforcement learning with optimization methods for combinatorial optimization

To solve the CDAP effectively, the proposed RL-SO algorithm relies on a combination of RL and optimization methods interacting to improve the search capacity of the optimization method and the learning ability of RL. RL has been successfully applied to solve many combinatorial optimization problems. It can be directly applied to solve combinatorial optimization problems in an end-to-end learning manner by simply training the RL model to produce solutions directly from the given input instances. Bello et al. (2016) explored this approach for the traveling salesman problem. However, as observed by Bengio et al. (2021), this approach can fail when evaluated on unseen problem instances that are too far from those used to train the learning predictor.

In recent years, RL methods have been integrated into optimization method frameworks to solve combinatorial optimization problems more effectively. This integration aims to improve the performance of the optimization methods in solution quality, convergence rate, and robustness. The RL technique has been used to guide the design of different optimization method elements, including parameter setting, operator selection, initialization, cutting, and branding. For example, Benlic et al. (2017) used the RL technique to determine the values of several important parameters in their proposed hybrid breakout local search method. Zheng et al. (2021) incorporated the RL technique to select appropriate perturbation operators during the search process of their iterated greedy algorithm. Zhou et al. (2016), Cai et al. (2019), and Gu et al. (2022) used RL to generate good initial solutions for local searches or diversified initial solutions for population based algorithms using the knowledge of previously visited solutions. Sghir et al. (2015) applied RL to schedule several search operators (local search, crossover, and perturbation operators) under a multi-agent based optimization framework. Tang et al. (2020) incorporated a deep RL technique into their cutting plane method to adaptively select cutting planes. Scavuzzo et al. (2022) used RL as a learning tool for branching in MIP. For more information on RL applied to optimization methods for solving combinatorial optimization problems, interested readers can refer to Bengio et al. (2021) and Mazyavkina et al. (2021).

On the other hand, optimization methods can also be integrated into the RL framework to accelerate the convergence of the learning process. For example, Bertsekas (2012), and Peng et al. (2021) introduced dynamic programming methods to update the learning policy, which accelerates the learning efficiency of the RL process. Ojha et al. (2017) introduced evolutionary algorithms and swarm intelligence for RL model training. Despite the effectiveness and potential demonstrated by integrating optimization methods into RL, research on this integration remains limited compared to the service of RL for optimization methods.

1.1.3. Strategic oscillation for strongly constrained optimization problems

The proposed RL-SO approach relies on an SO strategy that combines feasible and infeasible searches to effectively explore the search space of the CDAP. This differs from all existing CDAP local search methods, which are limited to examining only the feasible search space.

The SO allows an algorithm to surpass the boundaries of a feasible search space to visit infeasible solutions (Glover & Hao, 2011). This approach is highly effective for combinatorial optimization problems with strong constraints. As demonstrated in many studies (Glover & Hao, 2011; Qin et al., 2016), restricting the search process only to the feasible region for such constrained problems can easily block the algorithm because the problem constraints can make the feasible search space largely disconnected. Constraint relaxation is a suitable strategy in such situations. By tunneling through feasible and infeasible search regions, the search method can locate high-quality solutions that are difficult to obtain if the search is limited to only feasible regions. For example, Qin et al. (2016) proposed an oscillation based tabu search procedure to solve the quadratic multiple knapsack problem, which allows the search process to visit infeasible solutions by relaxing the knapsack capacity constraint. Li et al. (2022) proposed a local search procedure combining feasible and infeasible searches for the airport gate assignment problem in which the gate conflict constraint is relaxed by temporarily assigning aircraft with overlapping time to the same gate. Martin-Santamaria et al. (2022) applied the SO method to solve the balanced minimum sum-of-squares clustering problem, in which the cluster size constraints are allowed to be relaxed by increasing each cluster size by a percentage during the search. The SO method has also been applied to solve other optimization problems, such as the α-neighbor p-center problem (Sánchez-Oro et al., 2022) and the capacitated hub location problems with modular links (Corberán et al., 2016).

Different oscillation strategies have been proposed to maintain a balanced search between the feasible and infeasible search regions for the algorithm to explore the search space more effectively. In the study by Qin et al. (2016), the infeasible search procedure only considers moves that reallocate objects from heavy-weight knapsacks to light-weight knapsacks to keep the search close to the boundaries of the feasible search regions. Thus, the capacity constraint of each knapsack is only slightly violated, and more solutions are generated near the boundaries of the feasible search regions. Chen et al. (2016) designed a penalty based evaluation function considering the quality of the solution and violation of the capacity constraint to ensure a controlled exploration of infeasible solutions. Lu et al. (2018) adopted an adaptive search strategy in which the algorithm is forced back to the feasible search space if too many infeasible solutions are visited consecutively. By contrast, the search is encouraged to return to the infeasible region if many feasible solutions are visited consecutively.

1.1.4. Different tabu strategies under the tabu search framework

The proposed RL-SO algorithm uses a mixed tabu strategy that takes advantage of the attribute-based and solution-based tabu strategies. This mixed tabu strategy differs from all current tabu search procedures, which use only a single tabu strategy (i.e., either the attribute-based or solution-based tabu strategy) to avoid previously visited solutions.

Tabu search (Laguna, 2018) is a metaheuristic based on the idea of using a "tabu list" to keep track of previously visited solutions to prevent the algorithm from getting stuck in local optima. Tabu search has demonstrated its high efficiency in solving many combinatorial optimization problems. Several advanced versions of tabu search have also been developed, such as the reactive tabu search (Battiti & Tecchiolli, 1994), the guided tabu search (Zachariadis et al., 2009), and the focal distance tabu search (Glover & Lü, 2021).

Though different versions of tabu search can be found in the literature, the tabu strategies for implementing the "tabu list" to keep track of previously visited solutions primarily fall into the following two categories under the tabu search framework: (1) The attribute-based tabu strategy (Glover, 1997), which effectively avoids short-term cycling in the search process by forbidding the reverse move for a certain number of iterations, known as tabu tenure. This tabu strategy has been successfully applied to various combinatorial optimization problems, such as the maximum clique problem (Gendreau et al., 1993), the traveling salesman and routing problems (Gendreau et al., 1994, 1998), the graph coloring problem (Galinier & Hao, 1999), the binary optimization problems (Hanafi et al., 2023), and the project scheduling problem (He et al., 2023); (2) The solution-based tabu strategy (Woodruff & Zemel, 1993; Wang et al., 2017) that utilizes hashing techniques to mark visited solutions and determine the tabu status of new solutions. This tabu strategy can accurately record each visited solution, preventing the search from short- or long-term cycling. Recently, this tabu strategy has demonstrated its effectiveness in solving several combinatorial optimization problems, including the knapsack problems (Lai et al., 2019; Wei & Hao, 2021), the dispersion problems (Wang et al., 2017; Lai et al., 2018; Lu et al., 2023), the obnoxious p-median problem (Chang et al., 2021), and the bipartite Boolean quadratic problems (Wu et al., 2020). Generally, the attribute-based tabu strategy provides more capabilities to diversify the search. In contrast, the solution-based tabu strategy enables a more in-depth exploration of the search space.

1.2. Research gaps and our contributions

As discussed in the previous sections, significant effort has been dedicated to developing different solution methods and mathematical programming formulations to address the CDAP. Nevertheless, more effective solution methods and mathematical programming formulations can still be developed for the CDAP considering the following research gaps: (1) The existing MIP formulations for the CDAP generally involve many variables and constraints, significantly influencing the efficiency of the formulations. The performance of MIP formulations can be enhanced by reducing the number of variables and constraints; (2) Recently, machine learning techniques have been extensively combined with optimization methods to solve various combinatorial optimization problems. This approach has yielded promising results, as evidenced by recent studies (Bengio et al., 2021; Mazyavkina et al., 2021). However, machine learning methods have rarely been used to solve the CDAP currently; (3) The CDAP has some prominent features that can be used to guide the design of search algorithms. By effectively exploring these features, the algorithms' performance can be significantly enhanced. For example, the CDAP can be viewed as a strongly constrained combinatorial optimization problem, particularly when the capacity constraints are tight. As discussed in Section 1.1.3, methods tunneling through feasible and infeasible regions are shown to be particularly effective for such problems. However, these methods have not yet been investigated for solving the CDAP; (4) Tabu search has been shown to be an effective approach for solving the CDAP (Guemri et al., 2019). Currently, the general tabu search framework relies only on a single tabu strategy to prevent the search from cycling (i.e., either the attribute-based or solution-based tabu strategy). No algorithm combining these two tabu strategies into the same local search procedure has been tested for solving combinatorial optimization problems. A better search balance can be expected if these two tabu strategies are suitably combined.

To fill these gaps, this study investigates a flow based MIP (FBMIP) formulation and an original heuristic algorithm combining RL with SO based local search to solve the widely studied CDAP model (Zhu et al., 2009; Guignard et al., 2012; Nassief et al., 2016; Guemri et al., 2019; Gelareh et al., 2020). The aim of this paper is twofold. Given the importance of CDAP, our first goal is to enrich the CDAP literature with a better formulation and a new solution approach, capable of finding higher-quality solutions with fewer computing efforts. The other goal is to investigate some methodological contributions on how reinforcement learning techniques can work with optimization methods together, as well as how to improve the well-known local search methods for combinatorial optimization problems. The detailed contributions of this work are summarized as follows:

- **Methodological contributions:** (1) Based on the idea of the flow of commodities in incoming trucks transported from inbound to outbound doors, this study obtained a flow based formulation using three indexed continuous variables to linearize the quadratic term in the objective. The number of variables

in the resulting formulation is bounded by $O(|\mathcal{M}||\mathcal{I}||\mathcal{J}|)$ ($|\mathcal{M}|$, $|\mathcal{I}|$ and $|\mathcal{J}|$ denote the number of incoming trucks, inbound doors, and outbound doors, respectively), and is much smaller than the number of variables in existing MIP formulations in the literature. (Zhu et al., 2009; Nassief et al., 2016; Gelareh et al., 2020). (2) This study is the first to investigate a method combining machine learning with optimization methods to solve the CDAP. Specifically, our RL-SO algorithm not only uses RL techniques to enhance our local search method but also employs the linear programming (LP) method to improve the learning capabilities of the RL framework. Notably, we propose a warm-start method to initialize the action-value function in the RL framework, which is based on the observation that the LP relaxation of the MIP formulation for the CDAP can effectively estimate the probability value in the action-value function. This initialization enables RL to start from an action-value function with appropriate initial values, significantly reducing the learning time of the RL process. To the best of the authors' knowledge, this study is the first to employ a dedicated optimization method to initialize an action-value function in the context of Q-learning. (3) This study proposes an SO strategy combining feasible and infeasible searches to effectively explore the search space of the CDAP. This strategy is based on a prominent feature of the CDAP that considers the CDAP as a strongly constrained problem imposed by the capacity constraints. By relaxing the capacity constraints in a controlled manner, the SO based local search can oscillate between feasible and infeasible search spaces to effectively locate high-quality solutions. In particular, to prevent the search from moving too far from the boundary of the feasible search space, this study develops an adaptive penalty based fitness function that guides the local search process for a fruitful examination of candidate solutions; (4) This study proposes an improved tabu strategy under the general tabu search framework that can effectively avoid being trapped in local optima or missing high-quality solutions in its explored neighborhood by taking advantage of the attribute-based and solution-based tabu strategies. To the best of the authors' knowledge, this study is the first to explore the possibility of integrating the attribute-based and solution-based tabu strategies into the same local search procedure. Given the generality of the proposed mixed tabu strategy, it can be widely applicable to many other combinatorial optimization problems.

- **Computational contributions:** Extensive experiments were performed to demonstrate the competitiveness of the proposed methods by comparing them with the current best-performing formulations and state-of-the-art algorithms. In particular, this study demonstrates that for the benchmark instances widely used in the literature, the flow based formulation can solve larger instances to optimality and produce better lower bounds (LBs) than existing MIP formulations. In addition, the proposed RL-SO algorithm performs statistically better than state-of-the-art algorithms on the benchmark instances by reaching all but two previous best-known solutions (i.e., UBs) and discovering new UBs for 43 out of 99 tested instances.

The remainder of this paper is organized as follows: Section 2 describes the FBMIP formulation for the CDAP; Section 3 explains the general framework of the RL-SO and its key algorithmic components; Section 4 presents the computational results and compares them with the current best-performing formulations and state-of-the-art algorithms; Section 5 analyzes some of the important characteristics of the proposed RL-SO algorithm; Finally, Section 6 concludes the study and provides directions for future research.

2. Problem statement and mathematical formulation

As discussed in Section 1, many CDAP models have been proposed in the literature by considering different capacity and assignment constraints and various cross-docking terminal shapes (Buijs et al., 2014; Boysen & Fliedner, 2010; Gelareh et al., 2020; Van Belle et al., 2012). Following the mainstream research on the problem, this study focuses on the CDAP model first proposed by Zhu et al. (2009) and widely studied in the literature (Zhu et al., 2009; Guignard et al., 2012; Nassief et al., 2016; Guemri et al., 2019; Gelareh et al., 2020). This model considers an I-shaped cross-dock and assigns each incoming/outgoing truck to an available inbound/outbound door while ensuring the capacity constraint of each inbound/outbound door.

The objective is to minimize the total cost of transporting pallets from inbound to outbound doors within the cross-docking terminal. This model is the basis of many CDAP variants and has been extended to many other similar CDAP models.

Table 1: The sets, parameters and variables used to formulate the CDAP

Notation	Description
Sets:	
$\mathcal{M}$	Set of incoming trucks (origins)
$\mathcal{N}$	Set of outgoing trucks (destinations)
$\mathcal{I}$	Set of inbound doors
$\mathcal{J}$	Set of outbound doors
Parameters:	
C_i	Capacity of an inbound door $i \in \mathcal{I}$
C_j	Capacity of an outbound door $j \in \mathcal{J}$
D_{ij}	Distance between the inbound door $i \in \mathcal{I}$ and the outbound door $j \in \mathcal{J}$
T_{mn}	Number of pallets to be transported from origin $m \in \mathcal{M}$ to destination $n \in \mathcal{N}$
U_m	Total number of pallets delivered from origin $m \in \mathcal{M}$, which equals to $\sum_{n\in\mathcal{N}} T_{mn}$
V_n	Total number of pallets moved to destination $n \in \mathcal{N}$, which equals to $\sum_{m\in\mathcal{M}} T_{mn}$
Variables:	
x_{mi}	A binary decision variable taking the value of 1 if incoming truck $m \in \mathcal{M}$ is assigned to the inbound door $i \in \mathcal{I}$, and 0 if otherwise
y_{nj}	A binary decision variable taking the value of 1 if outgoing truck $n \in \mathcal{N}$ is assigned to the outbound door $j \in \mathcal{J}$, and 0 if otherwise
w_{mij}	A continuous variable represents the pallets of truck $m \in \mathcal{M}$ transporting from inbound door $i \in \mathcal{I}$ to outbound door $j \in \mathcal{J}$

2.1. Nonlinear integer programming formulation

Table 1 summarizes the sets, parameters, and variables used to formulate the CDAP. Based on these defined notations, the CDAP can be formulated using the following mixed integer nonlinear programming (MINP) formulation:

$$\min \quad F = \sum_{m\in\mathcal{M}} \sum_{i\in\mathcal{I}} \sum_{n\in\mathcal{N}} \sum_{j\in\mathcal{J}} D_{ij} T_{mn} x_{mi} y_{nj}, \tag{1}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{I}} x_{mi} = 1, \ \forall m \in \mathcal{M}, \tag{2}$$

$$\sum_{j\in\mathcal{J}} y_{nj} = 1, \ \forall n \in \mathcal{N}, \tag{3}$$

$$\sum_{m\in\mathcal{M}} U_m x_{mi} \le C_i, \ \forall i \in \mathcal{I}, \tag{4}$$

$$\sum_{n\in\mathcal{N}} V_n y_{nj} \le C_j, \ \forall j \in \mathcal{J}, \tag{5}$$

$$x_{mi} \in \{0,1\}, \ \forall m \in \mathcal{M}, \ \forall i \in \mathcal{I}, \tag{6}$$

$$y_{nj} \in \{0,1\}, \ \forall n \in \mathcal{N}, \ \forall j \in \mathcal{J}. \tag{7}$$

The objective (1) minimizes the total pallet-handling cost inside the cross-docking terminal. Constraints (2) and (3) guarantee that each origin/destination is assigned to exactly one inbound/outbound door.

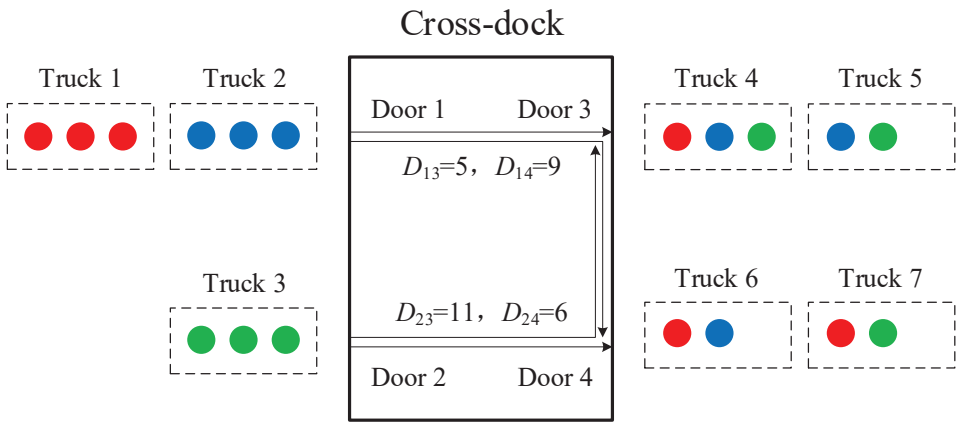

Figure 1: An example of the CDAP.

Constraints (4) and (5) ensure that the capacity of each inbound/outbound door is satisfied. Constraints (6) and (7) specify that the decision variables are binary.

Figure 1 shows an example of the CDAP model considered in this study. Trucks 1, 2, and 3 are incoming trucks with three pallets, whereas trucks 4, 5, 6, and 7 are outgoing trucks. A circle represents a pallet, and pallets of the same color are from the same incoming truck. Truck 1 needs to send 1/0/1/1 pallet to trucks 4/5/6/7, truck 2 needs to send 1/1/1/0 pallet to trucks 4/5/6/7, and truck 3 needs to send 1/1/0/1 pallet to trucks 4/5/6/7. Doors 1 and 2 are inbound, whereas doors 3 and 4 are outbound. Each door has a six-pallet capacity. The distances between each pair of inbound/outbound doors are $D_{13} = 5$, $D_{14} = 9$, $D_{23} = 11$, and $D_{24} = 6$. A solution with a total pallet-handling cost of 70 is obtained, calculated as $F = (T_{14} + T_{24} + T_{25}) \times D_{13} + (T_{16} + T_{17} + T_{26}) \times D_{14} + (T_{34} + T_{35}) \times D_{23} + T_{37} \times D_{24} = 70$ by assigning trucks 1 and 2 to door 1, truck 3 to door 2, trucks 4 and 5 to door 3, and trucks 6 and 7 to door 4.

2.2. Flow based three indexed MIP formulation

Inspired by the formulation of the quadratic assignment problem (Erdoğan & Tansel, 2007), the following flow based MIP formulation is obtained based on the definition of w_{mij} as the pallets of truck m transported from door i to door j:

$$\min \quad F = \sum_{m\in\mathcal{M}} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} D_{ij} w_{mij}, \tag{8}$$

$$\text{s.t.} \quad (2) - (7),$$

$$\sum_{j\in\mathcal{J}} w_{mij} = U_m x_{mi}, \forall m \in \mathcal{M}, i \in \mathcal{I}, \tag{9}$$

$$\sum_{i\in\mathcal{I}} w_{mij} = \sum_{n\in\mathcal{N}} T_{mn} y_{nj}, \forall m \in \mathcal{M}, j \in \mathcal{J}, \tag{10}$$

$$w_{mij} \geq 0, \forall m \in \mathcal{M}, i \in \mathcal{I}, j \in \mathcal{J}. \tag{11}$$

The objective (8) is equivalent to the objective (1). Constraints (9) and (10) are the multi-commodity flow balance equations for the pallets. Constraints (9) ensure that if truck m is assigned to door i, all pallets delivered by truck m must be transported from door i to the other doors. Constraints (10) ensure that all pallets between incoming truck m and other outgoing trucks assigned to door j must travel to door j from the door to which truck m is assigned. Using pallet flow based constraints in the CDAP enabled the formulation of the problem with three indexed variables rather than four indexed variables as all current MIP formulations for the CDAP.

Table 2 compares the number of variables and constraints in the FBMIP model with the $M^{2',0}$ model (Gelareh et al., 2020), the current best-performing MILP formulation in the literature (Appendix A). As shown in Table 2, the FBMIP model has significantly fewer variables and constraints than the $M^{2',0}$. Therefore, the FBMIP is expected to be computationally advantageous compared to the $M^{2',0}$, as demonstrated in Section 4.

Table 2: Number of variables and constraints in the FBMIP and $M^{2',0}$ (Gelareh et al., 2020).

	FBMIP	$M^{2',0}$
#Continuous variables	$\|\mathcal{M}\|\|\mathcal{I}\|\|\mathcal{J}\|$	$\|\mathcal{M}\|\|\mathcal{N}\|\|\mathcal{I}\|\|\mathcal{J}\|$
#Binary variables	$\|\mathcal{M}\|\|\mathcal{I}\| + \|\mathcal{N}\|\|\mathcal{J}\|$	$\|\mathcal{M}\|\|\mathcal{I}\| + \|\mathcal{N}\|\|\mathcal{J}\|$
#Constraints	$\|\mathcal{M}\|\|\mathcal{I}\| + \|\mathcal{M}\|\|\mathcal{J}\| + \|\mathcal{M}\| + \|\mathcal{I}\| + \|\mathcal{N}\| + \|\mathcal{J}\|$	$\|\mathcal{M}\|\|\mathcal{N}\|\|\mathcal{I}\| + \|\mathcal{M}\|\|\mathcal{N}\|\|\mathcal{J}\| + \|\mathcal{M}\| + \|\mathcal{I}\| + \|\mathcal{N}\| + \|\mathcal{J}\|$

Theorem 1. *Let $(\boldsymbol{x}, \boldsymbol{y})$ be a feasible solution to the CDAP with objective value $F_{\text{MINP}}(\boldsymbol{x}, \boldsymbol{y})$. Then, a unique solution $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ exists, such that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ is feasible to* FBMIP *with objective value $F_{\text{FBMIP}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}) = F_{\text{MINP}}(\boldsymbol{x}, \boldsymbol{y})$.*

Proof. The feasibility of $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ can be guaranteed by the assignment constraints (i.e., constraints (2), (3), (6), and (7)) and capacity constraints (i.e., constraints (4), (5), (6), and (7)).

For each $m \in \mathcal{M}$, let e_m present the door i that truck m is assigned to (i.e., $x_{mi} = 1$). For each $i \in \mathcal{I}$, let E_i be the set containing all trucks $m \in \mathcal{M}$ assigned to door i (i.e., $x_{mi} = 1$). Similarly, for each $n \in \mathcal{N}$, let e_n present the door j, where truck n is assigned (i.e., $x_{nj} = 1$). For each $j \in \mathcal{J}$, let E_j be the set containing all trucks $n \in \mathcal{N}$ assigned to door j (i.e., $y_{nj} = 1$).

Given that $x_{mi} = 0, \forall i \neq e_m$, constraints (9) and (11) ensure that $w_{mij} = 0, \forall i \neq e_m, m \in \mathcal{M}$. The left of constraints (10) is w_{me_mj} because all terms except $i = e_m$ are zero. The right of constraints (10) is $\sum_{n \in E_j} T_{mn}$ because all terms except $n \in E_j$ are zero. Therefore, w is uniquely determined by the following equations: $w_{me_mj} = \sum_{n \in E_j} T_{mn}, \forall j \in \mathcal{J}, m \in \mathcal{M}$ and $w_{mij} = 0, \forall i \neq e_m, m \in \mathcal{M}, j \in \mathcal{J}$. Thus, given a solution $(\boldsymbol{x}, \boldsymbol{y})$ to MINP, a solution $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ can be constructed to FBMIP.

Such a constructed solution guarantees that constraints (10) and (11) are satisfied. Then, constraints (9) need to be checked to ensure that they are also satisfied. If $i \neq e_m$, then both sides of constraints (9) are zero. If $i = e_m$, the left side is $\sum_{j \in \mathcal{J}} w_{me_mj}$, and the right side is U_m. Given that $w_{me_mj} = \sum_{n \in E_j} T_{mn}$ by construction, the left side is $\sum_{j \in \mathcal{J}} \sum_{n \in E_j} T_{mn}$. According to constraints (3), every truck $n \in \mathcal{N}$ is assigned to exactly one door $j \in \mathcal{J}$. Hence, $\sum_{j \in \mathcal{J}} \sum_{n \in E_j} T_{mn} = \sum_{n \in \mathcal{N}} T_{mn}$, which equals U_m. All the abovementioned constraints ensure that if $(\boldsymbol{x}, \boldsymbol{y})$ is a feasible solution to MINP, then a unique and feasible solution $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ exists for FBMIP.

$F_{\text{FBMIP}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w}) = F_{\text{MINP}}(\boldsymbol{x}, \boldsymbol{y})$ is proved by observing that every term $D_{ij} T_{mn} x_{mi} y_{nj} = 0$ unless $m \in E_i$ and $n \in E_j$. Therefore, the MINP objective is $\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \sum_{m \in E_i} \sum_{n \in E_j} D_{ij} T_{mn}$. Given that $w_{me_mj} = \sum_{n \in E_j} T_{mn}, \forall j \in \mathcal{J}, m \in \mathcal{M}$ and $w_{mij} = 0, \forall i \neq e_m, m \in \mathcal{M}, j \in \mathcal{J}$, the objective of FBMIP is $\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \sum_{m \in E_i} \sum_{n \in E_j} D_{ij} T_{mn}$, which is the same as that of MINP. □

3. RL-SO for the CDAP

This section presents the proposed RL-SO approach for the CDAP. The algorithm integrates several key components responsible for its performance, including a Q-learning mechanism that guides the algorithm towards promising areas based on the search history, an SO method that explores both feasible and infeasible solution spaces, and a mixed tabu strategy that combines two memory mechanisms to prevent search cycling.

3.1. Main scheme

The RL-SO algorithm integrates the SO approach into a general Q-learning based framework. Algorithm 1 summarizes the general scheme of the RL-SO. The learning functions Q and R are initialized only once

Algorithm 1: Pseudocode of RL-SO for the CDAP

Input: The CDAP instance, time limit t_{max}.
Output: The best solution S^* found so far.

```
1 t_0 ← Time()
2 Q, R ← Linear relaxation formulation based estimation (FBMIP)        Section 3.2.2
3 while Time() − t_0 < t_max do
4     S ← Q-learning reinforced truck-to-door assignment (Q, R)        Section 3.2
5     S_l ← Strategic oscillation (S)                                  Section 3.3
6     R ← Reward updating (S, S_l, R)                                  Section 3.4
7     if f(S_l) < f(S*)) then
8         S* ← S_l
9 return S*
```

at the beginning of the algorithm (Line 2). Subsequently, the algorithm enters its main loop (Lines 3-8) comprising three key components: a Q-learning reinforced truck-to-door assignment procedure (Section 3.2) to generate an initial solution S based on functions Q and R, an SO phase (Section 3.3) to improve the initial solution produced by the Q-learning reinforced truck-to-door assignment procedure, and a reward updating phase (Section 3.4) to update the reward function R by comparing the initial solution S and the improved local optimum solution S_l. The algorithm iterates these three main procedures until a given time limit t_{max} is reached. The best solution S^* found during the search is finally returned as the result of the algorithm. The succeeding subsections describe the three key components of the RL-SO approach in detail.

3.2. Reinforcement learning procedure

Recently, machine learning, particularly RL, has been successfully applied to solve many typical combinatorial optimization problems (Mazyavkina et al., 2021) such as the traveling salesman problem (Khalil et al., 2017; Zheng et al., 2021), the bin packing problem (Zhao et al., 2021; Jiang et al., 2021), the maximum independent set problem (Manchanda et al., 2020), the graph coloring problem (Zhou et al., 2016), and other difficult problems (Khalil et al., 2017; Benlic et al., 2017; Wang et al., 2020; Alicastro et al., 2021; Gu et al., 2022). Building upon this trend, this study proposes a Q-learning reinforced method for constructing high-quality starting solutions for the SO procedure.

3.2.1. Q-learning reinforced truck-to-door assignment

Q-learning is an RL method combining the Monte Carlo algorithm and dynamic programming (Watkins & Dayan, 1992). The relevant notations in the Q-learning framework, such as states, actions, transitions, and rewards, are described below. This study only considers the assignment of trucks in $\mathcal{M}$ to doors in $\mathcal{I}$ to present the Q-learning procedure because the assignment of trucks in $\mathcal{N}$ to doors in $\mathcal{J}$ can be completed similarly.

- **States (ST)**: The Q-learning reinforced method constructs a solution incrementally by assigning a truck $m \in \mathcal{M}$ to a door $i \in \mathcal{I}$. The current state $s_t \in ST$ of the system is the truck just assigned to a door in the partial solution.
- **Actions (AC)**: In a state s_t, an action a_{mi} should be selected according to the transition strategy, extending the partial solution by assigning a truck $m \in \mathcal{M}$ to a door $i \in \mathcal{I}$.
- **Transition**: When action a_{mi} is performed, the algorithm initiates a transfer to a new state s_m because truck m has just been assigned.
- **Transition strategy**: To select an action a_{mi} in a state s_t, the algorithm employs an ε-greedy transition strategy. This strategy selects the action a_{mi} with the largest value of $Q(s_t, a_{mi})$ with a probability of ε, while randomly selecting an action with a probability of $1 - \varepsilon$. The action-value function $Q(s_t, a_{mi})$ is a mapping from states (ST) and actions (AC) into real numbers Q: $ST \times AC \rightarrow Q$. It operates on the matrix Q (Figure 2), where the element $Q_{t,(m-1)|I|+i}$ is an estimation of the action value obtained by performing action a_{mi} in state s_t.

- **Rewards**: The reward function $R(s_t, a_{mi})$ is also a mapping from states (ST) and actions (AC) into real numbers R: $ST \times AC \rightarrow R$ (Figure 2). It operates on a matrix R, where its element $R_{t,(m-1)|I|+i}$ takes a real number indicating the reward for selecting the action a_{mi} in the current state s_t. The reward function R is updated after the constructed solution is improved through the SO method by checking the differences between the starting solution S and the local optimum solution S_l. The detailed updating method is described in Section 3.4.

Based on the above notations, the Q-learning procedure for constructing a solution comprises a sequence of steps $\{1, ..., |M|\}$. In step k, the procedure is in a state s_t and performs an action a_{mi} to assign a truck to a door, selected according to the transition strategy. Subsequently, it goes to a new state $s_{t'}$. Meanwhile, a one-step temporal-difference control (Sutton & Barto, 2018) is applied as follows to update the action-value function $Q(s_t, a_{mi})$ with reward $R(s_t, a_{mi})$:

$$Q(s_t, a_{mi}) = (1-\gamma)Q(s_t, a_{mi}) + \alpha[R(s_t, a_{mi}) + \gamma \max_{a_{m'i'} \in AT} Q(s_{t'}, a_{m'i'})], \tag{12}$$

where s_t is the current state, a_{mi} is the current action, $s_{t'}$ is the next state, $a_{m'i'}$ is the next action, $\gamma \in [0, 1]$ is the discount factor, $\alpha \in [0, 1]$ is the learning rate, $R(s_t, a_{mi})$ is the reward to choose this current action (Section 3.4), and $Q(s_{t'}, a_{m'i'})$ is the action-value for the next state.

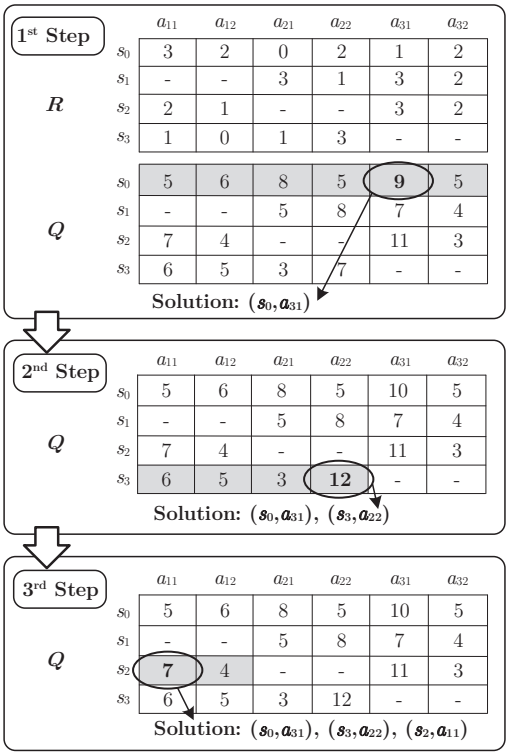

Figure 2: An example of the Q-learning reinforced truck-to-door assignment procedure.

Figure 2 illustrates how the Q-learning procedure works on an instance with three trucks and two doors. In Figure 2, the action-value function $Q(s_t, a_{mi})$ operates on matrix Q, where the rows correspond to states, and the columns correspond to the actions. Moreover, an element $Q_{t,2(m-1)+i}$ is the expected action value of a_{mi} in state s_t. Similarly, the reward function $R(s_t, a_{mi})$ operates on matrix R, where its element $Q_{t,2(m-1)+i}$ is the expected reward value of a_{mi} in state s_t. In the first step, the state is s_0, and no truck

has yet been assigned. One of the six feasible actions marked in gray in the first row can be selected. It is assumed that $\varepsilon = 1$, reducing to a completely greedy strategy. Thus, action a_{31} is selected according to the greedy strategy. Subsequently, the state transits to s_3 because truck 3 is the last assigned truck. The value of $Q(s_0, a_{31})$ is updated according to Equation (12) (with the assumption that $\alpha = 1$ and $\gamma = 0.5$). In the second step, the state is s_3, and action a_{22} is selected to be performed by the transition strategy (in the row of state s_3, only the actions marked in gray can be considered because it is impossible to assign truck m to any door if truck m has already been assigned). The value of $Q(s_3, a_{22})$ is updated accordingly. Subsequently, the procedure proceeds to state s_2 in step 3, where a_{11} is selected, and $Q(s_2, a_{11})$ is updated. After these three steps, each truck is assigned to a door, and the Q-learning procedure terminates.

3.2.2. Linear relaxation formulation based estimation for action-value function

A reasonable initial Q estimation enables the algorithm to generate a good initial solution, thereby rapidly guiding the search toward a promising region. This study is the first to introduce a linear programming relaxation formulation based estimation for initializing Q to obtain a reasonable initial Q. This approach is new in the field of Q-learning and has the potential to enhance the efficiency and effectiveness of Q-learning methods. Specifically, constraints (6) and (7) in the FBMIP formulation are relaxed as follows:

$$0 \leq x_{mi} \leq 1, \ \forall m \in \mathcal{M}, \ \forall i \in \mathcal{I}, \tag{13}$$

$$0 \leq y_{nj} \leq 1, \ \forall n \in \mathcal{N}, \ \forall j \in \mathcal{J}. \tag{14}$$

The relaxed FBMIP formulation can be optimally solved by relaxing the decision variables x and y from binary to continuous variables. The value for $x_{mi}, \forall m \in \mathcal{M}, i \in \mathcal{I}$ can be directly applied to initialize each action value in Q for action a_{mi} in all states s_t (the same technique is also applied to $y_{nj}, \forall n \in \mathcal{N}, j \in \mathcal{J}$ in the Q-learning procedure). The rationale behind this initialization is that a large value of x_{mi} in the optimum solution to the linear relaxation of the CDAP formulation generally implies a high probability that truck m is assigned to door i in the optimum integer solution. Thus, each action a_{mi} comprising the assignment of truck m to door i is associated with a large action value in Q in all states s_t. Each element in R is initially set to be the same as each element in Q at the beginning of the Q-learning procedure.

3.3. Strategic oscillation

The SO method (Glover, 1997) allows for exploration of the infeasible search space to provide more search freedom. As demonstrated in several studies on strongly constrained problems (Liu et al., 2014; Qin et al., 2016; Li et al., 2022), visiting intermediary infeasible solutions during the search process can efficiently enhance the performance of neighborhood based local searches because it may facilitate transitions between structurally different feasible solutions. Following the above idea, the search is allowed to oscillate between feasible and infeasible search spaces by relaxing the capacity constraints. This study designed an extended evaluation function Z combining the objective function of the CDAP (total pallet-handling cost; refer to Section 2) with a penalty function associated with the violation degree of the capacity constraints (Section 3.3.2).

Algorithm 2 summarizes the general scheme of the SO, while the subsequent subsections detail its main components. Starting from an initial feasible or infeasible solution S, each iteration of the SO procedure involves determining the overall best admissible solution (i.e., a solution that is not forbidden by the tabu strategy and has the maximum extended evaluation function value) from the neighborhood of the current solution S' produced by the four move operators (shift, swap, multishift, and multiswap). During the search process, the best-found solution S_l is updated with S' if S' is feasible and is better than S_l in terms of the total pallet-handling cost. If an improvement in terms of the objective function is not achieved during ω consecutive iterations, the best feasible solution S_l found during the search is returned as the final output of the SO procedure.

3.3.1. Neighborhoods and fast evaluation technique

A solution of the CDAP corresponds to a partition of the set of trucks $\mathcal{M} \cup \mathcal{N}$ into $|\mathcal{I} \cup \mathcal{J}|$ disjoint subsets, where E_k $(k \in \mathcal{I} \cup \mathcal{J})$ denotes the set of trucks assigned to door k. E_k $(k \in \mathcal{I})$ and E_k $(k \in \mathcal{J})$

Algorithm 2: Pseudocode of SO

Input: A starting solution S.
Output: The local optimum feasible solution S_l.

$\varphi \leftarrow 0$ /*Initializing the penalty strength*/
$N \leftarrow 0$ /*Initializing the number of the consequent non-improve iterations*/
$O_l \leftarrow +\infty$ /*Objective value of the local optimum feasible solution*/
while $N < \omega$ **do**
  $S' \leftarrow$ The best admissible neighboring solution of S
  if S' is a feasible solution and $Z(S') < O_l$ **then**
    $O_l \leftarrow Z(S')$, $S_l \leftarrow S'$, $S \leftarrow S'$, $N \leftarrow 0$;
  else if S' is an infeasible solution and $Z(S') < Z(S)$ **then**
    $S \leftarrow S'$, $N \leftarrow 0$;
  else
    $S \leftarrow S'$, $N \leftarrow N + 1$;
  if five consecutive solutions are feasible **then**
    $\varphi \leftarrow \varphi - \tau$ /*Reduce penalty strength*/
  else if five consecutive solutions are infeasible **then**
    $\varphi \leftarrow \varphi + \tau$ /*Boost penalty strength*/
  $iter \leftarrow iter + 1$
return S_l

can receive trucks only from $\mathcal{M}$ and $\mathcal{N}$, respectively. Therefore, a solution of the CDAP can be denoted by $S = \{E_k | k \in \mathcal{I} \cup \mathcal{J}\}$.

Local search typically employs move operators to transform a given solution S into a neighboring solution S'. The proposed approach jointly uses four move operators to generate neighboring solutions (denoted by **shift**, **swap**, **multi-shift**, and **multi-swap**). To describe these operators concisely, only the relevant moves on the inbound side are considered while the operations on the outbound side are determined in a similar manner.

- **Shift:** Transfer a truck m from its current door i to another door i'.
- **Swap:** Exchange two trucks m and m' from two different doors e_m and $e_{m'}$.
- **Multi-shift:** Transfer all trucks in E_i from their current door i to another door i'.
- **Multi-swap:** Exchange all trucks between two different doors i and i'.

The method proposed by Guemri et al. (2019) is adopted to ensure a fast evaluation of the move gain values, indicating the change to objective F (Equation (1) in Section 2) brought by these move operators. This method uses efficient data structures to accelerate neighborhood evaluations.

3.3.2. Feasible and infeasible search

The search space Ω explored by the proposed algorithm comprises all feasible and infeasible solutions (an infeasible solution violates capacity constraints). Given an inbound/outbound door $k \in \mathcal{I} \cup \mathcal{J}$, let $W_k (k \in \mathcal{I} \cup \mathcal{J})$ be the total number of pallets allocated to door k:

$$W_k = \begin{cases} \sum_{m \in E_k} U_m & , \text{if } k \in \mathcal{I}, \\ \sum_{n \in E_k} V_n & , \text{if } k \in \mathcal{J}. \end{cases} \tag{15}$$

The penalty function $P(S)$ corresponds to the violation degree of S with respect to the capacity constraints, given by the overall overloaded parts of all the doors to the capacity limit, i.e., $P(S) = \sum_{k \in \mathcal{I} \cup \mathcal{J}} t_k$, where

$$t_k = \begin{cases} W_k - C_k & , \text{if } W_k > C_k, \\ 0 & , \text{if otherwise.} \end{cases} \tag{16}$$

Subsequently, given a candidate solution $S \in \Omega$ in the enlarged search space, its quality is evaluated by an extended evaluation function (fitness) $Z(S)$, which is a linear combination of the objective function $F(S)$ and the penalty function $P(S)$ as follows:

$$Z(S) = F(S) + \varphi P(S), \tag{17}$$

where φ is a parameter used to control the relative importance of the penalty function $P(S)$. Increasing φ decreases the favorability of infeasible solutions, whereas decreasing φ encourages infeasible solutions. In this study, φ is adaptively adjusted according to the search context. In particular, φ is increased by setting $\varphi = \varphi + \tau$ if five consecutively visited solutions are all feasible. By contrast, φ is decreased by setting $\varphi = \varphi - \tau$ if five consecutively visited solutions are all infeasible. The rationale behind this adaptive strategy is to maintain a balance between feasible and infeasible searches. When the search is confined in feasible regions, it is encouraged to move into infeasible regions by reducing φ. Conversely, when too many infeasible solutions are visited, the search is forced to move back to the feasible search space by increasing φ. The move gain of a candidate solution can be rapidly updated as follows:

$$\Delta_Z(mv) = \Delta_F(mv) + \varphi \Delta_P(mv). \tag{18}$$

The move gain $\Delta_F(mv)$ of each move mv relative to the objective function F can be efficiently calculated based on the method described in Guemri et al. (2019). Similarly, given that each move induced by shift, swap, multi-shift, and multi-swap involves only two doors, the total number of pallets of the two doors can be directly computed. Subsequently, with the assistance of an $(|\mathcal{I}| + |\mathcal{J}|)$-dimensional vector W, where W_k $(k \in \mathcal{I} \cup \mathcal{J})$ records the total number of pallets allocated to k, the move value $\Delta_P(mv)$ of a given move mv related to the change in terms of the penalty function P can be conveniently determined using Equation (16) in constant time $O(1)$.

3.3.3. Mixed tabu strategy

This study proposes a mixed tabu strategy combining the advantages of the attribute-based (Glover, 1997) and solution-based (Woodruff & Zemel, 1993; Wang et al., 2017; Wu et al., 2020) tabu strategies to prevent the neighborhood based local search from short- or long-term cycling.

An attribute-based tabu search is accomplished by forbidding the reverse of a performed move in the next few iterations. The tabu strategy has been widely used in the literature. However, this tabu search can only prevent the search from falling into short-term cycling. Some early tabu decisions become obsolete after several iterations. Thus, the same solution can be revisited several times.

The solution-based tabu strategy employs hashing techniques (Woodruff & Zemel, 1993) to mark visited solutions effectively and determine the tabu status of a new solution rapidly. A solution-based tabu search can prevent a search from any short- or long-term cycling by accurately recording each visited solution, given that no visited solution is revisited. However, the absolute prohibition of all visited solutions may prevent the algorithm from exploring promising trajectories. This study proposes a mixed tabu strategy that takes advantage of these two strategies. In the proposed mixed tabu strategy, each solution is allowed to be visited at most β times during the search. The same hashing techniques adopted in a solution-based tabu search are applied to keep track of the number of times each solution is visited during the search (Wang et al., 2017; Wu et al., 2020). In addition, the traditional attribute-based tabu strategy is used, where the reverse move of a performed move is forbidden for the next few iterations to prevent the search from rapidly returning to the recently visited solution or avoid short-term cycles between two or more consecutively visited solutions. Only the solutions meeting the following criteria are allowed in the proposed mixed tabu strategy: (1) The solution is induced by allowed moves. Each time a truck m is moved from its original door i to another door i', moving the truck m back to door i for the next tt subsequent iterations is forbidden, where tt denotes the tabu tenure; (2) The number of times the solution is visited is less than β, where β is a parameter. The detailed implementation of the mixed tabu strategy is provided in the Appendix B.

3.4. Reward updating procedure

After the SO procedure is applied to improve the initial solution produced by the Q-learning truck-to-door assignment procedure, a reward updating procedure is triggered to gather useful information from the discovered local optimum solution. This procedure is applied during the subsequent search process to guide the proposed RL-SO approach toward new promising areas. The RL-SO iteratively explores the search space by alternating between the SO procedure and the Q-learning reinforced method to balance search intensification and diversification.

The reward updating procedure updates matrix R by comparing the starting solution S with local optimum solution S_l by checking whether a truck in S_l is moved from its original door to another door or if it stays in the same door as in S. Subsequently, the reward updating procedure rewards the unchanged (correct) door, penalizes the changed (incorrect) door, and compensates for the expected door using a reward value rwd, which is adaptively set according to the quality of the solution S_l and favors high-quality solutions. In particular, if the local optimum solution S_l is better than the best solution found thus far, the reward rwd is set to $ibsc+1$, where $ibsc$ records the number of times the best solution has been consecutively updated. Otherwise, reward rwd is set to 1.

If the assigned door i of each truck m remains unchanged from S to S_l, then its originally assigned door i is rewarded, and its value $R(s_t, a_{mi})$ in matrix R is updated as follows:

$$R(s_t, a_{mi}) = R(s_t, a_{mi}) + rwd, \ \forall t \in \{0\} \cup \mathcal{M}. \tag{19}$$

If the truck m is moved from its original door i to another door i' in the local optimum solution S_l, its former door i is penalized as follows:

$$R(s_t, a_{mi}) = R(s_t, a_{mi}) - rwd, \ \forall t \in \{0\} \cup \mathcal{M}. \tag{20}$$

The new door i' is also compensated by updating its value in matrix R as follows:

$$R(s_t, a_{mi'}) = R(s_t, a_{mi'}) + rwd, \ \forall t \in \{0\} \cup \mathcal{M}. \tag{21}$$

We also give an additional reward rwd to the corresponding state-action pair when the starting solution S is constructed. If a_{mi} is performed in state s_t in the Q-learning procedure, its reward value $R(s_t, a_{mi})$ is updated as:

$$R(s_t, a_{mi}) = R(s_t, a_{mi}) + rwd. \tag{22}$$

After the reward updating procedure is applied to the reward function R, a reward smoothing technique is employed for the following reasons. Decisions made long ago can be useless and may mislead the search. Therefore, these decisions are considered less critical and should be removed periodically. The reward smoothing technique is inspired by the forgetting mechanisms in local search algorithms for satisfiability (Hutter et al., 2002). It works as follows. If the value of each element $R(s_t, a_{mi})$ achieves a predefined threshold r_0, then it is reduced by multiplying a smoothing coefficient ρ ($0 < \rho < 1$) to forget some earlier decisions. This study experimentally set $r_0 = 100$ and $\rho = 0.5$.

4. Computational results

This section reports the extensive computational results of the FBMIP formulation and the proposed RL-SO algorithm on benchmark instances widely used in the literature and compares them with state-of-the-art formulations and solution approaches on the CDAP.

Table 3: Settings of the parameters.

Parameter	Section	Description	Considered values	Final value
α	3.2	Learning rate of Q-learning	{0.3, 0.5, 0.6, 0.7, 0.8}	0.5
γ	3.2	Discount factor of Q-learning	{0.3, 0.5, 0.6, 0.7, 0.8}	0.5
ε	3.2	Probability of ε-greedy	{0.6, 0.7, 0.8, 0.9, 0.95}	0.8
ω	3.3	Search depth of the SO	{1e3, 2e3, 3e3, 4e3, 5e3}	2e3
tt	3.3	Tabu tenure	{5, 10, 15, 20, 25}	10
β	3.3	The allowed visit time for each solution	{2, 5, 10, 12, 20}	5

4.1. Benchmark instances and parameter setting

The computational assessments were based on two sets of 99 benchmark instances in the literature proposed by Guignard et al. (2012).

- **SetA.** This set includes 50 instances and is generated as follows. The number of origins is set to be equal to that of the destinations selected from {8, 9, 10, 11, 12, 15, 20}. Moreover, the number of inbound doors is equal to that of the outbound doors, selected from {4, 5, 6, 7, 10}. The distance between a pair of inbound/outbound doors depends on the positions of the two doors in the cross-docking terminal. The distance between two face-to-face doors is set to 8. By contrast, the distance between two doors that are not face-to-face is set to $8+I$, where I is the I-th successive door counted from its faced door. The number of pallets moving from the origin to the destination is defined by a flow matrix produced by setting the values of 25% of the elements in this matrix as random integers taken from the interval [10, 50] while setting the values of the remaining elements to zero. The capacity of each door is set to an identical value computed by dividing the total number of pallets from all origins by the total number of doors and then adding a capacity slack to this quotient. The capacity slack is set to be {5%,10%,15%,20%,30%} of the door capacity.

- **SetB.** This set includes 49 large-scale instances generated in the same manner as in **SetA**. The number of origins/destinations in this set is selected from {25, 50, 75, 100}, and the number of inbound/outbound doors is selected from {10, 20, 30, 43}.

All instances in the two sets are named according to the format "aa×bbScc", where "aa" refers to the number of origins/destinations, "bb" presents the number of inbound/outbound doors, and "cc" denotes the capacity slack.

The proposed RL-SO algorithm was implemented in C++ and compiled using GNU g++ with the -O3 option flag [1]. All experiments were performed on a 2.6 GHz Intel E5-2670 computer with 2G RAM, running Linux. The FBMIP formulation was solved using the general MIP solver CPLEX 22.1.0.

RL-SO requires six parameters (Table 3). IRACE (Birattari et al., 2002), a tool for automatic parameter configuration, was used to tune these parameters. IRACE was run using 20 randomly selected instances. The tuning budget was set to 2000 RL-SO executions, with a time limit of 300 s per execution. Table 3 lists the parameter settings recommended by IRACE. This setting can be considered the default setting of the algorithm and was used for all experiments in this study.

4.2. Computational results of the FBMIP formulation and comparisons with existing MIP formulations

Various MIP formulations for the CDAP have been reported. Gelareh et al. (2020) introduced eight MIP formulations and compared them with three previous MIP formulations of the CDAP, one investigated by Zhu et al. (2009) and the other two introduced by Nassief et al. (2016). Gelareh et al. (2020) conducted a comprehensive comparative analysis to identify the best MIP formulation among the 11 compared formulations. Their analysis identified the formulation proposed by Gelareh et al. (2020) (denoted by $M^{2',0}$ in this work, see Appendix A) as the overall best formulation. Most of these compared formulations are

[1] The code of the proposed algorithm will be publicly available at https://github.com/MINGJIE666/CDAP.

Table 4: Comparison between FBMIP and $M^{2',0}$ (Gelareh et al., 2020).

	FBMIP			$M^{2',0}$				FBMIP			$M^{2',0}$		
Instance	UB	GAP	T(s)	UB	GAP	T(s)	Instance	UB	GAP	T(s)	UB	GAP	T(s)
8×4S5	**5174**	**0**	0.08	**5174**	**0**	0.19	12×6S30	**10228**	**0**	2.26	**10228**	**0**	7.40
8×4S10	**5169**	**0**	0.11	**5169**	**0**	0.19	15×6S5	**13927**	**0**	21.01	**13927**	**0**	31.14
8×4S15	**5112**	**0**	0.08	**5112**	**0**	0.16	15×6S10	**13803**	**0**	41.03	**13803**	**0**	61.32
8×4S20	**5086**	**0**	0.11	**5086**	**0**	0.19	15×6S15	**13765**	**0**	111.62	**13765**	**0**	39.62
8×4S30	**5063**	**0**	0.05	**5063**	**0**	0.17	15×6S20	**13720**	**0**	76.51	**13720**	**0**	13.14
9×4S5	**6047**	**0**	0.06	**6047**	**0**	0.33	15×6S30	**13567**	**0**	66.47	**13567**	**0**	11.13
9×4S10	**6027**	**0**	0.08	**6027**	**0**	0.38	15×7S5	**15054**	**0**	105.85	**15054**	**0**	220.38
9×4S15	**5976**	**0**	0.08	**5976**	**0**	0.20	15×7S10	**14810**	**0**	239.41	**14810**	**0**	316.23
9×4S20	**5937**	**0**	0.06	**5937**	**0**	0.14	15×7S15	**14657**	**0**	180.16	**14657**	**0**	250.24
9×4S30	**5904**	**0**	0.06	**5904**	**0**	0.13	15×7S20	**14514**	**0**	136.78	**14514**	**0**	95.65
10×4S5	**6518**	**0**	0.13	**6518**	**0**	0.69	15×7S30	**14409**	**0**	153.62	**14409**	**0**	29.94
10×4S10	**6325**	**0**	0.06	**6325**	**0**	0.48	20×10S5	**29945**	**3.92**	7200.00	30233	10.1	7200.00
10×4S15	**6296**	**0**	0.08	**6296**	**0**	0.59	20×10S10	**29367**	**4.62**	7200.00	29638	9.77	7200.00
10×4S20	**6267**	**0**	0.13	**6267**	**0**	0.20	20×10S15	**29135**	**5.15**	7200.00	29446	9.56	7200.00
10×4S30	**6193**	**0**	0.08	**6193**	**0**	0.28	20×10S20	**28945**	**5.56**	7200.00	28972	9.9	7200.00
10×5S5	**6616**	**0**	0.22	**6616**	**0**	0.26	20×10S30	**28549**	**5.91**	7200.00	28826	8.31	7200.00
10×5S10	**6476**	**0**	0.17	**6476**	**0**	1.19	25×10S10	**48999**	**10.91**	7200.00	50720	18.76	7200.00
10×5S15	**6397**	**0**	0.13	**6397**	**0**	1.05	25×10S20	**48357**	**11.21**	7200.00	49247	14.8	7200.00
10×5S20	**6342**	**0**	0.16	**6342**	**0**	0.63	25×10S30	**47825**	**12.73**	7200.00	49030	15.66	7200.00
10×5S30	**6308**	**0**	0.17	**6308**	**0**	0.94	50×10S10	**191913**	**19.29**	7200.00	208111	26.47	7200.00
11×5S5	**7812**	**0**	0.66	**7812**	**0**	1.78	50×10S20	**189620**	**18.7**	7200.00	204344	25.4	7200.00
11×5S10	**7572**	**0**	1.06	**7572**	**0**	1.56	50×10S30	**185238**	**16.7**	7200.00	199146	23.19	7200.00
11×5S15	**7535**	**0**	1.11	**7535**	**0**	1.41	50×20S10	**263508**	**40.54**	7200.00	283869	45.46	7200.00
11×5S20	**7439**	**0**	0.59	**7439**	**0**	0.75	50×20S20	**246567**	**37.03**	7200.00	267271	42.07	7200.00
11×5S30	**7420**	**0**	0.95	**7420**	**0**	0.95	50×20S30	**231687**	**32.28**	7200.00	254322	39.13	7200.00
12×5S5	**8072**	**0**	0.23	**8072**	**0**	1.12	50×30S20	**288761**	**46.78**	7200.00	-	-	7200.00
12×5S10	**7978**	**0**	2.13	**7978**	**0**	1.55	50×30S30	**282898**	**45.71**	7200.00	-	-	7200.00
12×5S15	**7939**	**0**	2.25	**7939**	**0**	0.88	75×10S10	**443921**	**22.16**	7200.00	-	-	7200.00
12×5S20	**7939**	**0**	1.80	**7939**	**0**	1.89	75×20S10	**549656**	**38.64**	7200.00	-	-	7200.00
12×5S30	**7923**	**0**	2.33	**7923**	**0**	1.75	75×30S10	**747982**	**54.38**	7200.00	-	-	7200.00
12×6S5	**10891**	**0**	4.02	**10891**	**0**	6.20	100×10S10	**781671**	**23.82**	7200.00	-	-	7200.00
12×6S10	**10456**	**0**	3.64	**10456**	**0**	6.50	100×20S10	**1013862**	**40.39**	7200.00	-	-	7200.00
12×6S15	**10362**	**0**	2.84	**10362**	**0**	5.40	100×30S10	**1267570**	**52.18**	7200.00	-	-	7200.00
12×6S20	**10312**	**0**	3.26	**10312**	**0**	3.20							

four indexed formulations that linearize the quadratic formulation MINP in Section 2 by introducing a four indexed binary variable $z_{minj} = x_{mi}y_{nj}$.

The FBMIP formulation was compared with the best-performing formulation $M^{2',0}$. Both formulations were solved using CPLEX 22.1.0 on the same computer to ensure a fair comparison. CPLEX 22.1.0 stops when the given instance is solved to optimality or when the given time limit of 2 h is reached. Table 4 summarizes the comparative results of the two formulations. For each instance, we report the achieved upper bound (UB), the percentage gap between the lower bound (LB) and UB (GAP), and the running time in seconds (T(s)) required to reach the optimum solution. If an instance cannot be solved to optimality within the time limit, the reported value for T(s) will be 7200 s.

The results reported in Table 4 demonstrate the competitiveness of the FBMIP formulation on the tested instances. In particular, both approaches can obtain optimal solutions for all instances with up to 15 incoming trucks. However, the FBMIP formulation reached the optimum solution faster than the $M^{2',0}$ in most cases. Both formulations failed to solve these instances to optimality within the given time limit when the large instances with 20 or more incoming trucks were considered. However, the FBMIP outperformed the $M^{2',0}$ on each instance by producing smaller UB and larger LB. In particular, the FBMIP obtained feasible solutions for all large instances with at least 50 incoming trucks. By contrast, the $M^{2',0}$ failed to produce feasible solutions within the given time limit.

Table 5: Comparison between MILP and $M^{2',0}$ (Gelareh et al., 2020) on median size instances.

	FBMIP			$M^{2',0}$				FBMIP			$M^{2',0}$		
Instance	UB	GAP	T(s)	UB	GAP	T(s)	Instance	UB	GAP	T(s)	UB	GAP	T(s)
16×8S5	**17254**	**0**	543.65	**17254**	**0**	1052.32	18×8S5	**20156**	**0**	4232.41	21532	3.2	7200.00
16×8S10	**17012**	**0**	458.96	**17012**	**0**	987.45	18×8S10	**19564**	**0**	4459.64	20586	3.1	7200.00
16×8S15	**16852**	**0**	475.32	**16852**	**0**	864.65	18×8S15	**19123**	**0**	3745.21	19423	1.5	7200.00
16×8S20	**16234**	**0**	412.39	**16234**	**0**	854.67	18×8S20	**18952**	**0**	4005.63	**18952**	**0**	6400.21
16×8S30	**15923**	**0**	395.68	**15923**	**0**	683.52	18×8S30	**18536**	**0**	3598.32	**18536**	**0**	6932.63
17×8S5	**18357**	**0**	1202.31	**18357**	**0**	2403.21	19×8S5	**26412**	**0**	6235.41	27123	7.3	7200.00
17×8S10	**18001**	**0**	1525.85	**18001**	**0**	2031.85	19×8S10	**26341**	**0**	4432.21	26852	6.5	7200.00
17×8S15	**17821**	**0**	1763.42	**17821**	**0**	1502.64	19×8S15	**26121**	**0**	3583.12	26556	7.1	7200.00
17×8S20	**17513**	**0**	2054.39	**17513**	**0**	1203.21	19×8S20	**25864**	**0**	5612.37	26153	5.8	7200.00
17×8S30	**17356**	**0**	2563.46	**17356**	**0**	1007.35	19×8S30	**25743**	**0**	4621.62	26054	5.2	7200.00

To further demonstrate the effectiveness of the FBMIP formulation, we produced 20 medium-sized instances using the same method as SetA and SetB. Each instance has a number of incoming trucks ranging from 16 to 19. Table 5 summarizes the comparative results between the FBMIP and the $M^{2',0}$ on these instances. The results show that the FBMIP was able to solve all 20 instances with up to 19 incoming trucks to optimality. By contrast, the $M^{2',0}$ formulation failed to solve three instances with 18 incoming trucks and five instances with 19 incoming trucks to optimality.

4.3. Comparative results of RL-SO with state-of-the-art methods

The following six leading algorithms from the literature were compared with the proposed RL-SO algorithm: the best-performing MIP formulation $M^{2',0}$ (Gelareh et al., 2020), two local search based heuristics named LS1 and LS2 (Guignard et al., 2012), the convex hull relaxation heuristic CHR proposed by Guignard et al. (2012), and two probabilistic tabu search heuristics PTS1 and PTS2 proposed by Guemri et al. (2019).

LS1 and LS2 were run on an AMD Phenom 9600 with a 2.31 GHz processor running Windows XP OS. The CHR was run on a 3.00 GHz Intel Xeon CPU E5450 processor running Linux (Guignard et al., 2012). PTS1 and PTS2 were implemented in Java and executed on an Intel Xeon E3-1505 M v5 processor with a 2.80 GHz CPU and 16 GB RAM. The stopping condition for PTS1 and PTS2 is defined by the maximum number of iterations in the local search, which is set to 10^5 for SetA and 2×10^5 for SetB. PTS1 and PTS2 were run 10 times using different random seeds (Guemri et al., 2019). $M^{2',0}$ was solved using CPLEX 22.1.0 with a time limit of 7200 s (Gelareh et al., 2020). A completely fair comparison between the proposed RL-SO and these reference algorithms is challenging due to the differences in computing platforms, coding languages, compiling options, and termination conditions. Thus, this study focused on the solution quality in terms of objective values and provided timing information for indicative purposes.

Given the stochastic nature of the proposed RL-SO, similar to Guemri et al. (2019), the algorithm was executed 10 times independently per instance. The time limit of the RL-SO per run was set to $3|\mathcal{M}|$ s, where $|\mathcal{M}|$ is the number of incoming trucks. Because the source codes of these reference algorithms were unavailable, the computation results reported by these comparative approaches, except those of the $M^{2',0}$, were directly compiled from their corresponding papers.

Tables 6 and 7 summarize the computational results of the proposed RL-SO compared with the reference algorithms for SetA and SetB, respectively. In these tables, the first column presents the name of each instance. The second column, "BKS", presents the objective value of the best-known solution reported in the literature for each instance. Columns "OF" and "AVG" report the best and average objective values achieved by each algorithm on all tested runs, respectively. Finally, column "T(s)" reports the average running time in seconds required by each algorithm to reach its best objective value.

Table 6 shows that the results obtained by the proposed RL-SO algorithm were highly competitive compared with those obtained by the reference algorithms in terms of the best and average objective values. In particular, the $M^{2',0}$ achieved optimal solutions for instances with up to 15 incoming trucks and seven inbound doors. The proposed RL-SO algorithm obtained optimum solutions for all instances with a known optimum solution within 2 s. For large instances with at least 15 incoming trucks, the exact method could not achieve an optimal solution within the given time limit. For these large instances, the proposed RL-SO obtained a better (smaller) objective value than the exact algorithm, except for 20×10S15. When the six heuristic approaches were compared, the RL-SO, PTS1, and PTS2 performed better than the LS1, LS2, and CHR in terms of the best objective values achieved. The RL-SO, PTS1, and PTS2 reached the best-known solutions for 49 of the 50 SetA instances. By contrast, the LS1, LS2, and CHR reached the best-known solutions only for 28, 26, and 27 instances, respectively. Moreover, the proposed RL-SO algorithm is more robust than the PTS1 and PTS2 in terms of the average objective value because it consistently reached the same objective value equal to its best objective value for all 10 runs within 2 s. Finally, the Wilcoxon signed-rank test revealed statistically significant differences between RL-SO and $M^{2',0}$, LS1, LS2, and CHR in terms of the best objective values, as indicated by the small p-values (< 0.05). Additionally, the small p-values (< 0.05) between RL-SO and PTS1 and PTS2 in terms of the average objective values confirm the robustness of the proposed RL-SO.

Table 6: Comparative results on 50 "SetA" instances with $M^{2',0}$(Gelareh et al., 2020), LS1 (Guignard et al., 2012), LS2 (Guignard et al., 2012), CHR (Guignard et al., 2012), PTS1 (Guemri et al., 2019), and PTS2 (Guemri et al., 2019).

		$M^{2',0}$		LS1		LS2		CHR		PTS1			PTS2			RL-SO		
Instance	BKS	OF	T(s)	OF	T(s)	OF	T(s)	OF	T(s)	OF	AVG	T(s)	OF	AVG	T(s)	OF	AVG	T(s)
8×4S5	**5174***	**5174**	0.08	**5174**	1	**5174**	2	**5174**	13	**5174**	5174.0	3.87	**5174**	5174.0	1.69	**5174**	**5174.0**	0.08
8×4S10	**5169***	**5169**	0.11	**5169**	2	**5169**	3	**5169**	14	**5169**	5169.0	3.68	**5169**	5169.0	1.12	**5169**	**5169.0**	0.09
8×4S15	**5112***	**5112**	0.08	**5112**	2	**5112**	2	**5112**	9	**5112**	5112.0	3.46	**5112**	5112.0	0.98	**5112**	**5112.0**	0.10
8×4S20	**5086***	**5086**	0.11	**5086**	2	**5086**	3	**5086**	17	**5086**	5086.0	3.29	**5086**	5086.0	0.87	**5086**	**5086.0**	0.11
8×4S30	**5063***	**5063**	0.05	**5063**	2	**5063**	2	**5063**	17	**5063**	5063.0	3.20	**5063**	5063.0	0.83	**5063**	**5063.0**	0.12
9×4S5	**6047***	**6047**	0.06	**6047**	2	**6047**	3	**6047**	15	**6047**	6047.0	4.50	**6047**	6047.0	1.25	**6047**	**6047.0**	0.08
9×4S10	**6027***	**6027**	0.08	**6027**	2	**6027**	3	**6027**	17	**6027**	6027.0	4.11	**6027**	6027.0	1.09	**6027**	**6027.0**	0.07
9×4S15	**5976***	**5976**	0.08	**5976**	3	**5976**	3	**5976**	21	**5976**	5976.0	3.93	**5976**	5976.0	0.92	**5976**	**5976.0**	0.09
9×4S20	**5937***	**5937**	0.06	**5937**	2	**5937**	2	**5937**	19	**5937**	5937.0	3.90	**5937**	5937.0	0.97	**5937**	**5937.0**	0.10
9×4S30	**5904***	**5904**	0.06	**5904**	2	**5904**	2	**5904**	14	**5904**	5904.0	3.74	**5904**	5904.0	0.90	**5904**	**5904.0**	0.12
10×4S5	**6518***	**6518**	0.13	**6518**	5	**6518**	6	**6518**	16	**6518**	6518.0	5.10	**6518**	6518.0	1.38	**6518**	**6518.0**	0.06
10×4S10	**6325***	**6325**	0.06	**6325**	3	**6325**	3	**6325**	30	**6325**	6325.0	4.87	**6325**	6325.0	1.09	**6325**	**6325.0**	0.08
10×4S15	**6296***	**6296**	0.08	**6296**	3	**6296**	3	**6296**	28	**6296**	6296.0	4.55	**6296**	6296.0	0.98	**6296**	**6296.0**	0.09
10×4S20	**6267***	**6267**	0.13	**6267**	3	**6267**	4	**6267**	26	**6267**	6267.0	4.40	**6267**	6267.0	0.97	**6267**	**6267.0**	0.11
10×4S30	**6193***	**6193**	0.08	**6193**	3	**6193**	3	**6193**	19	**6193**	6193.0	4.13	**6193**	6193.0	1.07	**6193**	**6193.0**	0.11
10×5S5	**6616***	**6616**	0.22	**6616**	6	**6616**	6	**6616**	13	**6616**	6616.0	4.54	**6616**	6616.0	1.80	**6616**	**6616.0**	0.10
10×5S10	**6476***	**6476**	0.17	**6476**	4	**6476**	5	**6476**	23	**6476**	6476.0	4.45	**6476**	6476.0	1.51	**6476**	**6476.0**	0.13
10×5S15	**6397***	**6397**	0.13	**6397**	3	**6397**	4	**6397**	16	**6397**	6397.0	4.14	**6397**	6397.0	1.37	**6397**	**6397.0**	0.10
10×5S20	**6342***	**6342**	0.16	6374	3	6363	4	6354	18	**6342**	6342.0	4.02	**6342**	6342.0	1.32	**6342**	**6342.0**	0.09
10×5S30	**6308***	**6308**	0.17	6324	3	6333	4	6333	18	**6308**	6308.0	3.86	**6308**	6308.0	1.31	**6308**	**6308.0**	0.13
11×5S5	**7812***	**7812**	0.66	**7812**	12	**7812**	12	**7812**	16	**7812**	7812.0	5.23	**7812**	7812.0	2.12	**7812**	**7812.0**	0.11
11×5S10	**7572***	**7572**	1.06	**7572**	5	**7572**	6	**7572**	18	**7572**	7572.0	4.95	**7572**	7572.0	1.61	**7572**	**7572.0**	0.21
11×5S15	**7535***	**7535**	1.11	7542	5	7542	5	7542	20	**7535**	7535.0	4.76	**7535**	7535.0	1.36	**7535**	**7535.0**	0.25
11×5S20	**7439***	**7439**	0.59	7465	5	7465	5	7478	17	**7439**	7439.0	4.47	**7439**	7439.0	1.29	**7439**	**7439.0**	0.36
11×5S30	**7420***	**7420**	0.95	7428	4	7424	5	**7420**	18	**7420**	7420.0	4.23	**7420**	7421.6	1.36	**7420**	**7420.0**	0.12
12×5S5	**8072***	**8072**	0.23	**8072**	5	8072	7	**8072**	13	**8072**	8072.0	5.81	**8072**	8072.0	1.76	**8072**	**8072.0**	0.85
12×5S10	**7978***	**7978**	2.13	**7978**	4	7988	5	7991	15	**7978**	7978.0	5.53	**7978**	7978.0	1.58	**7978**	**7978.0**	0.41
12×5S15	**7939***	**7939**	2.25	**7939**	4	**7939**	6	**7939**	18	**7939**	7939.0	5.36	**7939**	7939.0	1.48	**7939**	**7939.0**	0.52
12×5S20	**7939***	**7939**	1.80	**7939**	5	**7939**	6	**7939**	18	**7939**	7939.0	5.06	**7939**	7939.0	1.43	**7939**	**7939.0**	0.63
12×5S30	**7923***	**7923**	2.33	7942	6	7925	6	7925	21	**7923**	7923.0	4.68	**7923**	7923.0	1.40	**7923**	**7923.0**	0.74
12×6S5	**10891***	**10891**	4.02	**10891**	10	10894	14	10894	14	**10891**	10891.0	5.36	**10891**	10891.0	2.72	**10891**	**10891.0**	0.32
12×6S10	**10456***	**10456**	3.64	10475	6	10496	7	10480	14	**10456**	10456.0	5.13	**10456**	10456.0	1.98	**10456**	**10456.0**	0.95
12×6S15	**10362***	**10362**	2.84	10395	5	10420	6	10374	17	**10362**	10362.0	4.89	**10362**	10378.5	1.82	**10362**	**10362.0**	0.63
12×6S20	**10312***	**10312**	3.26	10331	7	10339	6	10323	16	**10312**	10312.0	4.61	**10312**	10312.0	1.92	**10312**	**10312.0**	0.84
12×6S30	**10228***	**10228**	2.26	**10228**	6	**10228**	7	10277	20	**10228**	10228.0	4.22	**10228**	10228.0	1.84	**10228**	**10228.0**	0.63
15×6S5	**13927***	**13927**	21.01	13960	13	**13927**	16	13983	18	**13927**	13927.0	7.32	**13927**	13927.0	2.47	**13927**	**13927.0**	0.71
15×6S10	**13803***	**13803**	41.03	13856	11	13852	11	13872	15	**13803**	13803.0	7.02	**13803**	13810.7	2.20	**13803**	**13803.0**	0.85
15×6S15	**13765***	**13765**	111.62	13769	9	13799	12	**13765**	18	**13765**	13765.0	6.54	**13765**	13792.3	2.23	**13765**	**13765.0**	1.01
15×6S20	**13720***	**13720**	76.51	13769	10	13754	11	**13720**	17	**13720**	13720.0	6.17	**13720**	13750.0	2.31	**13720**	**13720.0**	0.96
15×6S30	**13567***	**13567**	66.47	**13567**	10	**13567**	10	13626	16	**13567**	13567.0	5.68	**13567**	13585.2	2.22	**13567**	**13567.0**	0.82
15×7S5	**15054***	**15054**	105.85	**15054**	25	**15054**	25	15096	26	**15054**	15054.0	6.90	**15054**	15063.1	3.36	**15054**	**15054.0**	0.96
15×7S10	**14810***	**14810**	239.41	14841	15	14818	17	**14810**	22	**14810**	14810.0	6.49	**14810**	14843.1	2.70	**14810**	**14810.0**	1.01
15×7S15	**14657***	**14657**	180.16	14678	13	14680	14	14678	22	**14657**	14657.2	6.16	**14657**	14658.2	2.68	**14657**	**14657.0**	1.36
15×7S20	**14514***	**14514**	136.78	14596	14	14533	16	14533	19	**14514**	14514.0	5.80	**14514**	14537.6	2.72	**14514**	**14514.0**	1.12
15×7S30	**14409***	**14409**	153.62	14415	13	14423	13	14414	19	**14409**	14409.0	5.31	**14409**	14413.2	2.61	**14409**	**14409.0**	0.85
20×10S5	**29907**	30233	7200.00	29945	254	30033	45	29933	87	**29907**	29909.6	9.16	**29907**	30004.4	6.39	**29907**	**29907.0**	0.71
20×10S10	**29236**	29638	7200.00	29286	36	29286	36	29498	52	**29236**	29253.3	8.49	**29236**	29567.3	5.01	**29236**	**29236.0**	1.06
20×10S15	**29134**	29446	7200.00	29278	35	29184	37	29529	65	29135	29135.5	7.77	29135	29345.7	4.81	29135	29135.0	1.63
20×10S20	**28945**	28972	7200.00	29130	38	28992	40	29089	63	**28945**	28951.2	7.18	**28945**	29051.5	4.81	**28945**	**28945.0**	0.95
20×10S30	**28533**	28826	7200.00	28800	39	28541	41	28571	62	**28533**	28539.6	6.85	**28533**	28741.3	4.82	**28533**	**28533.0**	1.21
AVG	10741.8	10769.0	743.3	10764.7	13.6	10755.5	10.3	10768.5	22.8	**10741.9**	10742.5	5.2	**10741.9**	10764.4	2.0	**10741.9**	**10741.9**	0.5
#BEST	50	45		28		26		27		**49**			**49**			**49**		
p-value	3.2E-01	4.3E-02		4.0E-05		2.7E-05		2.7E-05		1	2.8E-02		1	4.4E-04				

*: the result is proved to optimality.

Table 7 shows that the RL-SO improved the best-known results for 43 of the 49 instances in SetB. Compared with the LS1, LS2, CHR, PTS1, and PTS2, the proposed RL-SO algorithm delivered highly competitive results in terms of the best and average objective values. In particular, the RL-SO achieved better results in terms of the best objective value on 49, 49, 49, 44, and 45 cases and worse results on 0, 0, 0, 1, and 0 cases. The small p-values (< 0.05) from the Wilcoxon signed-rank test further confirmed the statistically significant differences between the RL-SO and reference algorithms in terms of the best and average objective values.

In summary, this comparative assessment confirmed the effectiveness and robustness of the proposed RL-SO algorithm on the CDAP datasets.

4.4. RL-SO for other combinatorial optimization problems

Given the general nature of the main components of the proposed RL-SO algorithm, such as the Q-learning procedure, the feasible and infeasible search, and the mixed tabu strategy, the RL-SO algorithm can be used to solve other combinatorial optimization problems. To demonstrate this, we tested the RL-SO algorithm on two other widely studied problems, namely the uncapacitated cross-dock door assignment problem (UCDAP) and the quadratic knapsack problem with conflict graph (QKPCG).

Table 7: Comparative results on 49 "SetB" instances with LS1 (Guignard et al., 2012), LS2 (Guignard et al., 2012), CHR (Guignard et al., 2012), PTS1 (Guemri et al., 2019), and PTS2 (Guemri et al., 2019).

		LS1		LS2		CHR		PTS1			PTS2			RL-SO		
Instance	BKS	OF	T(s)	OF	T(s)	OF	T(s)	OF	AVG	T(s)	OF	AVG	T(s)	OF	AVG	T(s)
25×10S5	**49013**	49144	194	49335	198	49904	105.5	**49013**	49014.8	26.11	**49013**	49013.0	13.81	**49013**	**49013.0**	0.14
25×10S10	**48672**	48949	116	48941	129	49332	138.9	**48672**	48699.3	23.49	48740	48869.5	12.50	**48672**	**48672.0**	0.16
25×10S15	**48407**	48556	122	48504	131	48770	132.5	**48407**	48415.6	20.08	**48407**	48477.0	12.10	**48407**	**48407.0**	1.48
25×10S20	**47926**	48215	85	48235	90	48338	120.0	47934	47949.3	19.11	**47926**	47977.6	12.24	**47926**	**47926.0**	0.27
25×10S30	**47314**	47480	77	47426	81	47652	113.5	**47314**	47356.5	19.61	**47314**	47368.1	12.25	**47314**	**47314.0**	0.44
25×20S30	51533	51921	181	51741	200	52447	112.0	51533	51618.4	19.81	51562	52334.2	28.51	**51523**	**51523.0**	12.99
50×10S5	191160	191773	4039	191788	4496	192114	178.0	191160	191241.3	78.29	191186	192350.7	26.58	**191040**	**191062.0**	56.94
50×10S10	189166	189409	5345	189833	5313	190508	155.9	189166	189478.5	66.16	189573	190316.3	26.19	**189134**	**189190.0**	77.04
50×10S15	187315	188006	3363	188264	3120	187753	164.0	187315	187417.9	69.16	187377	188834.9	26.45	**187256**	**187257.2**	60.10
50×10S20	186085	186800	3854	186578	3729	187901	181.7	186085	186246.2	72.28	185975	186788.2	26.24	**185975**	185996.9	95.92
50×10S30	183249	183961	1652	184013	1567	184532	195.8	183249	183373.7	79.04	183234	183943.9	25.85	**183145**	**183178.3**	94.33
50×20S5	237656	238048	7074	239673	7308	240288	504.0	237656	238244.5	59.28	239835	241379.3	50.66	**237396**	**237490.2**	69.84
50×20S10	233341	235178	7877	234807	7741	236124	626.8	233341	234045.4	44.56	235326	236932.9	51.59	**233319**	**233843.4**	74.52
50×20S15	229638	230758	7815	230666	7490	233324	604.8	229638	230429.2	45.58	230375	233110.0	48.04	**229342**	**229756.7**	84.32
50×20S20	226448	227698	7161	227883	7328	228918	472.0	226448	227134.2	45.65	228332	230201.4	50.93	**226051**	**226512.0**	74.02
50×20S30	220748	221892	7138	222060	6951	223687	357.0	220748	221313.8	48.48	221322	223423.5	45.85	**220291**	**220659.3**	80.16
50×30S15	273166	275973	5975	275121	6950	276237	262.7	273166	274455.9	50.52	276938	278798.1	91.23	**272504**	**273507.1**	105.56
50×30S20	261725	264199	2467	263790	3056	264971	389.9	261725	263099.9	43.55	264729	267802.5	86.29	**260923**	**262330.6**	95.70
50×30S30	252639	254056	1879	254795	1962	257529	260.9	252639	253680.2	45.23	255151	258277.8	77.02	**251956**	**252318.5**	75.44
50×43S30	330285	332318	6977	330661	7381	332947	323.0	330285	332378.3	47.46	335036	341233.4	121.07	**329309**	**330412.3**	93.35
75×10S5	439055	440248	9114	440420	9446	442749	168.7	439055	439478.3	158.11	440181	441088.8	44.82	**438575**	**438712.0**	108.32
75×10S10	434275	435985	8842	435970	9178	437642	219.8	434275	435134.7	157.91	435595	437051.9	44.88	**433876**	**434143.1**	126.28
75×10S15	430005	431405	8429	431686	8521	432086	209.8	430005	430781.3	171.16	430663	432183.0	45.28	**429758**	**430152.5**	115.27
75×10S20	**426385**	427468	8772	427522	8127	429513	78.2	**426385**	427176.8	179.44	427168	428920.9	44.74	426431	426680.4	138.27
75×10S30	419651	420851	6754	420660	6643	421983	215.0	419651	420123.3	198.28	420619	421566.1	44.48	**419526**	**419565.3**	110.55
75×20S5	529131	531762	8174	532873	8746	533701	789.0	529131	529964.2	86.09	532968	535724.6	72.92	**528769**	**529105.9**	131.81
75×20S10	520127	523447	9269	521970	9665	525098	803.0	520127	521708.1	86.09	524001	526829.3	72.91	**519927**	**520569.6**	99.70
75×20S15	512170	514760	9778	514981	9614	513166	602.0	512170	512788.7	87.11	515098	519040.7	73.93	**512114**	**512366.2**	121.13
75×20S20	504502	506346	8617	506114	9793	508862	627.8	504502	505141.8	90.33	506313	511407.4	72.29	**504434**	**504823.7**	120.67
75×20S30	491796	493417	9549	493977	9855	494897	412.9	491796	492429.4	97.83	493403	498179.1	73.09	**491774**	**492037.6**	123.36
75×30S10	630259	636697	6898	634304	6986	636740	2087.7	630259	631982.2	78.17	637957	644569.3	113.86	**628585**	**629374.6**	152.93
75×30S15	613001	620356	7481	618688	7491	617984	1454.7	613001	614840.5	81.81	621149	626365.0	112.66	**611439**	**612324.0**	146.96
75×30S20	599329	600422	8697	601199	8389	604486	1773.0	599329	601011.5	84.85	605055	610181.4	114.44	**597785**	**598392.3**	140.36
75×30S30	577843	580490	9236	582766	9159	582388	1203.0	577843	579640.3	88.66	583551	586215.3	103.99	**576926**	**578071.2**	174.83
100×10S5	771172	773971	5571	773498	5339	777008	229.0	771172	771976.4	319.05	771368	774128.9	68.89	**771074**	771644.8	141.20
100×10S10	762282	764866	5024	763908	5399	766119	241.0	762282	763320.3	311.69	763469	765036.1	69.55	**761741**	**762470.3**	150.95
100×10S15	755040	757159	5832	757046	5333	757289	296.6	755040	755955.7	348.72	756236	758154.4	68.04	**754485**	**754810.0**	178.22
100×10S20	748611	750658	5150	750394	5160	751662	274.0	748611	749327.1	367.73	750047	751747.3	68.52	**748605**	**749108.6**	220.42
100×10S30	736248	738033	4375	737694	4886	739024	295.5	736248	737063.8	415.95	737505	739087.7	68.69	**735783**	**736235.5**	149.02
100×20S5	961900	966474	4920	970189	4500	970464	465.0	961900	964422.8	142.32	968861	973939.2	104.34	**961620**	**962419.6**	181.42
100×20S10	945835	951882	4474	949715	4940	958608	499.0	945835	948003.6	145.08	952317	958369.8	102.84	**945453**	**946951.9**	159.51
100×20S15	931525	935443	5047	936227	5007	936565	499.7	931525	933518.5	151.57	935246	940053.2	102.65	**930568**	**931289.5**	197.61
100×20S20	916505	921746	4938	922768	5203	925019	360.0	916505	918597.7	157.15	920851	923581.8	102.52	**916096**	**917149.3**	180.46
100×20S30	892755	894685	5555	896656	5332	903289	360.0	892755	894825.9	171.95	896202	899657.8	104.15	**892547**	**893693.6**	176.12
100×30S5	1154077	1170457	2818	1167044	2729	1173214	1814.0	1154077	1159790.8	121.82	1164617	1174570.3	141.40	**1150550**	**1152976.9**	248.14
100×30S10	1127161	1145700	4153	1142881	4307	1144807	1042.7	1127161	1130487.3	126.65	1140965	1149830.1	145.87	**1125044**	**1127892.4**	199.40
100×30S15	1103176	1113552	4482	1119040	4616	1121865	1116.0	1103176	1105138.4	130.40	1116441	1123978.2	146.01	**1101454**	**1104339.4**	196.40
100×30S20	1081933	1093126	4928	1096146	4806	1097480	943.0	1081933	1086086.7	133.10	1093174	1100436.0	143.75	**1080706**	**1082370.1**	222.37
100×30S30	1044499	1052682	5038	1057544	4958	1057666	900.0	1044499	1046736.6	140.48	1055745	1061462.5	138.23	**1043537**	**1045586.4**	200.54
AVG	501137.43	504253.5	5414.4	504448.9	5496.9	506013.3	518.1	501137.6	502307.1	117.4	504369.7	507363.0	70.5	**500605.7**	**501257.78**	117.7
#BEST	6	0		0		0		5			4			**48**		
p-value	1.2E-08	1.1E-09		1.1E-09		1.1E-09		8.3E-09	1.1E-09		7.6E-09	1.6E-09				

In the UCDAP, several incoming and outgoing trucks arrive at the yard of a cross-docking terminal, and each incoming/outgoing truck must be assigned to an inbound/outbound door. Each inbound/outbound door can accommodate at most one incoming/outgoing truck without considering any capacity requirement. The objective is to minimize the total pallet-handling cost inside the cross-docking terminal. Tsui & Chang (1990) first proposed the UCDAP. Several solution approaches have been proposed for solving the UCDAP, including the genetic algorithm (GA) proposed by Bermudez et al. (2001), the scatter search (SS) proposed by Tarhini et al. (2016), the greedy algorithm (GD) proposed by Zhang et al. (2018), and the ant colony system algorithm (ACS) proposed by Zhang et al. (2018). Since the UCDAP can be considered as a special case of the CDAP, the RL-SO algorithm can be directly applied to solve the UCDAP by setting the capacity of all doors and the amount of pallets in all trucks to the same value. To evaluate the RL-SO algorithm on the UCDAP, we used the same 40 instances in Zhang et al. (2018) and independently ran the RL-SO algorithm on each instance ten times under the same time limit as adopted in Zhang et al. (2018) (Table 8).

The QKPCG involves a knapsack with a capacity C and a set of items $V = 1, ..., n$, where each item i has a weight w_i and a profit p_i. Let E be the set of incompatible couples of items such that a couple of two items i and j belonging to E cannot be packed simultaneously in the knapsack (i.e., the disjunctive constraint). The objective of the problem is to select a subset $S \subseteq V$ of pairwise compatible items such that the total profit of the selected items is maximized while ensuring that the total weight of all selected items does not exceed the knapsack capacity C (i.e., the knapsack capacity constraint). The QKPCG was first proposed by Shi et al. (2017), and a variety of solution methods have been proposed in the literature to solve

Table 8: Time limit for each UCDAP instance set.

Instance set	#Instance	Time limit(s)	Instance set	#Instance	Time limit(s)
FC12	5	0.2	SC48	5	3
FC24	5	1	SC64	5	5
FC48	5	3	SC96	5	15
FC96	5	15	SC192	5	100
FC192	5	100	SC300	5	200

Table 9: Summarized comparisons of the RL-SO algorithm against the best-known values and the reference algorithms GA (Bermudez et al., 2001), SS (Tarhini et al., 2016), GD (Zhang et al., 2018), and ACS (Zhang et al., 2018) on the 40 UCDAP instances, the symbol '-' indicates that the corresponding results are not reported.

	Best				AVG			
Algorithm pair	#Win	#Tier	#Lose	p-value	#Win	#Tier	#Lose	p-value
RL-SO vs. BKS	**34**	6	0	2.1E-05	-	-	-	-
RL-SO vs. GD	**40**	0	0	1.8E-12	-	-	-	-
RL-SO vs. SS	**36**	4	0	1.7E-07	**40**	0	0	1.8E-12
RL-SO vs. GA	**36**	4	0	1.7E-07	**40**	0	0	1.8E-12
RL-SO vs. ACS	**34**	6	0	2.1E-05	**37**	3	0	1.1E-07

the problem, including the GLPK solver (Shi et al., 2017), the neighborhood search based metaheuristic (NSBM) proposed by Shi et al. (2017), the modified descent method based heuristic (MDM) proposed by Dahmani & Hifi (2021), and the population based search algorithm (PBSA) proposed by Dahmani et al. (2020). To adapt our RL-SO algorithm to solve the QKPCG, we made two minor modifications to the RL-SO algorithm while keeping other ingredients unchanged. Firstly, we extended the SO strategy in RL-SO for the CDAP to handle both the knapsack capacity constraint and the disjunctive constraint. Secondly, in the Q-learning procedure, a state is the latest chosen item, while an action consists of assigning an item to the knapsack. To evaluate our RL-SO algorithm on the QKPCG, we used the same 45 instances tested in Dahmani & Hifi (2021) and Dahmani et al. (2020). We independently ran RL-SO on each instance ten times under a time limit of 200 s, as adopted in Dahmani & Hifi (2021) and Dahmani et al. (2020).

Tables 9 and 10 provide a summary of the comparative results between the RL-SO algorithm and the state-of-the-art algorithms in the literature for the 40 UCDAP instances and 45 QKPCG instances, respectively. The reference results of the compared algorithms were obtained from the original paper. The first column in both tables shows the compared algorithm pair, where BKS in the first row denotes the best-known results reported in the literature. Columns "#Win", "#Tier" and "#Lose" present the number of instances for which the RL-SO algorithm obtained a better (#Win), equal (#Tier), or worse (#Lose) result compared to each reference algorithm in terms of the best solution value ("Column Best") and average solution value ("Column AVG"). Column "p-value" displays the p-values obtained from the Wilcoxon signed-rank test (with a significance level of 0.05).

As shown in Tables 9 and 10, our RL-SO algorithm outperforms the current best-known results on 34 and 3 instances for the UCDAP and the QKPCG, respectively. When compared with each reference algorithm, the RL-SO algorithm clearly shows an overall better performance in terms of the best solution value and average solution value. Moreover, the small p-value (< 0.05) indicates that the performance difference between RL-SO and each reference algorithm is statistically significant for both considered problems, further confirming the superiority of the RL-SO algorithm. Finally, Tables C.13 and C.14 in the Appendix C present the detailed results of RL-SO for the two considered problems.

5. Analysis

This section analyzes three essential components of the RL-SO algorithm: the Q-learning strategy that guides the search toward promising areas, the SO method that adaptively explores feasible and infeasible search spaces, and the mixed tabu strategy that prevents the search from falling into cycling.

Table 10: Summarized comparisons of the RL-SO algorithm against the best-known values and the reference algorithms GLPK (Shi et al., 2017), NSBM (Shi et al., 2017), MDM (Dahmani & Hifi, 2021), and PBSA (Dahmani et al., 2020) on the 45 QKPCG instances, the symbol '-' indicates that the corresponding results are not reported.

	Best				AVG			
Algorithm pair	#Win	#Tier	#Lose	p-value	#Win	#Tier	#Lose	p-value
RL-SO vs. BKS	**3**	42	0	0.11	-	-	-	-
RL-SO vs. GLPK	**45**	0	0	5.7E-14	-	-	-	-
RL-SO vs. NSBM	**37**	8	0	1.1E-07	**43**	2	0	1.6E-08
RL-SO vs. MDM	**15**	30	0	4.4E-04	-	-	-	-
RL-SO vs. PBSA	**3**	42	0	0.11	**36**	9	0	1.7E-07

5.1. Effect of the Q-learning strategy

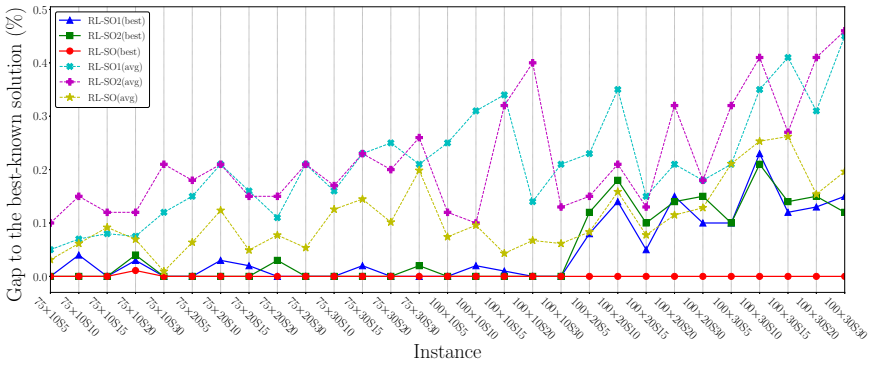

Figure 3: Comparisons of RL-SO with its variants RL-SO1 and RL-SO12.

Table 11: Wilcoxon signed-rank test for the results of each compared algorithm pair.

Algorithm pair	OF	AVG
RL-SO vs. RL-SO1	2.9E-04	4.8E-06
RL-SO vs. RL-SO2	1.5E-03	2.6E-06

As the core component of the RL-SO algorithm, a Q-learning strategy is applied to generate an initial solution for the SO procedure and guide the search toward promising regions (Section 3.2). The RL-SO algorithm was compared with two algorithmic variants (RL-SO1 and RL-SO2) to demonstrate the effectiveness of the Q-learning strategy. The RL-SO1 removes the Q-learning strategy and keeps the other components unchanged. A random multistart procedure was used for the RL-SO2 to replace the Q-learning framework, in which the initial solution for each round of the SO search was constructed by randomly assigning an incoming (outgoing) truck to an inbound (outbound) door. The three methods were compared using the 29 instances with the largest size under the same experimental protocol described in Section 4.1.

Figure 3 summarizes the comparative results between the RL-SO and its two variants. The RL-SO dominated the RL-SO1 and RL-SO2 in terms of the best and average objective values, respectively. For each instance, the RL-SO performed better than the other two variants. The RL-SO performed well in large instances with more than 75 incoming trucks. The p-values (<0.05) from the Wilcoxon signed-rank test shown in Table 11 indicated statistically significant differences between the RL-SO and its variants. This finding confirms the superiority of the RL-SO over the two variants.

Table 12: Comparative results on 29 "SetB" instances of RL-SO3 with RL-SO.

	RL-SO3			RL-SO		
Instance	OF	AVG	T(s)	OF	AVG	T(s)
75×10S5	438783	439451.2	**95.12**	**438575**	**438712.0**	108.32
75×10S10	434973	435326.2	**94.23**	**433876**	**434143.1**	126.28
75×10S15	430882	432452.3	**58.32**	**429758**	**430152.5**	115.27
75×10S20	427089	429521.3	**64.13**	**426431**	**426680.4**	138.27
75×10S30	419599	421325.1	**105.69**	**419526**	**419565.3**	110.55
75×20S5	529297	530123.5	165.21	**528769**	**529105.9**	**131.81**
75×20S10	520762	521953.8	101.16	**519927**	**520569.6**	**99.70**
75×20S15	512648	516346.6	**112.31**	**512114**	**512366.2**	121.13
75×20S20	505902	508352.3	150.21	**504434**	**504823.7**	**120.67**
75×20S30	492226	493296.4	135.19	**491774**	**492037.6**	**123.36**
75×30S10	631205	634325.1	167.36	**628585**	**629374.6**	**152.93**
75×30S15	614326	616365.4	**145.13**	**611439**	**612324.0**	146.96
75×30S20	598928	601325.6	**134.35**	**597785**	**598392.3**	140.36
75×30S30	579458	581362.6	**124.25**	**576926**	**578071.2**	174.83
100×10S5	772277	774235.6	**130.78**	**771074**	**771644.8**	141.20
100×10S10	763175	764568.9	**138.32**	**761741**	**762470.3**	150.95
100×10S15	755698	758594.8	**164.12**	**754485**	**754810.0**	178.22
100×10S20	750069	753641.7	**185.32**	**748605**	**749108.6**	220.42
100×10S30	736538	739846.1	251.32	**735783**	**736235.5**	**149.02**
100×20S5	964569	968357.4	210.29	**961620**	**962419.6**	**181.42**
100×20S10	950638	952879.3	168.36	**945453**	**946951.9**	**159.51**
100×20S15	933153	935647.1	201.21	**930568**	**931289.5**	**197.61**
100×20S20	919310	923567.6	**140.23**	**916096**	**917149.3**	180.46
100×20S30	897209	899368.1	196.74	**892547**	**893693.6**	**176.12**
100×30S5	1157235	1159635.1	265.85	**1150550**	**1152976.9**	**248.14**
100×30S10	1131514	1138521.8	**168.85**	**1125044**	**1127892.4**	199.40
100×30S15	1108038	1111237.3	**143.56**	**1101454**	**1104339.4**	196.40
100×30S20	1085633	1090654.9	263.12	**1080706**	**1082370.1**	**222.37**
100×30S30	1049995	1054364.8	209.46	**1043537**	**1045586.4**	**200.54**
AVG	727969.97	730574.07	**154.83**	**725489.03**	**726388.16**	159.04
#BEST	0	0	**16**	**29**	**29**	13
p-value	2.6E-06	2.6E-06	0.46			

5.2. Importance of the SO method

As discussed in Section 3.3.2, the RL-SO algorithm employs feasible and infeasible search mechanisms to explore the search space of the CDAP. The RL-SO was compared with an RL-SO variant (RL-SO3) to verify the importance of the mixed search mechanism. Only feasible solutions were allowed for the RL-SO variant. Any infeasible solution was penalized by setting the penalty coefficient φ of the extended evaluation function ((Equation (17), Section 3.3.2) to an extremely large value to implement the RL-SO3 variant.

Table 12 summarizes the comparative results between the RL-SO and the RL-SO3. The RL-SO dominated the RL-SO3 in terms of the best and average objective values. The small p-values confirmed the statistically significant difference between the RL-SO and the variant RL-SO3. In terms of the average computation time, the RL-SO required slightly more time than the RL-SO3 (159.04 s vs. 154.83 s) to reach the best solutions. However, the quality of the solution produced by the RL-SO was significantly better than that produced by the RL-SO3. This experiment demonstrated that the feasible and infeasible search mechanism positively contributes to the performance of the algorithm.

5.3. Importance of the mixed tabu strategy

The mixed tabu strategy described in Section 3.3.3 is a critical component of the proposed RL-SO algorithm. To demonstrate the benefits of this strategy, the RL-SO was compared with an RL-SO variant (RL-SO4), which relies only on the traditional tabu search strategy.

The two compared algorithms were tested on the 29 largest instances with at least 75 incoming trucks (Figure 4). Each algorithm was independently run ten times, and each run was given a time limit of $3|M|$ s to solve each instance. Figure 4 shows a statistically significant difference between the RL-SO and RL-SO4 in terms of the best and average objective values, indicated by the p-values from the Wilcoxon signed-rank test (4.31E-04 and 2.56E-06 in terms of the best and average objectives, respectively). This experiment indicates that the mixed tabu strategy positively contributes to the performance of the algorithm.

6. Conclusion

This study investigated the cross-dock door assignment problem (CDAP), a complex and vital decision problem in supply chain management. A flow based three indexed mixed integer programming (MIP) formulation was introduced for the CDAP. The formulation is based on the flow of goods from incoming trucks

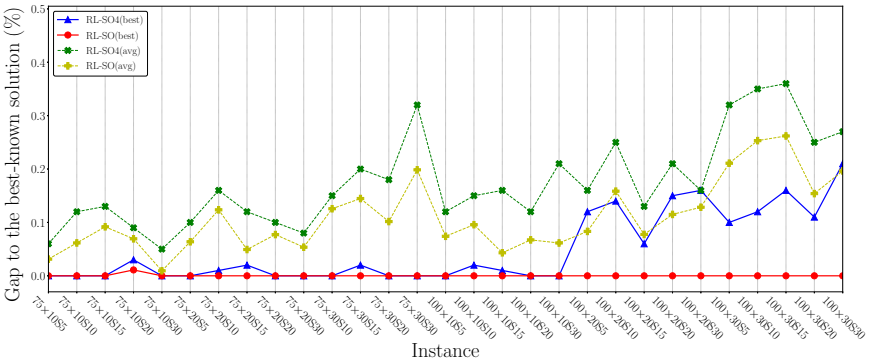

Figure 4: Comparisons between RL-SO with its variant RL-SO4

passing through inbound and outbound doors. It uses three indexed continuous variables rather than four indexed variables to tackle the quadratic term in the objective of the CDAP. This study proposed an effective reinforcement learning (RL) based local search algorithm (denoted by RL-SO) integrating a Q-learning procedure and a strategic oscillation (SO) approach. The Q-learning procedure uses the RL technique to gain useful information from historically visited solutions to produce new promising truck-to-door assignments. To further improve the learning abilities of the Q-learning procedure, a linear programming (LP) relaxation formulation based estimation was used to initialize the action-value function, significantly shortening the learning process. The SO approach adaptively explores feasible and infeasible search regions to obtain high-quality solutions. Moreover, it employs a mixed tabu strategy that takes advantage of the traditional attribute-based and solution-based tabu strategies to avoid being trapped in search cycling and missing high-quality solutions.

The performances of the FBMIP formulation and the proposed RL-SO algorithm were tested on two sets of benchmark instances in the literature. Compared with the best MIP formulation in the literature, the FBMIP enables solving eight more problem instances optimally and produces better LBs and UBs for the unsolved instances. Moreover, the proposed RL-SO algorithm proved to be highly competitive compared with state-of-the-art CDAP algorithms in the literature. In particular, it obtained improved best-known results (i.e., new UBs) for 43 of the 99 tested instances. To show the generality of the algorithm, we also tested it on two other well-studied problems, namely the Uncapacitated Cross-dock Door Assignment Problem (UCDAP) and the Quadratic Knapsack Problem with Conflict Graph (QKPCG), and highly competitive results were achieved by the algorithm.

Additional experiments were performed to study the impact of the essential components of the proposed RL-SO, confirming the usefulness of the Q-learning procedure, the feasible and infeasible search, and the mixed tabu strategy. Since these strategies have a general nature and significantly contribute to the performance of the proposed RL-SO algorithm, it would be interesting for future research to explore the application of the RL-SO algorithm to address other dock door assignment problems with different objective functions and practical operational rules, or other strongly constrained combinatorial optimization problems.

Acknowledgments

We would like to thank Prof. Monique Guignard, and Dr. Placide Nduwayo with his coauthors, for sharing the benchmark instances. We are grateful to the reviewers for their constructive comments, which helped us to significantly improve this paper. This work was partially supported by the National Natural Science Foundation Program of China (Grant No. 72122006).

References

Alicastro, M., Ferone, D., Festa, P., Fugaro, S., & Pastore, T. (2021). A reinforcement learning iterated local search for makespan minimization in additive manufacturing machine scheduling problems. *Computers & Operations Research*, *131*, 105272.

Bartholdi, J. J., & Gue, K. R. (2004). The best shape for a crossdock. *Transportation Science*, *38*, 235–244.

Bartholdi III, J. J., & Gue, K. R. (2000). Reducing labor costs in an ltl crossdocking terminal. *Operations Research*, *48*, 823–832.

Battiti, R., & Tecchiolli, G. (1994). The reactive tabu search. *ORSA journal on computing*, *6*, 126–140.

Bello, I., Pham, H., Le, Q. V., Norouzi, M., & Bengio, S. (2016). Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, .

Bengio, Y., Lodi, A., & Prouvost, A. (2021). Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, *290*, 405–421.

Benlic, U., Epitropakis, M. G., & Burke, E. K. (2017). A hybrid breakout local search and reinforcement learning approach to the vertex separator problem. *European Journal of Operational Research*, *261*, 803–818.

Bermudez, R., Cole, M. H., & Center, M.-B. T. (2001). *Genetic Algorithm Approach to Door Assignments in Breakbulk Terminals*. Technical Report.

Bertsekas, D. (2012). *Dynamic programming and optimal control: Volume I* volume 1. Athena scientific.

Birattari, M., Stützle, T., Paquete, L., Varrentrapp, K. et al. (2002). A racing algorithm for configuring metaheuristics. In *Gecco* (pp. 11–18). volume 2.

Bodnar, P., de Koster, R., & Azadeh, K. (2017). Scheduling trucks in a cross-dock with mixed service mode dock doors. *Transportation Science*, *51*, 112–131.

Boysen, N., & Fliedner, M. (2010). Cross dock scheduling: Classification, literature review and research agenda. *Omega*, *38*, 413–422.

Buijs, P., Vis, I. F., & Carlo, H. J. (2014). Synchronization in cross-docking networks: A research classification and framework. *European Journal of Operational Research*, *239*, 593–608.

Cai, Q., Hang, W., Mirhoseini, A., Tucker, G., Wang, J., & Wei, W. (2019). Reinforcement learning driven heuristic optimization. *arXiv preprint arXiv:1906.06639*, .

Chang, J., Wang, L., Hao, J.-K., & Wang, Y. (2021). Parallel iterative solution-based tabu search for the obnoxious p-median problem. *Computers & Operations Research*, *127*, 105155.

Chen, Y., Hao, J.-K., & Glover, F. (2016). An evolutionary path relinking approach for the quadratic multiple knapsack problem. *Knowledge-Based Systems*, *92*, 23–34.

Cohen, Y., & Keren, B. (2009). Trailer to door assignment in a synchronous cross-dock operation. *International Journal of Logistics Systems and Management*, *5*, 574–590.

Corberán, Á., Peiró, J., Campos, V., Glover, F., & Martí, R. (2016). Strategic oscillation for the capacitated hub location problem with modular links. *Journal of Heuristics*, *22*, 221–244.

Dahmani, I., & Hifi, M. (2021). A modified descent method-based heuristic for binary quadratic knapsack problems with conflict graphs. *Annals of Operations Research*, *298*, 125–147.

Dahmani, I., Hifi, M., Saadi, T., & Yousef, L. (2020). A swarm optimization-based search algorithm for the quadratic knapsack problem with conflict graphs. *Expert Systems with Applications*, *148*, 113224.

Erdoğan, G., & Tansel, B. (2007). A branch-and-cut algorithm for quadratic assignment problems based on linearizations. *Computers & Operations Research*, *34*, 1085–1106.

Galinier, P., & Hao, J.-K. (1999). Hybrid evolutionary algorithms for graph coloring. *Journal of combinatorial optimization*, *3*, 379–397.

Gaudioso, M., Monaco, M. F., & Sammarra, M. (2021). A lagrangian heuristics for the truck scheduling problem in multi-door, multi-product cross-docking with constant processing time. *Omega*, *101*, 102255.

Gelareh, S., Glover, F., Guemri, O., Hanafi, S., Nduwayo, P., & Todosijević, R. (2020). A comparative study of formulations for a cross-dock door assignment problem. *Omega*, *91*, 102015.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management science*, *40*, 1276–1290.

Gendreau, M., Laporte, G., & Semet, F. (1998). A tabu search heuristic for the undirected selective travelling salesman problem. *European Journal of Operational Research*, *106*, 539–545.

Gendreau, M., Soriano, P., & Salvail, L. (1993). Solving the maximum clique problem using a tabu search approach. *Annals of operations research*, *41*, 385–403.

Glover, F. (1997). Tabu search and adaptive memory programming—advances, applications and challenges. In *Interfaces in Computer Science and Operations Research* (pp. 1–75). Springer.

Glover, F., & Hao, J.-K. (2011). The case for strategic oscillation. *Annals of Operations Research*, *183*, 163–173.

Glover, F., & Lü, Z. (2021). Focal distance tabu search. *Science China Information Sciences*, *64*, 1–12.

Gu, X., Zhao, S., & Wang, Y. (2022). Reinforcement learning enhanced multi-neighborhood tabu search for the max-mean dispersion problem. *Discrete Optimization*, *44*, 100625.

Gue, K. R. (1999). The effects of trailer scheduling on the layout of freight terminals. *Transportation Science*, *33*, 419–428.

Guemri, O., Nduwayo, P., Todosijević, R., Hanafi, S., & Glover, F. (2019). Probabilistic tabu search for the cross-docking assignment problem. *European Journal of Operational Research*, *277*, 875–885.

Guignard, M., Hahn, P. M., Pessoa, A. A., & da Silva, D. C. (2012). Algorithms for the cross-dock door assignment problem. In *Proceedings of the Fourth International Workshop on Model-based Metaheuristics* (pp. 145–162).

Hanafi, S., Wang, Y., Glover, F., Yang, W., & Hennig, R. (2023). Tabu search exploiting local optimality in binary optimization. *European Journal of Operational Research*, .

He, Y., Jia, T., & Zheng, W. (2023). Tabu search for dedicated resource-constrained multiproject scheduling to minimise the maximal cash flow gap under uncertainty. *European Journal of Operational Research*, .

Hutter, F., Tompkins, D. A., & Hoos, H. H. (2002). Scaling and probabilistic smoothing: Efficient dynamic local search for sat. In *International Conference on Principles and Practice of Constraint Programming* (pp. 233–248). Springer.

Jiang, Y., Cao, Z., & Zhang, J. (2021). Learning to solve 3-d bin packing problem via deep reinforcement learning and constraint programming. *IEEE Transactions on Cybernetics*, (pp. 1–12).

Khalil, E., Dai, H., Zhang, Y., Dilkina, B., & Song, L. (2017). Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, *30*.

Laguna, M. (2018). Tabu search. In *Handbook of heuristics* (pp. 741–758). Springer.

Lai, X., Hao, J.-K., & Yue, D. (2019). Two-stage solution-based tabu search for the multidemand multidimensional knapsack problem. *European Journal of Operational Research*, *274*, 35–48.

Lai, X., Yue, D., Hao, J.-K., & Glover, F. (2018). Solution-based tabu search for the maximum min-sum dispersion problem. *Information Sciences*, *441*, 79–94.

Li, M., Hao, J.-K., & Wu, Q. (2022). Learning-driven feasible and infeasible tabu search for airport gate assignment. *European Journal of Operational Research*, *302*, 172–186.

Liu, R., Xie, X., & Garaix, T. (2014). Hybridization of tabu search with feasible and infeasible local searches for periodic home health care logistics. *Omega*, *47*, 17–32.

Lu, Y., Benlic, U., & Wu, Q. (2018). A memetic algorithm for the orienteering problem with mandatory visits and exclusionary constraints. *European Journal of Operational Research*, *268*, 54–69.

Lu, Z., Martínez-Gavara, A., Hao, J.-K., & Lai, X. (2023). Solution-based tabu search for the capacitated dispersion problem. *Expert Systems with Applications*, *223*, 119856.

Manchanda, S., Mittal, A., Dhawan, A., Medya, S., Ranu, S., & Singh, A. (2020). Gcomb: Learning budget-constrained combinatorial algorithms over billion-sized graphs. *Advances in Neural Information Processing Systems*, *33*, 20000–20011.

Martin-Santamaria, R., Sánchez-Oro, J., Pérez-Peló, S., & Duarte, A. (2022). Strategic oscillation for the balanced minimum sum-of-squares clustering problem. *Information Sciences*, *585*, 529–542.

Mazyavkina, N., Sviridov, S., Ivanov, S., & Burnaev, E. (2021). Reinforcement learning for combinatorial optimization: A survey. *Computers & Operations Research*, *134*, 105400.

Nassief, W., Contreras, I., & As' Ad, R. (2016). A mixed-integer programming formulation and lagrangean relaxation for the cross-dock door assignment problem. *International Journal of Production Research*, *54*, 494–508.

Oh, Y., Hwang, H., Cha, C. N., & Lee, S. (2006). A dock-door assignment problem for the korean mail distribution center. *Computers & Industrial Engineering*, *51*, 288–296.

Ojha, V. K., Abraham, A., & Snášel, V. (2017). Metaheuristic design of feedforward neural networks: A review of two decades of research. *Engineering Applications of Artificial Intelligence*, *60*, 97–116.

Peng, N., Xi, Y., Rao, J., Ma, X., & Ren, F. (2021). Urban multiple route planning model using dynamic programming in reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, *23*, 8037–8047.

Qin, J., Xu, X., Wu, Q., & Cheng, T. (2016). Hybridization of tabu search with feasible and infeasible local searches for the quadratic multiple knapsack problem. *Computers & Operations Research*, *66*, 199–214.

Sánchez-Oro, J., López-Sánchez, A., Hernández-Díaz, A., & Duarte, A. (2022). Grasp with strategic oscillation for the α-neighbor p-center problem. *European Journal of Operational Research*, *303*, 143–158.

Scavuzzo, L., Chen, F., Chételat, D., Gasse, M., Lodi, A., Yorke-Smith, N., & Aardal, K. (2022). Learning to branch with tree mdps. *Advances in Neural Information Processing Systems*, *35*, 18514–18526.

Sghir, I., Hao, J.-K., Jaafar, I. B., & Ghédira, K. (2015). A multi-agent based optimization method applied to the quadratic assignment problem. *Expert Systems with Applications*, *42*, 9252–9262.

Shi, X., Wu, L., & Meng, X. (2017). A new optimization model for the sustainable development: Quadratic knapsack problem with conflict graphs. *Sustainability*, *9*, 236.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.

Tang, Y., Agrawal, S., & Faenza, Y. (2020). Reinforcement learning for integer programming: Learning to cut. In *International conference on machine learning* (pp. 9367–9376). PMLR.

Tarhini, A. A., Yunis, M. M., & Chamseddine, M. (2016). Natural optimization algorithms for the cross-dock door assignment problem. *IEEE Transactions on Intelligent Transportation Systems*, *17*, 2324–2333.

Tsui, L. Y., & Chang, C.-H. (1990). A microcomputer based decision support tool for assigning dock doors in freight yards. *Computers & Industrial Engineering*, *19*, 309–312.

Tsui, L. Y., & Chang, C.-H. (1992). An optimal solution to a dock door assignment problem. *Computers & Industrial Engineering*, *23*, 283–286.

Van Belle, J., Valckenaers, P., & Cattrysse, D. (2012). Cross-docking: State of the art. *Omega*, *40*, 827–846.

Wang, Y., Pan, S., Li, C., & Yin, M. (2020). A local search algorithm with reinforcement learning based repair procedure for minimum weight independent dominating set. *Information Sciences*, *512*, 533–548.

Wang, Y., Wu, Q., & Glover, F. (2017). Effective metaheuristic algorithms for the minimum differential dispersion problem. *European Journal of Operational Research*, *258*, 829–843.

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*, 279–292.

Wei, Z., & Hao, J.-K. (2021). Multistart solution-based tabu search for the set-union knapsack problem. *Applied Soft Computing*, *105*, 107260.

Woodruff, D. L., & Zemel, E. (1993). Hashing vectors for tabu search. *Annals of Operations Research*, *41*, 123–137.

Wu, Q., Wang, Y., & Glover, F. (2020). Advanced tabu search algorithms for bipartite boolean quadratic programs guided by strategic oscillation and path relinking. *INFORMS Journal on Computing*, *32*, 74–89.

Zachariadis, E. E., Tarantilis, C. D., & Kiranoudis, C. T. (2009). A guided tabu search for the vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research*, *195*, 729–743.

Zhang, Y.-H., Gong, Y.-J., Chen, W.-N., Gu, T.-L., Yuan, H.-Q., & Zhang, J. (2018). A dual-colony ant algorithm for the receiving and shipping door assignments in cross-docks. *IEEE Transactions on Intelligent Transportation Systems*, *20*, 2523–2539.

Zhao, H., She, Q., Zhu, C., Yang, Y., & Xu, K. (2021). Online 3d bin packing with constrained deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 741–749). volume 35.

Zheng, J., He, K., Zhou, J., Jin, Y., & Li, C.-M. (2021). Combining reinforcement learning with lin-kernighan-helsgaun algorithm for the traveling salesman problem. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 12445–12452). volume 35.

Zhou, Y., Hao, J.-K., & Duval, B. (2016). Reinforcement learning based local search for grouping problems: A case study on graph coloring. *Expert Systems with Applications*, *64*, 412–422.

Zhu, Y.-R., Hahn, P. M., Liu, Y., & Guignard, M. (2009). New approach for the cross-dock door assignment problem. In *Proceedings of the XLI Brazilian Symposium on Operations Research* (pp. 1226–1236).

Appendix A. The CDAP formulation in the literature

We recall the $M^{2',0}$ formulation introduced by Gelareh et al. (2020):

$$\min \quad F = \sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{I}}\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}} D_{ij}T_{mn}z_{minj} \tag{A.1}$$

$$\text{s.t.} \quad (2)-(7),$$

$$\sum_{i\in\mathcal{I}} z_{minj} = y_{nj}, \forall m\in\mathcal{M}, n\in\mathcal{N}, j\in\mathcal{J} \tag{A.2}$$

$$\sum_{j\in\mathcal{J}} z_{minj} = x_{mi}, \forall m\in\mathcal{M}, n\in\mathcal{N}, i\in\mathcal{I} \tag{A.3}$$

$$0 \le z_{minj} \le 1, \forall m\in\mathcal{M}, n\in\mathcal{N}, i\in\mathcal{I}, j\in\mathcal{J} \tag{A.4}$$

Appendix B. Implementation of the mixed tabu strategy

Our mixed tabu strategy is a combination of the traditional attribute-based tabu strategy (Glover, 1997) and the solution-based tabu strategy (Woodruff & Zemel, 1993; Wang et al., 2017; Wu et al., 2020). The basic idea behind the solution-based tabu strategy involving marking each visited solution by multiple hash vectors and checking whether a candidate neighboring solution was previously visited by comparing it with each marked solution. Given a solution denoted by S, where $x_{mi} = 1$ if truck m is assigned to door i and $x_{mi} = 0$ if otherwise, we associate the assignment of truck m to door i with a dummy weight μ_{mi}. We adopt the following type of hash function h, whose hash values can be calculated easily.

$$h(S) = \Big(\sum_{m\in\mathcal{M}}\sum_{i\in\mathcal{I}}\mu_{mi}x_{mi} + \sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}}\mu_{nj}y_{nj}\Big) \mod L \tag{B.1}$$

where L is the length of the hashing vectors, and each dummy weight takes a prefixed value randomly generated in $[0, ..., 10000]$. We obtain the three hash functions, $h_1(S)$, $h_2(S)$, and $h_3(S)$, by setting different values of μ_{ij}.

The hash value of a neighboring solution S' by operating shift(m, i, i') move can be computed as follows:

$$h(S') = h(S) - \mu_{mi} + \mu_{mi'} \mod L \tag{B.2}$$

The time complexity to compute the hash function value of any neighboring solution induced by shift(m, i, i') or swap(m, m') is $O(1)$.

The hash value of a neighboring solution S' by operating multi-shift(i, i') move can be computed by:

$$h(S') = h(S) - B_{ii} + B_{ii'} \mod L \tag{B.3}$$

where $B_{ii'} = \sum_{m\in E_i}\mu_{mi'}$ presents the total weight obtained by assigning all trucks $m \in E_i$ to door i'. The time complexity to compute the hash function value of any neighboring solution in multi-shift(i, i') or multi-swap(i, i') is $O(1)$ by using B. B is updated in $O(|M||I|)$ time after each iteration.

We calculate the hash values $h_k(S')$ $(k = 1, 2, 3)$ of S' using Equation (B.2) or (B.3), and the indexes $h_k(S')$ of hash vectors H_k $(k = 1, 2, 3)$ are increased by one, i.e., the visiting time of solution S' is increased by one.

$$H_k[h_k(S')] = H_k[h_k(S')] + 1, k = 1, 2, 3 \tag{B.4}$$

For any truck m involved in the move, moveing m back to its original door i for the next tt iterations is forbidden.

$$tl(mi) = iter + tt \tag{B.5}$$

Only solutions meeting the following criteria are allowed: (1) $tl(mi) < iter$ and (2) $H_1[h_1(S')] < \beta$ or $H_2[h_2(S')] < \beta$ or $H_3[h_3(S')] < \beta$. Such solutions are defined as eligible solutions.

Appendix C. Computational results on UCDAP and QKPCG.

Table C.13: Comparative results on 40 UCDAP instances with GD (Zhang et al., 2018), SS (Tarhini et al., 2016), GA (Bermudez et al., 2001), and ACS (Zhang et al., 2018).

		GD	SS		GA		ACS		RL-SO		
Instance	BKS	OF	OF	AVG	OF	AVG	OF	AVG	OF	AVG	T(s)
FC12a	**19151.06**	19790.84	**19151.06**	19164.22	**19151.06**	19361.71	**19151.06**	19190.55	**19151.06**	**19151.06**	0.00
FC12b	**36643.11**	39000.21	**36643.11**	36661.25	**36643.11**	36882.78	**36643.11**	**36643.11**	**36643.11**	**36643.11**	0.00
FC12c	**56695.52**	60453.89	**56695.52**	56697.11	**56695.52**	56777.77	**56695.52**	**56695.52**	**56695.52**	**56695.52**	0.00
FC12d	**87732.20**	90425.56	**87732.20**	88312.19	**87732.20**	88120.47	**87732.20**	**87732.20**	**87732.20**	**87732.20**	0.00
FC24a	56349.74	73829.01	56660.19	61509.70	58960.66	61227.21	56349.74	56764.96	**56295.83**	**56295.83**	0.06
FC24b	138190.85	166894.92	138696.30	146341.73	140748.27	146973.05	138190.85	141568.18	**138190.85**	**138190.85**	0.02
FC24c	227278.56	271550.22	230661.22	234741.43	231110.10	234914.21	227278.56	230489.16	**227278.56**	**227278.56**	0.05
FC24d	340710.01	372523.80	342875.16	346313.55	344238.56	347422.15	340710.01	342590.87	**340710.01**	**340710.01**	0.07
FC48a	214678.32	291638.75	228944.22	240301.34	233844.41	249233.61	214678.32	222881.03	**208094.23**	**209159.74**	1.67
FC48b	563028.09	700302.36	572269.02	594699.07	598401.58	619560.75	563028.09	575359.25	**549417.03**	**553747.66**	1.42
FC48c	1051025.67	1216516.44	1055544.92	1086056.44	1086430.53	1104812.88	1051025.67	1063927.48	**1039297.50**	**1042290.10**	2.25
FC48d	1491426.12	1643617.72	1492960.38	1519199.33	1521410.64	1538809.15	1491426.12	1502554.95	**1481308.79**	**1482501.89**	1.64
FC96a	1146614.90	1611540.72	1176080.77	1237016.32	1315069.72	1366079.32	1146614.90	1192038.88	**1115370.06**	**1120371.91**	8.21
FC96b	2919418.33	3612130.64	2964379.62	3027248.75	3106296.61	3212751.50	2919418.33	2971180.02	**2866490.01**	**2878524.85**	6.44
FC96c	4782081.58	5509100.39	4789727.59	4856046.84	4963458.85	5061254.58	4782081.58	4812146.75	**4692190.09**	**4711156.22**	7.25
FC96d	7236850.63	7832802.77	7265340.38	7312004.01	7393625.95	7459814.73	7236850.63	7271295.54	**7188103.89**	**7192722.60**	7.87
FC192a	5178822.30	6798878.87	5209296.17	5270447.67	5796335.14	5946671.38	5178822.30	5243792.56	**4891863.53**	**4925118.50**	59.29
FC192b	11855968.63	13964856.36	11870904.81	11951968.07	12568533.12	12768173.58	11855968.63	11918365.21	**11540994.27**	**11558457.94**	63.48
FC192c	18609180.31	20722213.60	18648235.98	18764655.03	19354990.05	19613982.17	18609180.31	18713729.83	**18330991.61**	**18348774.78**	55.69
FC192d	27149822.94	28921027.07	27155541.75	27243191.59	27730048.36	27923306.76	27149822.94	27202833.20	**26903925.51**	**26919385.47**	35.73
SC48a	243029.61	287139.83	243659.41	247053.88	244863.69	251402.99	243029.61	245955.32	**242960.63**	**242960.63**	0.87
SC48b	559026.09	620569.20	560020.56	567203.62	562387.72	568986.22	559026.09	562369.29	**558890.55**	**558890.55**	0.27
SC48c	**949636.78**	1041376.04	951372.20	956636.16	955850.07	960743.09	**949636.78**	953221.73	**949636.78**	**949636.78**	0.26
SC48d	**1334062.19**	1410908.81	1335715.58	1339204.16	1336933.98	1343062.22	**1334062.19**	1338859.53	**1334062.19**	**1334062.19**	0.39
SC64a	446653.50	548131.48	446653.50	455557.26	455187.34	464936.15	447630.84	452989.42	**445411.12**	**445935.82**	2.32
SC64b	1025689.93	1191997.35	1028973.38	1041385.88	1037583.36	1050611.90	1025689.93	1033733.93	**1024634.17**	**1024676.14**	2.39
SC64c	1632907.57	1804211.30	1633641.39	1643659.05	1642391.08	1649550.25	1632907.57	1637768.39	**1629865.46**	**1630201.05**	2.39
SC64d	2257359.48	2374841.05	2257335.06	2267789.10	2266725.65	2274744.09	2257359.48	2264420.02	**2256563.85**	**2256629.96**	3.10
SC96a	1093760.68	1294226.62	1093760.68	1112837.80	1120573.50	1138542.86	1097244.29	1114206.19	**1087201.96**	**1087394.26**	6.33
SC96b	2501125.05	2849290.12	2506841.11	2533649.08	2537052.25	2560055.06	2501125.05	2522159.93	**2499351.20**	**2500021.61**	10.02
SC96c	3854540.75	4164282.98	3867741.11	3889926.09	3892131.30	3916201.93	3854540.75	3880443.27	**3852744.90**	**3853090.52**	8.67
SC96d	5380388.39	5711507.32	5382754.44	5407996.25	5411594.57	5439763.81	5380388.39	5410317.62	**5380130.63**	**5380361.62**	8.36
SC192a	5777475.55	7118906.79	5800374.05	5877762.37	5970586.38	6046223.64	5777475.55	5852008.89	**5726836.88**	**5728903.87**	58.46
SC192b	12921073.98	14642126.67	12928441.98	13021704.61	13103982.22	13237988.15	12921073.98	13002310.47	**12852429.05**	**12854418.54**	49.76
SC192c	19736893.76	21414021.44	19736893.76	19835367.87	19915108.44	20045848.79	19738043.91	19825934.94	**19653535.27**	**19656832.40**	58.75
SC192d	27910134.02	29452942.33	27935918.04	28026272.43	28111883.59	28173283.24	27910134.02	28006066.81	**27860099.45**	**27861921.84**	54.37
SC300a	4320268.66	17032541.30	14431764.18	14561572.14	14864949.90	15054465.63	4320268.66	14489309.81	**14276938.29**	**14285147.95**	143.62
SC300b	31913740.31	35669638.69	31923025.74	32155457.57	32571902.72	32741426.25	31913740.31	32093159.21	**31816411.01**	**31842164.15**	126.21
SC300c	50021193.15	53981808.28	50057517.64	50212334.74	50534100.07	50737469.08	50021193.15	50162149.78	**49872756.00**	**49949773.12**	156.68
SC300d	68299555.48	71478678.01	68376779.86	68469274.90	68714408.02	68877012.86	68299555.48	68411154.51	**68228377.88**	**68237238.63**	116.68

Table C.14: Comparative results on 45 QKPCG instances with GLPK (Shi et al., 2017), NSBM (Shi et al., 2017), MDM (Dahmani & Hifi, 2021), and PBSA (Dahmani et al., 2020).

		GLPK	NSBM		MDM	PBSA		RL-SO		
Instance	BKS	OF	OF	AVG	OF	OF	AVG	OF	AVG	T(s)
1qkpcg1	**17071**	15482	16642	6280.20	17061	**17071**	16859.50	**17071**	**17071.00**	98.14
1qkpcg2	**13500**	11774	12272	12006.20	**13500**	**13500**	13369.80	**13500**	**13500.00**	56.88
1qkpcg3	**16156**	15129	15843	15612.60	**16156**	**16156**	16111.50	**16156**	**16156.00**	113.02
1qkpcg4	**19921**	17151	19659	19594.20	**19921**	**19921**	19806.50	**19921**	**19921.00**	59.01
1qkpcg5	**16870**	15147	15499	15378.80	**16870**	**16870**	16843.70	**16870**	**16870.00**	123.21
2qkpcg1	**11217**	9018	11205	10895.00	11172	**11217**	11181.00	**11217**	**11217.00**	87.84
2qkpcg2	**10933**	9150	10565	10438.20	10911	**10933**	10842.60	**10933**	**10933.00**	52.32
2qkpcg3	11248	10002	11163	11084.80	11232	11248	11218.10	**11312**	**11312.00**	78.62
2qkpcg4	**15599**	12589	**15599**	**15599.00**	**15599**	**15599**	**15599.00**	**15599**	**15599.00**	27.10
2qkpcg5	**12580**	10810	**12580**	12541.60	**12580**	**12580**	12520.60	**12580**	**12580.00**	77.51
3qkpcg1	**7455**	5843	**7455**	7337.00	**7455**	**7455**	**7455.00**	**7455**	**7455.00**	5.31
3qkpcg2	**7343**	6068	**7343**	**7343.00**	**7343**	**7343**	**7343.00**	**7343**	**7343.00**	59.10
3qkpcg3	**7285**	6091	7258	7200.60	**7285**	**7285**	**7285.00**	**7285**	**7285.00**	66.84
3qkpcg4	**8006**	7978	7991	7988.60	**8006**	**8006**	**8006.00**	**8006**	**8006.00**	95.95
3qkpcg5	**7350**	6290	**7350**	7156.40	**7350**	**7350**	7307.60	**7350**	**7350.00**	87.65
4qkpcg1	**20726**	18025	19457	18931.40	20401	**20726**	20116.40	**20726**	**20726.00**	80.90
4qkpcg2	**21677**	17049	19240	18749.00	**21677**	**21677**	21406.80	**21677**	**21677.00**	59.69
4qkpcg3	**21953**	15835	20864	20538.20	21897	**21953**	21332.20	**21953**	**21953.00**	128.42
4qkpcg4	**31123**	26428	31082	30737.80	**31123**	**31123**	**31123.00**	**31123**	**31123.00**	72.07
4qkpcg5	**20792**	15768	18727	18083.40	20788	**20792**	20660.60	**20792**	**20792.00**	100.96
5qkpcg1	**15517**	10663	14871	14745.00	15353	**15517**	15189.90	**15517**	**15517.00**	79.74
5qkpcg2	15529	10720	15068	14956.20	15529	15529	15511.10	**15594**	**15594.00**	63.80
5qkpcg3	**15370**	10846	15132	14946.20	15178	**15370**	15086.10	**15370**	**15370.00**	77.24
5qkpcg4	**18954**	13639	18380	18272.20	**18954**	**18954**	18938.80	**18954**	**18954.00**	122.98
5qkpcg5	**15715**	10883	15607	15313.20	**15715**	**15715**	15702.20	**15715**	**15715.00**	74.55
6qkpcg1	**8969**	6032	**8969**	8840.00	8712	**8969**	8688.00	**8969**	**8969.00**	113.67
6qkpcg2	**9658**	6041	9193	9098.80	**9658**	**9658**	9403.50	**9658**	**9658.00**	79.77
6qkpcg3	**8578**	6429	8508	8504.40	**8578**	**8578**	8555.10	**8578**	**8578.00**	83.05
6qkpcg4	**9657**	7366	9630	9156.40	**9657**	**9657**	**9657.00**	**9657**	**9657.00**	50.15
6qkpcg5	**9338**	5866	9335	9335.00	**9338**	**9338**	**9338.00**	**9338**	**9338.00**	102.84
7qkpcg1	**27010**	18900	24602	24187.80	**27010**	**27010**	26659.40	**27010**	**27010.00**	75.89
7qkpcg2	**30343**	22545	28686	28354.60	30290	**30343**	30115.50	**30343**	**30343.00**	108.75
7qkpcg3	**26685**	15637	23420	22444.60	**26685**	**26685**	26497.40	**26685**	**26685.00**	4.74
7qkpcg4	**34110**	27146	32641	32350.20	**34110**	**34110**	33687.20	**34110**	**34110.00**	80.62
7qkpcg5	**25947**	18714	22735	21762.40	**25947**	**25947**	25815.40	**25947**	**25947.00**	85.55
8qkpcg1	**18668**	12388	18583	18583.00	18628	**18668**	18331.90	**18668**	**18668.00**	83.79
8qkpcg2	**20021**	13221	19620	19406.20	19841	**20021**	19813.50	**20021**	**20021.00**	76.97
8qkpcg3	17911	11054	17469	16998.60	**17911**	17911	17806.90	**17974**	**17974.00**	25.59
8qkpcg4	**21168**	13060	20715	20524.00	**21168**	**21168**	**21168.00**	**21168**	**21168.00**	97.09
8qkpcg5	**18757**	12960	18480	18000.40	**18757**	**18757**	18701.50	**18757**	**18757.00**	78.94
9qkpcg1	**11180**	6823	**11180**	11180.00	**11180**	**11180**	10914.00	**11180**	**11180.00**	100.45
9qkpcg2	**10598**	7673	10577	10534.20	**10598**	**10598**	10338.60	**10598**	**10598.00**	59.83
9qkpcg3	**11022**	5291	10869	10852.60	**11022**	**11022**	10849.30	**11022**	**11022.00**	87.70
9qkpcg4	**11751**	8241	**11751**	11308.00	**11751**	**11751**	11707.00	**11751**	**11751.00**	15.09
9qkpcg5	**10761**	6792	10713	10602.40	10743	**10761**	10517.10	**10761**	**10761.00**	110.35